Mandatory Project 2 – Efficient Position Updating

This project should help you learn to implement and evaluate systems based on techniques for pervasive positioning for efficient position updating for mobile phones.

You should describe your programs and experiments in a short report (3-4 pages plus figures). The report should include a short description of your design and implementation choices, and a description of how to use the programs. I must be able to run your client programs on an Android Phone and server programs on a Daimi Linux machine (unless you have made special arrangements with me). You should also describe and illustrate the experiments you have performed.

The report must be handed in December 16, 14.00. Upload the report as a single pdf-file and your code + kml files as a zip file. Files must be named "GroupX.pdf" and GroupX.zip" where X is replaced by your group number.

The project will not be graded but you should be able to present your work and the underlying theory and algorithms at the oral exam.

The project should be done in groups of 2-3 students.

Part I

You should implement the following client programs on an Android Phone. You can use any programming language you like, but the programs must run on an Android Phone. I would propose to use either Java or Python for ASE.

Mandatory: Make a client program that implements a periodic reporting strategy with a configurable time period between fixes. The program should set up a TCP connection or do a HTTP post to a server program on a configured IP address and port. When the strategy receives a GPS fix from the built-in GPS it should send a position update over the connection to the server. The program should also log the number of GPS fixes to a file stored on the phone.

Mandatory: Extend the client program to implement a distance-based reporting strategy for a configurable distance. The strategy should observe the distance between the GPS fixes it receives from the built-in GPS and send a position update to the server only if the distance threshold has been passed. You have to use requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0, locationListener) and filter the GPS positions in your own code.

Mandatory: Extend the client program to implement distance-based reporting strategy that requests as few GPS fixes as possible from the built-in GPS by assuming that the phone can only move with a configurable maximum speed.

Mandatory: Extend the client program to implement a distance-based reporting strategy that requests as few GPS fixes as possible from the built-in GPS by only using the GPS for distance-based reporting when accelerometer readings suggest that the device is moving.

Optional: Extend the client program with a strategy that uses both the maximum speed and accelerometer based optimizations.

Optional: Extend the client program with a version that instead of the maximum speed uses the GPS estimated speed to minimize the number of GPS fixes.

You should implement the following server program. You can use any programming language you like, but the program must run on a Daimi Linux machine (unless you have made special arrangements with me).

Mandatory: Make a server program that can accept a TCP connection or HTTP post and log incoming GPS fixes to a KML file where each GPS fix is represented as time-stamped placemarks and logs a count of the number of uplink messages to a file (from phone to server).

Part II

In the second part of this project you should design and implement experiments to examine the number of GPS fixes and uplink position updates needed to track a target. You might want to extend your client program with a small GUI where you can choose strategy and set configurable parameters. You should

Mandatory: Collect generated logs and KML files at the server and logs on the phone for the following scenarios assuming that you are tracking a pedestrian target. For each scenario you should collect at least 10 minutes of data on a route that you can recognize in Google Earth (e.g. along the parking lots on the roof of Storcenter nord). Make sure that you walk more than 200 meters away from the starting point. You should include a few minutes stopping time in the walk.

Run the client with the periodic reporting strategy with a time interval of 1 second.

Run the client with the distance-based reporting strategy with distance configured as 50 meters.

Run the client with the maximum-speed distance-based reporting strategy with distance configured as 50 meters and maximum speed as 10.44m/s (Usain Bolt, Berlin 2009).

Run the client with the maximum-speed distance-based reporting strategy with distance configured as 50 meters and maximum speed as 2 m/s.

Run the client with the movement-aware distance-based reporting strategy with distance configured as 50 meters.

Mandatory: Create screenshots using Google Earth for each of the scenarios of the collected KML files. The screenshots should include a path that marks the actually walked route. Comment on the results in the report and discuss how GPS errors impacted the results.

Mandatory: Make a list with the following entries for each scenario: strategy, number of GPS fixes, number of uplink messages, time span, GPS fixes per second, uplink messages per second and comment on them in the report with

respect to relevant literature. Discuss what pervasive positioning applications the different strategies are relevant for.