Mobile and Sensor Systems: Lecture 8 Practical: Mobile Phone Programming

Neal Lathia
Overview

- Quick summary of Android
- Examples of collecting sensor data
- Usage of the ES Sensor Manager library
- Programming exercise
Android Programming

- Android = Java
- Basic components
  - Activities, Services
  - Sensors
  - Alarms
  - Threads, AsyncTasks
- More details at:
listener = new SensorEventListener()
{

    // This method is required by the API and is called when the
    // accuracy of the
    // readings being generated by the accelerometer changes.
    // We don't do anything when this happens.
    public void onAccuracyChanged(Sensor sensor, int accuracy)
    {
    }

    // This method is called when the accelerometer takes a reading:
    // despite the name, it is called whether even if it's the same as
    // the previous one
    public void onSensorChanged(SensorEvent event)
    {
        try
        {
            if (isSensing)
            {
                synchronized (sensorReadings)
                {
                    if (isSensing)
                    {
                        float[] data = new float[3];
                        for (int i = 0; i < 3; i++)
                        {
                            data[i] = event.values[i];
                        }
                        sensorReadings.add(data);
                        sensorReadingTimestamps.add(System.currentTimeMillis());
                    }
                }
            }
        }
    catch (Exception e)
        {
            e.printStackTrace();
        }
    }
};
Emotion Sense Sensor Library

- Android sensor programming: each sensor API has different methods

- Emotion Sense Sensor Library
  - Provides an easy way to access sensor data
  - Supports a large number of sensors
  - Simple API with two modes (get, subscribe)
  - Includes energy efficient sensing modules
Pull Sensors

Accelerometer, Location, Microphone
Wi-Fi, Bluetooth, Camera
Active apps, SMS/Call Log Content

Push Sensors

Battery, Connection State
Proximity, Screen
Phone Calls/SMS Events
// 1. get the instance

ESSensorManager sm =
    ESSensorManager.getSystemService(context)

// 2. ask for some data

MicrophoneData data
    = (MicrophoneData) sm.getDataFromSensor(
        SensorUtils.SENSOR_TYPE_MICROPHONE)
// 1. make a subscription
int sid = sm.subscribeToSensorData(
    SensorUtils.SENSOR_TYPE_MICROPHONE, listener)

// 2. deal with data pushed to you
class Listener implements SensorDataListener
{
    public void onDataSensed(SensorData d){..}
    public void onCrossingLowBatteryThreshold(..) {..}
}
Demo
Exercise Requirements

• Android phone, USB cable

• Eclipse / Android Studio
  • Configure these for Android development, e.g. by installing the ADT plugin for Eclipse

• Demo project
  • https://github.com/xsenselabs (for ES library)
  • https://github.com/xsenselabs/ESLibrary-Examples/tree/master/SensorManagerLectureDemo

• ES Library and Android Docs
  • https://github.com/xsenselabs/SensorManager/tree/master/docs
Exercise 1

GOAL: Detect whether the user is moving or stationary

1. Implement an activity and add a button to the user interface.

2. On clicking the button, the program should detect whether the user is moving or not
Exercise 1

• Classification technique:

1. Capture data from the accelerometer sensor for 4 seconds (Use sensorManager.setSensorConfig() to set SENSE_WINDOW_LENGTH_MILLIS to 4 seconds)

2. Calculate the magnitude of acceleration for each of the vectors; \( m_i = \sqrt{x_i^2 + y_i^2 + z_i^2} \)

3. Calculate the Std. Deviation of the magnitudes. If (Std. Deviation > threshold) then the user is moving
Exercise 2

Detect the presence of environmental noise

- Capture data for about 3 seconds from the Microphone sensor
- Calculate the average of the amplitude values
- If (average > threshold) then status = ‘noise’
Exercise 3

Detect the presence of environmental noise after detecting the ‘isNear’ event on the proximity sensor

• Subscribe to the proximity sensor using ESSensorManager

• In the `onDataSensed()` method of the listener, check if `isNear()` is true and then invoke the noise detection procedure.
Questions

Contact: Neal Lathia
neal.lathia@cl.cam.ac.uk