Introduction to Mobile Programming Assignment -3

Mobile application which records and displays the orientation of the smartphone based on accelerometer data

Sardor Hazratov
Computer Engineering student at
Yildiz Technical University
Istanbul , Turkey
serdarjan1995@gmail.com
2017

***Abstract*—Obtaining the orientation of the smartphone is not hard job and there are several solutions. One of them is using data adopted from accelerometer. Accelerometer gives us 3 float numbers which they are x-axis, y-axis and z-axis. While the phone is put somewhere on flat surface one of these axes shows the value of the gravity. By comparing all data from sensor we obtain the orientation of the smartphone.**

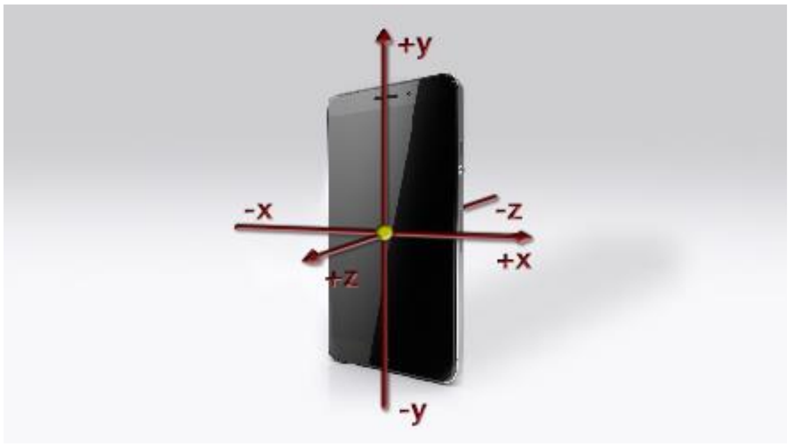
Keywords—Android; mobile phone; accelerometer; orientation of the smartphone; sensor;

I. ACCELEROMETER AND SMARTPHONE

The accelerometer is a new technology which has upgraded the user's experience in touch screen smart devices like mobiles and tablet PCs. The main function of this is to adapt the orientation change when the position is changed from vertical to horizontal and vice-versa. This technology actually measures acceleration, in other words the rate of change velocity. Accelerometer of the smartphone is used in many applications. It is used regularly in games like controlling vehicle or moving the object.

II. OBTAINING THE ORIENTATION OF THE SMARTPHONE VIA ACCELEROMETER

In quiet position one of the axes that accelerometer data shows us must be equal to the gravity of the Earth, nearly 9.8 m/s^2 with small precision error. So detecting which axis is that help us to find orientation. But firstly we need to have a look at how accelerometer data is represented. The picture below shows the axes of accelerometer on the smartphone:



Picture 1: Accelerometer axes

III. Desingning the application

Designing relies on imagination. In this assignment we use simple features of Android Studio. We should read data from accelerometer and write them to a file. In this case our app should obtain permission to read data from sensor and permission to write to storage.

We must add these lines to AndroidManifest.xml

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-feature android:name="android.hardware.sensor.accelerometer" android:required="true"/>
```

Next job is represent data obtained from accelerometer in graph. Unfortunately Android Studio does not provide libraries for graphs. In our solution we used library GraphView. It can be obtained from www.android-graphview.org and it provides documentation with examples to build up several types of charts. There is two way of using this library. We use mavev dependency features. We need to modify grandle object (Module:app):

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {
        exclude group: 'com.android.support', module: 'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:25.2.0'
    compile 'com.android.support:support-v4:25.2.0'
    compile 'com.jjoe64:graphview:4.2.1' //GraphView Library
    testCompile 'junit:junit:4.12'
}
```

Our layout:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="#dddddd">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <TextView
            android:id="@+id/textview_content_x"
            android:layout_width="120dp"
            android:layout_height="wrap_content"
            android:gravity="start"
            android:keepScreenOn="true"
            android:text="x="
            android:textColor="#50151d"
            android:textSize="18sp"
            android:layout_marginTop="11dp"
            android:layout_alignParentStart="true" />

        <TextView
            android:id="@+id/textview_content_y"
            android:layout_width="120dp"
            android:layout_height="wrap_content"
            android:gravity="start"
            android:keepScreenOn="true"
            android:text="y="
            android:textColor="#50151d"
            android:textSize="18sp"
            android:layout_alignParentStart="true"
```

```
                android:layout_marginTop="11dp" />



            <TextView
                android:id="@+id/textview_content_z"
                android:layout_width="120dp"
                android:layout_height="wrap_content"
                android:gravity="start"
                android:keepScreenOn="true"
                android:text="z="
                android:textColor="#50151d"
                android:textSize="18sp"
                android:layout_alignParentStart="true"
                android:layout_marginTop="11dp" />

        </LinearLayout>

        <TextView
            android:id="@+id/textview_orientation"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="PORTRAIT UP"
            android:gravity="center"
            android:textColor="#33b5e5"
            android:textSize="30sp"
            android:textStyle="bold"/>

        <com.jjoe64.graphview.GraphView
            android:layout_width="wrap_content"
            android:layout_height="150dp"
            android:id="@+id/graph"/>
        <RelativeLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:layout_marginTop="30dp">

            <Button
                android:id="@+id/exit_button"
                style="@style/Widget.AppCompat.Button.Colored"
                android:text="EXIT"
                android:layout_width="100dp"
                android:layout_height="wrap_content"
                android:background="@android:color/holo_green_light"
                android:textSize="20sp"
                android:layout_alignParentTop="true"
                android:layout_alignParentEnd="true"
                android:layout_marginEnd="60dp" />

            <Button
                android:text="Save to fıle"
                android:layout_width="100dp"
                android:layout_height="wrap_content"
                android:id="@+id/save_file_button"
                style="@style/Widget.AppCompat.Button"
                android:background="@android:color/holo_purple"
                android:textSize="18sp"
                android:layout_marginStart="60dp"
                android:layout_alignParentTop="true"
                android:layout_alignParentStart="true" />
        </RelativeLayout>
    </LinearLayout>
```

Now it is time for Java codes:

We have a class named `MyGraphPoints.` It is responsible for graph points y respecting to the sensors x, y, z axis, and graph points x for time.

```java
public class MyGraphPoints {
    private int x;
    private int y;
    private int z;
    private int t;

    public MyGraphPoints(int x, int y, int z, int t){
        setT(t);
        setX(x);
        setY(y);
        setZ(z);
    }

    public int getX() {
        return x;
    }

    public void setX(int x) {
        this.x = x;
    }

    public int getY() {
        return y;
    }

    public void setY(int y) {
        this.y = y;
    }

    public int getZ() {
        return z;
    }

    public void setZ(int z) {
        this.z = z;
    }

    public int getT() {
        return t;
    }

    public void setT(int t) {
        this.t = t;
    }

    public String getAll(){
        return "X:"+x+" Y:"+y+" Z:"+z+" T:"+t;
    }
}
```

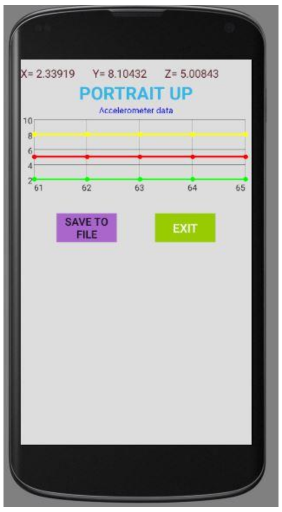
Main Activity:

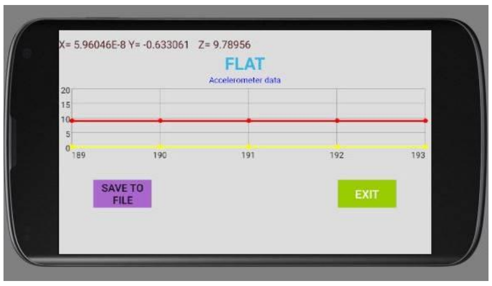
```java
import android.annotation.SuppressLint;
import android.content.Context;
import android.graphics.Color;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Environment;
import android.support.v7.app.ActionBar;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.os.Handler;
import android.text.format.DateFormat;
import android.util.Log;
import android.view.MotionEvent;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.jjoe64.graphview.GraphView;
import com.jjoe64.graphview.series.DataPoint;
import com.jjoe64.graphview.series.LineGraphSeries;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.Date;

/**
 * An example full-screen activity that shows and hides the system UI (i.e.
 * status bar and navigation/system bar) with user interaction.
 */
public class PhoneOrientationPortrait extends AppCompatActivity implements SensorEventListener{
    private SensorManager sMgr;
    private Sensor accelerometer;
    private TextView textviewContentX;
    private TextView textviewContentY;
    private TextView textviewContentZ;
    private TextView textviewOrientaion;
    private GraphView graph;
    private Button exitButton;
    private Button saveButton;
    private boolean writeEnabled=false;
    private LineGraphSeries<DataPoint> seriesX;
    private LineGraphSeries<DataPoint> seriesY;
    private LineGraphSeries<DataPoint> seriesZ;
    private ArrayList<MyGraphPoints> data = new ArrayList<MyGraphPoints>();
    private int currentTime=-1;
    private int columnsOnGraph=0;
    private File file;
    private OutputStream outputStream;

    private static final boolean AUTO_HIDE = true;
    private static final int AUTO_HIDE_DELAY_MILLIS = 3000;
    private static final int UI_ANIMATION_DELAY = 300;
    private final Handler mHideHandler = new Handler();
    private View mContentView;
    private final Runnable mHidePart2Runnable = new Runnable() {
        @SuppressLint("InlinedApi")
        @Override
        public void run() {
            mContentView.setSystemUiVisibility(View.SYSTEM_UI_FLAG_LOW_PROFILE
                    | View.SYSTEM_UI_FLAG_FULLSCREEN
                    | View.SYSTEM_UI_FLAG_LAYOUT_STABLE
                    | View.SYSTEM_UI_FLAG_IMMERSIVE_STICKY
                    | View.SYSTEM_UI_FLAG_LAYOUT_HIDE_NAVIGATION
                    | View.SYSTEM_UI_FLAG_HIDE_NAVIGATION);
        }
    };
    private final Runnable mShowPart2Runnable = new Runnable() {
        @Override
        public void run() {
            ActionBar actionBar = getSupportActionBar();
            if (actionBar != null) {
                actionBar.show();
            }
        }
    };
    private boolean mVisible;
    private final Runnable mHideRunnable = new Runnable() {
        @Override
        public void run() {
            hide();
        }
    };
    private final View.OnTouchListener mDelayHideTouchListener = new View.OnTouchListener() {
        @Override
        public boolean onTouch(View view, MotionEvent motionEvent) {
            if (AUTO_HIDE) {
                delayedHide(AUTO_HIDE_DELAY_MILLIS);
            }
            return false;
        }
    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_phone_orientation);
        mVisible = true;
        mContentView = findViewById(R.id.textview_orientation);
        mContentView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                toggle();
            }
        });

        sMgr = (SensorManager)this.getSystemService(SENSOR_SERVICE);
        accelerometer = sMgr.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
        sMgr.registerListener(this,accelerometer,1000000);
        textviewContentX = (TextView)findViewById(R.id.textview_content_x);
        textviewContentY = (TextView)findViewById(R.id.textview_content_y);
        textviewContentZ = (TextView)findViewById(R.id.textview_content_z);
        textviewOrientaion = (TextView)findViewById(R.id.textview_orientation);
        exitButton = (Button)findViewById(R.id.exit_button);
        exitButton.setOnClickListener(new View.OnClickListener(){

            @Override
            public void onClick(View view) {
                destroyApp();
            }
        });
        saveButton = (Button)findViewById(R.id.save_file_button);
```

```java
    saveButton.setOnClickListener(new View.OnClickListener(){

        @Override
        public void onClick(View view) {
            if(writeEnabled){
                writeEnabled=false;
                saveButton.setText("Save to file");
                saveButton.setBackgroundColor(Color.parseColor("#ffaa66cc"));
            }
            else{
                writeEnabled=true;
                saveButton.setText("Stop Writing");
                saveButton.setBackgroundColor(Color.RED);
            }

        }
    });
    graph = (GraphView) findViewById(R.id.graph);
    graph.setTitle("Accelerometer data");
    graph.setTitleColor(Color.BLUE);

    if(isExternalStorageWritable()){
        file = new File(getExternalFilesDir(null), "orientations_log.txt");
        try{
            outputStream  = new FileOutputStream(file);

        }
        catch (Exception e){
            saveButton.setEnabled(false);
            Toast.makeText(this,"Error File creating",Toast.LENGTH_LONG);
        }
    }

}

@Override
protected void onPostCreate(Bundle savedInstanceState) {
    super.onPostCreate(savedInstanceState);
    delayedHide(100);
}

private void toggle() {
    if (mVisible) {
        hide();
    } else {
        show();
    }
}

private void hide() {
    ActionBar actionBar = getSupportActionBar();
    if (actionBar != null) {
        actionBar.hide();
    }
    mVisible = false;

    mHideHandler.removeCallbacks(mShowPart2Runnable);
    mHideHandler.postDelayed(mHidePart2Runnable, UI_ANIMATION_DELAY);
}

@SuppressLint("InlinedApi")
private void show() {
    mContentView.setSystemUiVisibility(View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN
            | View.SYSTEM_UI_FLAG_LAYOUT_HIDE_NAVIGATION);
    mVisible = true;
```

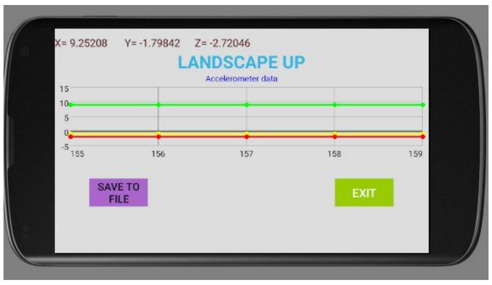
```java
        mHideHandler.removeCallbacks(mHidePart2Runnable);
        mHideHandler.postDelayed(mShowPart2Runnable, UI_ANIMATION_DELAY);
    }

    private void delayedHide(int delayMillis) {
        mHideHandler.removeCallbacks(mHideRunnable);
        mHideHandler.postDelayed(mHideRunnable, delayMillis);
    }


    @Override
    public void onSensorChanged(SensorEvent sensorEvent) {
        float x = sensorEvent.values[0];
        float y = sensorEvent.values[1];
        float z = sensorEvent.values[2];
        graph.removeAllSeries();

        if (accelerometer.getType() == Sensor.TYPE_ACCELEROMETER) {
            textviewContentX.setText("X= " + x);
            textviewContentY.setText("Y= " + y);
            textviewContentZ.setText("Z= " + z);
        }
        if( (z>8.40f && z<10.9f) && (y>-2.30f && y<2.30f)){
            textviewOrientaion.setText("FLAT");
        }
        else if( (z<-8.40f && z>-10.9f) && (y>-2.30f && y<2.30f)){
            textviewOrientaion.setText("FLAT UPSIDEDOWN");
        }
        else if( (y>-4.30f && y<4.30f) && (z>-3.30f && z<3.30f) && (x>8.30f && x<10.9f) ){
            textviewOrientaion.setText("LANDSCAPE UP");
        }
        else if( (y>-4.30f && y<4.30f) && (z>-3.30f && z<3.30f) && (x<-8.30f && x>-11.1f) ){
            textviewOrientaion.setText("LANDSCAPE DOWN");
        }
        else if( (y>8.0f && y<11.8f) && (z>-3.30f && z<3.30f) && (x<3.75f && x>-3.75f) ){
            textviewOrientaion.setText("PORTRAIT UP");
        }
        else if( (y<-8.0f && y>-11.8f) && (z>-3.30f && z<3.30f) && (x<3.75f && x>-3.75f) ){
            textviewOrientaion.setText("PORTRAIT DOWN");
        }
        currentTime++;
        data.add(new MyGraphPoints((int)x,(int)y,(int)z,currentTime));
        if(outputStream!=null && writeEnabled){
            Date d = new Date();
            CharSequence s  = DateFormat.format("EEEE, MMMM d, yyyy ", d.getTime());
            String wrt = s.toString();
            wrt+=" orientation: "+textviewOrientaion.getText()+" x: "+x+" y: "+y+" z: "+z+"\n";
            try {
                outputStream.write(wrt.getBytes());
            } catch (IOException e) {
                Toast.makeText(this,"Error File writing",Toast.LENGTH_LONG);
            }
        }

        if(columnsOnGraph<5){
            columnsOnGraph++;
        }
        if(currentTime>2){
            seriesX = new LineGraphSeries<>(makeDataX(columnsOnGraph,data));
            seriesX.setTitle("X");
            seriesX.setColor(Color.GREEN);
            seriesX.setDrawDataPoints(true);
            seriesX.setDataPointsRadius(8);
            seriesX.setThickness(6);
            graph.addSeries(seriesX);

            seriesY = new LineGraphSeries<>(makeDataY(columnsOnGraph,data));
            seriesY.setTitle("Y");
            seriesY.setColor(Color.YELLOW);
            seriesY.setDrawDataPoints(true);
            seriesY.setDataPointsRadius(8);
            seriesY.setThickness(6);
            graph.addSeries(seriesY);

            seriesZ = new LineGraphSeries<>(makeDataZ(columnsOnGraph,data));
            seriesZ.setTitle("Z");
            seriesZ.setColor(Color.RED);
            seriesZ.setDrawDataPoints(true);
            seriesZ.setDataPointsRadius(8);
            seriesZ.setThickness(6);
            graph.addSeries(seriesZ);
        }

    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int i) {
        sMgr.registerListener(this,accelerometer,1000000);
    }

    protected void onResume(){
        super.onResume();
        sMgr.registerListener(this,accelerometer,1000000);
    }

    protected void onPause(){
        super.onPause();
        sMgr.unregisterListener(this);
    }

    protected void destroyApp(){
        try {
            outputStream.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
        sMgr.unregisterListener(this);
        finish();
        System.exit(0);
    }

    public DataPoint[] makeDataX(int count, ArrayList<MyGraphPoints> data){
        DataPoint[] dt = new DataPoint[count];
        int dtIndex=0;
        for(int i=data.size()-count; i<data.size(); i++) {
            dt[dtIndex++]=new DataPoint(data.get(i).getT(), data.get(i).getX());
        }
        return dt;
    }

    public DataPoint[] makeDataY(int count, ArrayList<MyGraphPoints> data){
        DataPoint[] dt = new DataPoint[count];
        int dtIndex=0;
        for(int i=data.size()-count; i<data.size(); i++) {
            dt[dtIndex++]=new DataPoint(data.get(i).getT(), data.get(i).getY());
        }
        return dt;
    }
    public DataPoint[] makeDataZ(int count, ArrayList<MyGraphPoints> data){
        DataPoint[] dt = new DataPoint[count];
        int dtIndex=0;
        for(int i=data.size()-count; i<data.size(); i++) {
            dt[dtIndex++]=new DataPoint(data.get(i).getT(), data.get(i).getZ());
        }
        return dt;
    }

    public boolean isExternalStorageWritable() {
        String state = Environment.getExternalStorageState();
        if (Environment.MEDIA_MOUNTED.equals(state)) {
            return true;
        }
        return false;
    }
}
```

Some screenshots of application:


X= 2.33919 Y= 8.10432 Z= 5.00843


X= 5.96046E-8 Y= -0.633061 Z= 9.78956

Picture 2b: Application screenshot on flat position


X= 9.25208 Y= -1.79842 Z= -2.72046

Picture 2c: Application screenshot in landscape position

Picture 2a: Application screenshot in portrait position

References:
http://www.techulator.com/resources/8930-How-does-smart-phone-accelerometer-work.aspx
http://www.android-graphview.org/simple-graph/