RSSI alapú helymeghatározás
- gyakorlat –

Összeállította:

Moldován István

Tóth László

1 Bevezetés

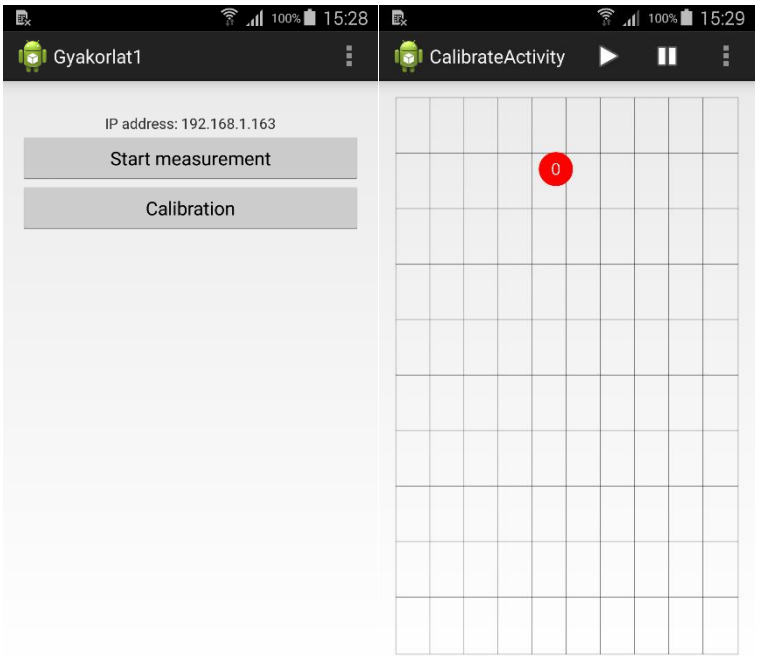
A labor célja a WiFi RSSI alapú helymeghatározás vizsgálata. A gyakorlat célja, hogy megismerjük a jelerősség mérésének kihívásait, megvizsgáljuk a jelerősség változását valós környezetben, valamint hogy megvalósítsunk egy helymeghatározási módszert RSSI alapon.

2 Mérési összeállítás

A mérést Android alapú telefonok segítségével végezzük. A telefon feladata a jelerősség mérése. A méréshez egy saját alkalmazás készült, amely a jelerősség adatokat egy beépített webkiszolgálón teszi elérhetővé. Így azok akármikor lekérdezhetők a mobil telefon IP címén.

1. ábra Mérési elrendezés

A mérési elrendezést az 1. ábra mutatja. A telefonok és a laptopok ugyanarra az AP-re csatlakoznak, így közvetlen kapcsolatot biztosítanak egymás felé. Az applikáció megjeleníti az eszköz IP címét (lásd: 2. ábra).



2. ábra Az Androidos alkalmazás

Az IP címen elérhető az aktuális fingerprint korlátozva a teremben található 4 AP-ra. Az alkalmazás a kalibrációt is lehetővé teszi, a kalibrációt a helyszínen fogjuk elvégezni és a kalibrációs adatokat mindenki számára elérhetővé tesszük.

3 Mérési feladatok

A mérés során előbb megvizsgáljuk a mobil készülék által szolgáltatott RSSI értéket illetve annak változását különböző paraméterek szerint (egy helyben, távolság függvényében, akadály függvényében, multipath esetén).

Ezután a példaprogram alapján megvalósítunk egy egyszerű helymeghatározási eljárást.

A méréshez készült Android-on futtatható APK (és forrás kódja is), valamint a méréshez segítségként nyújtott Matlab/Octave scriptek elérhetők a honlapon (http://www.tmit.bme.hu/vitmma07-2015), de az appendix-ben is megtalálhatók.

3.1 RSSI mérés

A mobil készülék az RSSI adatokat kiolvassa, és elérhetővé teszi egy HTTP interfészen. Azaz az adatok a laptopról lekérhetők, és a kiolvasott adatokból statisztikákat gyárthatunk. A kiolvasást közvetlenül MATLAB vagy OCTAVE-ból, a következő paranccsal tehetjük meg:

```
dataString = urlread('http://<mobil_ip_cime>:8080', 'Timeout', 3);
data = textscan(dataString, '%n %n %n %n');
fingerprint=[data{1} data{2} data{3} data{4}];
```

A visszaadott adatok a 4 AP irányába mért RSSI értékeket tükrözik. A parancs ismétlésével lehet gyűjteni az adatokat.

Feladat 1: Tegyük le a telefont egy adott helyre, és vizsgáljuk meg, hogy időben hogy változik a jelerősség!

Feladat 2: Egyenletesen távolodva az egyik AP-tól vizsgáljuk meg, hogy hogyan változik a jelerősség.

Feladat 3: A telefon útjába állva, illetve nyilvánvalóan több utas terjedés esetében hogyan változik a jelerősség? (Fém árnyékolót biztosítunk.)

3.2 Egyszerű helymeghatározási motor megvalósítása

A feladat a teremben elhelyezett 4 AP irányába mért jelerősség alapján egy fingerprint illesztés alapú helymeghatározási algoritmus megvalósítása. Az algoritmus lehet egyszerű KNN vagy Bayes vagy legkisebb négyzet alapú.

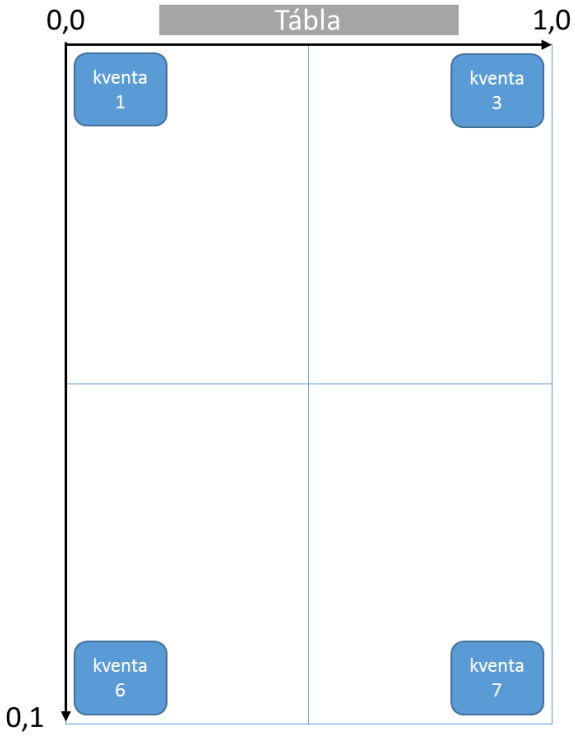
A megvalósítást Matlab/Octave környezetben támogatjuk előre megírt mintaprogram segítségével. Elsősorban kalibrációs adatokat készítünk, így a 4 AP mért jelerősségei (interpolálva) elérhetőek. A kalibrációs adatokat az init_pos.m függvény meghívásával inicializáljuk. Ennek eredményeképpen létrejönnek a következő változók:

```
NR_AP = 4;              AP-k száma
grid_size = 100;        A helymeghatározás pontosságát meghatározó rács
vq1, vq2, vq3, vq4      grid_sizexgrid_size mátrixok, melyek az 1,2,3,4
AP-k jelerősségeit mutatják adott pontokban.
```

A helymeghatározási algoritmushoz egy program vázat adunk segítségként, mely alapul szolgál az algoritmus megvalósítására.

A vq1-4 kalibráció és az fp fingerprint alapján a pozíciót kiszámolva megjeleníthető a térképen. Az adatok gyűjtése során a 3. ábra szerinti elrendezést használtunk.

A 4 AP a terem sarkaiban található. A kalibráció során a koordinátarendszer úgy volt felvéve, hogy a bal első sarokban helyezkedik el a (0,0) pont, az 1-es AP mellett. A koordináták arányszámot jelentenek, azaz 0.5 érték a terem felét jelképezi. Téglalap alakú terem esetén a kalibráció felbontása négyzetek helyett téglalapokat használ, utólagos feldolgozásnál ezt figyelembe kell venni!



3. ábra Koordinátarendszer és az AP-k helye a helyiségben

Tipp: egy metrikát használva megkereshetjük a legjobban illeszkedő mintát.

Feladat 4: Helymeghatározási algoritmus implementálása és kipróbálása.

Egy jó összefoglaló cikk az RSSI alapú helymeghatározásról itt található: http://math.tut.fi/posgroup/honkavirta_et_al_wpnc09a.pdf

4 Jegyzőkönyv

A mérés végén jegyzőkönyvet várunk, amely tartalmazza a mérések eredményét, a megvalósított algoritmust és annak pontosságát.

5 Összefoglaló

Az RSSI alapú helymeghatározás kihívásait és hozzávetőleges pontosságát mérjük fel ezen a mérésen.

Látható, hogy viszonylag egyszerűen lehet RSSI alapú helymeghatározást végezni, viszont mindenképp szükség van különböző technikákra a mérés pontosságának javítása érdekében. Konkrétan több mérés átlagolásával, a kiugró értékek szűrésével és a mozgási modell hozzáadásával javíthatjuk a helymeghatározás pontosságát.

6 Appendix 1. Inicializálás

```
% Az init_pos.m fájl

clear all; close all;

NR_AP = 4;
grid_size = 100;

calib % a calibrációs adatokat helyben osztjuk meg

[xq,yq] = meshgrid(0:.01:1, 0:.01:1);
vq1 = griddata(A(:,1), A(:,2), A(:,3), xq, yq,'v4');
vq2 = griddata(A(:,1), A(:,2), A(:,4), xq, yq,'v4');
vq3 = griddata(A(:,1), A(:,2), A(:,5), xq, yq,'v4');
vq4 = griddata(A(:,1), A(:,2), A(:,6), xq, yq,'v4');

[x,y] = meshgrid(1:grid_size, 1:grid_size);

rssi_corr=500000000*[90:-1:1].^3.731;
rssi_corr=ones(1,90);
```

7 Appendix 2. Pozícionáló algoritmus vázlat

```
% A calc_pos.m fájl

dataString = urlread('http://<mobil_ip_cime>:8080', 'Timeout', 3);
data = textscan(dataString, '%n %n %n %n');

fp=[data{1} data{2} data{3} data{4}];      % az aktuális fingerprint

for k=1:NR_AP                              % sanity check
if(fp(k)==0) fp(k)=-90; end;
if(fp(k)<-90) fp(k)=-90; end;
end;                                    % lehet feldolgozni

% ide jön az algoritmus
…
% eredmény: res(x,y)
% Megjelenítés
scatter(res(1,1), res(1,2), 'r');
axis([0 100 0 100]);
camroll(180);
```