What can Android Sense for you?

Adam Stroud and Greg Milette
1/16/2013
About Us

Greg: Consultant  
Creator of  
Digital Recipe  
Sidekick App  

Adam: Lead Android Developer for Runkeeper

We wrote this book
What is a "Sensor"

A capability that can capture measurements about the device and its external environment.
Sooo many sensors...

Camera
Microphone
NFC Scanner
Speech Recognition
Physical Sensors
Location Service
Why use Sensors?

Android Sensors can:

- *Hear* claps and singing
- *See* Android Logos
- *Understand* obscure spoken language
- *Scan* for NFCs (and do cool stuff)
- *Locate* a *device*
- *Determine* device position
Location Service

Using Android to determine where you are
Android Location Service

Provides location based functionality in Android

- Determine Device Location
  - Latitude
  - Longitude
  - Altitude

- Geocoding
  - Address-to-location translation

- Proximity Alerts
  - Notifications when device enters a specified area
Sources of Location Data

• A location provider is a source of location information

• Android has "three-ish" location providers
  ○ Network Provider
  ○ Makes use of Wifi access points and mobile network
  ○ GPS Provider
  ○ Uses GPS hardware on device
  ○ Passive Provider
  ○ Uses whatever other apps are currently using
Network Provider

- **Wifi Access Points**
  - MAC addresses and strength of nearby access points recorded

- **Mobile Network**
  - Uses distance/strength of cell towers

- **Queries Google Location Service**
  - Different from local location service
  - Data is somewhat crowd-sourced
GPS Provider

• Uses on-board GPS hardware along with global GPS system
• Most phones take advantage of A-GPS
  ○ Assisted GPS (A-GPS)
  ○ GPS information is downloaded using mobile network
How GPS Works

- GPS receiver contacts multiple GPS satellites
- Data is transmitted from satellite to GPS receiver
- Distance from satellite is computed using transmission time and speed of radio signal
- Distance from multiple satellites are used to determine position
Problems with GPS

• Signal can face interference
  ○ Environmental conditions
  ○ Vegetation
  ○ Atmospheric conditions
  ○ Signals travel slower though gases

• Not all phones have quality GPS hardware
  ○ Low power GPS hardware can cause slow location fix

• Need clear line of sight to sky
  ○ Unlikely to work indoors

• Multipath Problems
GPS Multipath

Direct Signals

Reflected Signals

By GPS_tracking_satellites.jpg: Vaughan Weather Navstar-2.jpg: NASA Canyon_midday.jpg: Realbrvhrt at en.wikipedia derivative work: Javiersanp [CC-BY-SA-3.0 (http://creativecommons.org/licenses/by-sa/3.0)], via Wikimedia Commons
<table>
<thead>
<tr>
<th></th>
<th>GPS Provider</th>
<th>Network Provider</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to First Fix (TTFF)</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Power Consumption</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Accuracy</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Supports Altitude</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>Supports Bearing</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>Supports Speed</td>
<td>True</td>
<td>True</td>
</tr>
</tbody>
</table>
Location Permissions

• Use of the location service requires Android permission(s)

• `ACCESS_COARSE_LOCATION`
  ○ Network Provider

• `ACCESS_FINE_LOCATION`
  ○ Network Provider
  ○ GPS Provider
  ○ Passive Provider

Note: No need to include multiple permissions to use Network and GPS providers
Demo: Get Current Location

- Use all enabled providers
- Displays information about location
  - Latitude
  - Longitude
  - Time to fix
  - Provider of location information
- Allows user to enable/disable location provider
Location Service API

- **LocationManager**
  - System service that provides access to location information

- **LocationListener**
  - Interface containing callback methods for processing location events

- **Location**
  - Contains location data from provider

- **LocationProvider**
  - Representation of the source of location data
Requesting Location Data

- Implement LocationListener
  - onLocationChanged()
  - onProviderDisabled()
  - onProviderEnabled()
  - onStatusChanged()
- Register LocationListener with LocationManager
- Process Location object in onLocationChanged()
- Unregister LocationListener
private LocationManager locationManager;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.current_location);

    locationManager = (LocationManager) getSystemService(LOCATION_SERVICE);
}

@Override
protected void onResume()
{
    // Retrieve only providers that user has enabled
    enabledProviders = locationManager.getProviders(true);

    for (String enabledProvider : enabledProviders) {
        // Request location information from provider.
        // The current class implements LocationListener
        locationManager.
        requestLocationUpdates(enabledProvider, 0, 0, 0, this);
    }
}
@Override
public void onLocationChanged(Location location)
{
    // Read location data and update display
    latValue.setText(String.valueOf(location.getLatitude()));
    long.setText(String.valueOf(location.getLongitude()));

    providerValue.setText(String.valueOf(location.getProvider()));
    accuracyValue.setText(String.valueOf(location.getAccuracy()));

    // Compute time to fix and update display
    long timeToFix = SystemClock.uptimeMillis() - uptimeAtResume;
    timeToFixValue.setText(String.valueOf(timeToFix / 1000));
}
@Override
protected void onPause()
{
    super.onPause();

    // Remove listener from location manager
    locationManager.removeUpdates(this);
}
Summary

• Android provides multiple sources of location data
• Location API is relatively simple to use
• Requesting location data can affect battery life
• Choice of location provider depends needs of app
Physical Sensors

Allowing Android to sense its place in the world
Sensors and Smartphones

- Previously, disjoint, separate pieces of hardware
- Now, unified on a single device that is mobile
- Use of these sensors allows apps to inject contextual based information to their algorithms
Types of Sensors

• Hardware (Raw) Sensors
  ◦ Provide raw data from a sensor
  ◦ Represents data from a single physical sensor

• Software (Synthetic/Virtual) Sensors
  ◦ Provides abstraction layer on top of raw sensors
  ◦ Combine data of multiple raw sensors
  ◦ Modifies raw sensor data to simplify consumption
  ◦ Different devices may have different implementations
Hardware vs. Software

Data-set was captured with device laying flat on its back.
Hardware + Filter Example

Raw Data

Filtered Data
Types of Sensor Data

Device sensors can provide three types of data:

- Environmental
  - Monitor conditions of the external environment
- Motion
  - Detect/determine the movement of a device
- Position
  - Determine the position and orientation of a device
Environmental Sensors

• Ambient Temperature
  ○ Room Temperature

• Ambient Light
  ○ Illumination

• Atmospheric Pressure

• Relative ambient air humidity

• Device Temperature
  ○ Device temperature
  ○ Worked differently across devices
  ○ Deprecated in favor of Ambient temperature
Demo: Live Sensor Data

Light Sensor

Pressure Sensor
Motion Sensors

- **Accelerometer**
  - Force and direction of acceleration (3-axis)

- **Gravity (Software)**
  - Isolates force of gravity by passing accelerometer data through a low-pass filter

- **Linear Acceleration (Software)**
  - Isolates acceleration data by passing accelerometer data through a high-pass filter
Motion Sensors Cont.

- Gyroscope
  - Angular speed around an axis (rate of rotation)
- Rotation Vector (Software)
  - Uses accelerometer, magnetometer and gyroscope to determine orientation of device
Coordinate Systems

Device Coordinate System

Global Coordinate System

Demo: Detecting Movement

- Detects movement using accelerometer and linear acceleration sensors
- Conditionally passes data through a high-pass filter
- Computes total acceleration to detect movement (same algorithm can be used to detect shake)
Position Sensors

- Magnetic field
  - Geomagnetic field for x, y and z axis
- Proximity
  - How close an object is to the front of a device
- Orientation (Software, deprecated)
  - Computes the azimuth, pitch and roll of a device
Demo: Proximity Sensor

Proximity Sensor

Sensor List
- GP2A Light sensor
- GP2A Proximity sensor
- BMP180 Pressure sensor
- MPL Gyroscope
- MPL Accelerometer
- MPL Magnetic Field
- MPL Orientation
- MPL Rotation Vector
- MPL Linear Acceleration
- MPL Gravity
- Rotation Vector Sensor
Demo: Determine Orientation

- Use different approaches to determine if device is face-up or face-down
- Provide insight to data
Demo: North Finder

- Indicates when phone's camera is pointed within 20 degrees of north.
- Basis for augmented reality app.
- Direction of camera is determined using the rotation vector sensor.
Problems with Sensor Data

- **Drift**
  - Slow wandering of values that are read
- **Noise**
  - Random fluctuation of a measured value
- **Zero Offset (Bias)**
  - Constant value applied to sensor readings
- **Time Delays/Dropped Data**
  - A busy device can cause incorrect timestamps, or dropped data.
Handling Sensor Error

• Re-zeroing
  ◦ Re-calibrate offset that is applied to sensor data

• Sensor Fusion
  ◦ Combining data from multiple sensors

• Filters
  ◦ Low-Pass
  ◦ Filters out high-frequency noise
  ◦ High-Pass
  ◦ Emphasizes higher-frequency/transient components

• Use of software sensors
  ◦ Many already use fusion and/or filtering
Summary

- Android provides multiple different sensors which apps can utilize
- Prefer software sensors over hardware sensors
- Sensor API usage pattern is very similar to the Location API usage
- After you access the sensor data, the real work begins
Audio Analysis

Goal: Analyze audio recordings captured from microphone

Analyze:
• Amplitude only
• Raw audio
Example: Clapper
int getMaxAmplitude()

Returns the maximum absolute amplitude that was sampled since the last call to this method.
MediaRecorder recorder = prepareRecorder();

while (continueRecording) {
    waitClipTime();
    int maxAmplitude =
        recorder.getMaxAmplitude();
    continueRecording = process(maxAmplitude);
}
Recorded Audio
Example: Guitar Tuner
private double rootMeanSquared(short[] nums)
{
    double ms = 0;
    for (int i = 0; i < nums.length; i++)
    {
        ms += nums[i] * nums[i];
    }
    ms /= nums.length;
    return Math.sqrt(ms);
}
Estimate Frequency: Zero Crossing
Demonstration
Images

Goal: Analyze images from camera
Image: How it works

- Control the camera
  - Focus
  - Work with phone hardware

- Process image efficiently
  - Make it smaller
  - Convert to black and white

- Detect
  - Find biggest continuous block
Converting to gray

RgbAbsDiffGray radg = new RgbAbsDiffGray(Color.GREEN);

Gray8Threshold g8t = new Gray8Threshold(-48, true);

mSeqThreshold = new Sequence(radg);

mSeqThreshold.add(g8t);
Demo: Logo detection
Speech Commands

Goal: Understand your spoken commands

Challenge: Recognize hard to recognize words
Collect speech with: RecognizerIntent

Intent intent = new Intent
(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);

intent.putExtra
(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
RecognizerIntent.LANGUAGE_MODEL_WEB_SEARCH);

intent.putExtra
(RecognizerIntent.EXTRA_PROMPT,"Speak");
Android collects speech using dialogs and beeps
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == VOICE_RECOGNITION_REQUEST_CODE) {
        if (resultCode == RESULT_OK) {
            List<String> heard =
                data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
            //Your code here
        }
    }
}
Challenge:

Example recognition results:

"how much human"
"how much for a min"
"how much cannon"
"how much Human"
"how much planning"
Phonetic Matching

Cumin (C550)
Cumen (C550)
Kingman (K525)
Komen (K550)
Canon (C550)
Cannon (C550)
Human (H550)

Time (T5000)
Thyme (T500)
Whine (W500)
Mind (M530)
Demos

Android Sensor Playground
NFC

Goal: Quick access to features

How:
- Write custom tag data
- Register to start when user scans tag
Characteristics of NFC tags

• Different storage sizes
  ○ Not much (Enough for a URL)

• Robustness
  ○ Survive a washer cycle?
  ○ Sticker
private NdefMessage createNdefFromJson() {
    String mimeType = "application/root.gast.speech.activation";
    byte[] mimeBytes = mimeType.getBytes(Charset.forName("UTF-8"));
    byte[] id = new byte[0];
    byte[] data = new byte[0];
    NdefRecord record =
        new NdefRecord(NdefRecord.TNF_MIME_MEDIA, mimeBytes, id, data);
    NdefMessage m = new NdefMessage(new NdefRecord[] { record });
    return m;
}
Respond to tag scan with MIME type

`<activity android:name=".speech.activation.SpeechActivationNfcTagReceiver" />

<intent-filter>
  <action android:name="android.nfc.action.NDEF_DISCOVERED" />
  <category android:name="android.intent.category.DEFAULT" />
  <data android:mimeType="application/root.gast.speech.activation" />
</intent-filter>
</activity>`
NFC Demo

IT assets tracking
Combinations of sensors: NFC, Speech Timer

1. Scan NFC
2. Trigger speech recognition
3. Timer goes off and says the time
Great Android Sensing Toolkit (GAST)

Code:
http://www.github.com/gast-lib

App (the name is Android Sensor Playground):
Contact Info

• Adam Stroud
  ○ Twitter: @adstro
  ○ Email: adam.stroud@gmail.com

• Greg Milette
  ○ Twitter: @gradisontech
  ○ Email: gregorym@gmail.com