Sensors

Sensors

- While smartphones differ from other devices in terms of their compution power and battery limitations they do have one significant advantage
  - They generally come packed with sensors
  - Used to measure various aspects of the current environment
  - In this lecture we will introduce the various sensor types, coordinate systems and also the issues that surround sensors

Why sensors are useful

- Sensors permit a device to collect data about its current environment
  - While the data is basic in nature it can be used to provide useful information about what a user is doing at any given time
  - Examples in the latest android devices include support for heart rate monitors and step counters
  - To be able to track users and inform them how to improve their health
  - This is only one example but there is many more

Scalar sensors

- Sensors in android are divided into two specific types
  - The first type we deal with are scalar sensors
  - A scalar sensor is a sensor that reads a single property about the environment
  - That property has no directional component. e.g. ambient temperature has no direction
  - Android has four main types of sensor here

Light sensors

- The light sensor is used to measure the ambient light level in an environment
  - Light levels are measured in units of lux
  - 0 represents complete darkness while increasing values represent increasing brightness. A well lit room will approximately be 300 to 500 lux
  - One of the main uses of a sensor like this is to automatically adjust screen brightness to adapt to the overall lighting level

Temperature sensors

- The temperature sensor is used to measure the ambient temperature of the current environment
  - A sensor type that is not found very often on devices
  - Temperature is measured in celcius or farhenheit
  - While there is no direct use for something like this. A possible idea for its use is to use lots of distributed temperature sensors to map local fluctuations in temperature

Proximity sensors

- Proximity sensors measure how close an object is to the sensor
  - Unit of measurement here is the centimeter
  - Generally used to disable/enable a touchscreen when a user moves the device to their ear or away from their ear
  - Also used when a device is put face down on a table to silence the device from accepting any calls

Barometric sensors

- Barometric sensors are used to measure athmospheric pressure
  - Like the temperature sensors they are not found on many devices
  - The unit of measurement here is the millibar, higher values indicate a higher pressure
  - Similar to the temperature sensors it only really makes sense if there are many of them in use for mapping local changes in air pressure

Vector sensors

- Vector sensors are sensors that measure data in more than one dimension
  - All vector sensors in Android will measure data in three dimensions
  - These sensors tend to be the most common in android devices
  - As they provide much information about the environment

Accelerometers

- An accelerometer is used to measure accelerations in all three directions
  - Acceleration being the rate in change of velocity. Higher absolute values here state that the device is undergoing quicker changes in speed
  - The SI unit here is meters per second squared (ms^{-2}) but it is also common to use G-force units as well ($1g = 9.81\ ms^{-2}$)
  - There are many uses for a sensor like this a few examples of which will be covered later

Gyroscopes

- Gyroscopes are used to measure the speed of rotation of the device
  - The unit measurement here is radians per second. Higher absolute values represent faster device rotation.
  - Another of the common sensors provided on Android devices
  - Generally used in combination with the accelerometer to generate a device orientation

Magnetic field sensors

- Magnetic field sensors are used to detect the magnetic field strength of the earth in all three axis
  - The unit of measurement here is the micro-Tesla (μT), higher values here will mean that the magnetic field is stronger.
  - Generally something like this is used for getting device orientation if the gyroscope is not available
  - However more recent uses include magnetic covers for devices such that a high magnetic value is screen off and a reduction in magnetic value is screen on.

Virtual sensors

- Using sensors like the accelerometer directly may not be the best approach.
  - In the case of the accelerometer you will always have a gravity component that you may not need in your application.
  - In this case android provides a series of virtual sensors that will provide filtered versions of data and common combinations of sensors
  - Here we look at the gravity, linear acceleration, and rotation vector virtual sensors

Linear accelerometer

- The linear accelerometer is a high pass filter applied to the values that come back from the accelerometer
  - The high pass filter will remove the gravity component from the accelerometer as it is constant and has a frequency of zero
  - This will only leave the accelerations on the device that are caused by the user
  - This is generally used in place of the raw accelerometer

Gravity sensor

- The gravity sensor on the other hand is the accelerometer with a low pass filter applied
  - Thus removing all of the user induced accelerations on the device
  - Generally there will not be much in the way of use for a sensor like this, although one use could be to created a bubble application to check if things are level
  - Generally more interested in removing this

Rotation vector

- The rotation vector is a combination of the accelerometer and gyroscope that is used to determine a three axis orientation of a device
  - Has a different coordinate system to the other sensors (will be explained later)
  - Will produce a set of values that will enable the device to determine its compass bearing
  - And also the pitch and roll of the device

Location sensors

- Perhaps some of the most important sensors within a device like this are the location sensors
  - To be able to locate a device on the surface of the planet
  - Generally used for navigation purposes
  - Also used for anti-theft measures

GPS

- The most accurate of the location sensors that uses the Global Positioning System satellite signals
  - Can provide accuracy to within a few meters
  - Generally used for navigation however is generally assisted by the rotation vector for compass information
  - However, does not work indoors

Network location

- “location system” that uses a database that google maintains of network access points (wifi and mobile)
  - Provides an approximation of the location of the device.
  - However unlike GPS it works both indoors and outdoors
  - Generally the two will be combined to provide a full location for the device indoor and outdoor

How sensors are managed in android

- Sensors are all managed through a series of sensor classes
  - To hide the individual differences in how each sensor is managed and to present a unified interface to the developer
- All applications must register interest in sensors directly with the Android OS.
  - Again in an effort to reduce battery usage android will deliver updates at the rate requested by each individual listener rather than having each application poll the sensor continiously

Permissions

- Getting access to the majority of sensors does not require any form of permissions
  - Very difficult to do any kind of malicious activity just by reading gyroscopes and accelerometers
- However, for the location sensors there is explict permissions required
  - As an application can track the user's location and that information a user may not wish to disclose to an application.

Sensors need to be queried

- You are not guarenteed that a specific sensor will appear on any device.
  - While a lot of sensors appear on a majority of devices there will be some like temperature sensors and barometric sensors that have very little use and may not be in a certian device
- As a result you are expected to query if a given sensor is on an device.
  - The test to do this is to ask for a default sensor of the type you are interested in. If a value of null is returned then the device has no sensors of that type

Sensor manager class

- The class that you must interface with in order to access sensors and to register listeners with sensors
  - You will query this service for all of the sensors you require in your application
  - Once you have determined what sensors you require you will register listeners for those sensors with this service
  - It will then take responsibility for triggering your listeners whenever a new sensor value is obtained

Sensor class

- Contains all the information about each sensor that a device has
  - Includes all the constants that differentiate between the different sensor types
  - You can use this to query the specification of individual sensors and for comparing them
    - Generally you will do this if you have access to more than a single type of sensor and you are looking for the best one

Sensor listeners

- Small segments of java code that you provide for reacting to a change in a sensor
  - Changes include new values in which case you may need to update your application and the UI
  - Also will include changes in the accuracy of the sensor.
    - Gives you the oppertunity to change to another sensor if there is another one available or to change your UI to take account of the loss in accuracy

Sensor delay types

- There are three main sensor delay types that are used in application development. It is possible for you to define your own rates as well
  - SENSOR_DELAY_FASTEST: get updates from the sensor as fast as the sensor can physically provide them
  - SENSOR_DELAY_GAME: get updates from a sensor at a rate that is appropriate for game processing
  - SENSOR_DELAY_UI: get updates from a sensor at a rate that is suitable for updating a display

Where sensors fit with the activity life cycle

- It is generally recommended that you should only register interest in sensors when your application is not in the stopped state
  - Generally a user is not interested in the sensor values that you application is tracking if the application is not active
  - Thus we usually deregister our listeners in the onStop() method and reregister in the onStart() method as a way of conserving battery power
  - Unless you have a good reason to keep a sensor active while your application is not active

GPS and the activity life cycle

- The same advice would also apply to the location sensors with a few minor exceptions
  - The main exception being navigation
  - This is because if a navigation application is interrupted and removes location listeners it will have to wait about 5 to 10 seconds before the listener will get a new location after being reregistered
  - Otherwise you should deregister interest as GPS does consume battery power

Issues to consider with sensors

- While sensors do provide a lot of information about the environment and your user there are issues to consider with their use
  - These must be considered carefully
  - As they will affect the trust of your users and also the amount of information you collect on them
  - And also the performance of your application and the device it is running on

User tracking

- One of the most important ones is the use of location sensors to track the users location
  - While there are legitimate and common sense uses for these sensors it is not uncommon to see abuses of this data
  - As it is easily possible to track a user's full movements throughout a long period of time
  - If you are going to track your user you must inform them up front and give them a chance to opt out

Clarity on data use

- Like permissions you should be fully clear on what data you use from sensors and for what use.
  - Applications that do this are generally not malicious and are more likely to be trusted by users.
  - If you collect such data from your user particularly location data and store it on a server somewhere you must provide secure storage for that data
  - Generally anonymising and scrubbing identifing information away from data collected is a good step to take

Battery usage

- You must play a delicate balancing act between sensor accuracy and performance and battery power
  - The more you query a sensor the quicker you will recieve updates and will likely improve the accuracy of your application.
  - The downside however is that your application will consume more battery power as a result
  - If you go the opposite way you will save battery power with the potential loss of accuracy

Sensor accuracy

- Another issue to consider is that every sensor will have different accuracies and also value ranges
  - e.g. the proximity sensor in the HTC One X is a very basic sensor that only provides two values 0cm and 8cm
  - Other sensors will provide a higher accuracy but they will come at a higher expense
  - Don't assume anything with regards to the accuracy and error rates of any sensor you use on an android device

When to use multiple sensors of the same type

- Some devices will provide more than one of the same sensor type
  - It is possible to register interest in more than one sensor of the same type
  - Generally you will only need to do this if you need a higher accuracy than you could get from a single sensor
  - By averaging or filtering between two or more sensors

Sensor values are not absolute

- As you saw with the proximity sensor example earlier the values you get from the sensor will not be absolute
  - While a sensor measures a continous variable in general the sensor will represent them in discret values.
  - Thus there will be an error associated between what the sensor reports and the actual value of that property in the environment
  - Usually more expensive sensors will have more discrete levels leading to a larger values that can be represented

Sensor coordinate system

- Most sensors with the exception of the location and rotation vector sensors will use the same coordinate system
  - Where the Z axis is coming out of the device screen
  - The Y axis is coming out of the top of the device
  - And the X axis is coming out of the right hand side of the device

Rotation coordinate system

- As the rotation vector provides a series of rotations it uses a different coordinate system
  - The z axis is perpendicular to the ground and rotation around this axis is the compass direction
  - The X axis is pointing out from the right of the device and rotations around this axis represent the pitch of the device
  - The Y axis is pointint out of the top of the device and rotations around this axis represent the roll of the device

Location coordinate system

- The location sensors take advantage of the world standard latitude and longitude system of surface coordinates
  - Takes the earth and maps the surface as a 2D plane
  - With latitude representing how far south or north the device is [-90, +90]
  - And longitude representing how far west or east the device is [-180, +180]