Járműkövető rendszer nyílt forráskódú alapon

TDK dolgozat
BME építőmérnöki Kar
Geodéziai és Térinformatikai Szekció
2016/17 I. félév

Huszka Csaba Zsolt
X9PH8I
Dr. Rózsa Szabolcs

Tartalom

Bevezetés

A dolgozatban egy nyílt forráskódra épülő flottakövető rendszer felépítéséről lesz szó. A kifejlesztett rendszer nem csak a jelenleg piacon lévő szolgáltatások megvalósítására törekszik szabad szoftverek felhasználásával, hanem egyúttal a későbbiekben azok tovább fejlesztését is célul tűzi ki, a rendszer pontosabbá és költséghatékonyabbá válása érdekében. Kicsit marketingstratégiai szemmel is nézve a problémát, további célkitűzés az is, hogy a terméket szélesebb körben is fel lehessen használni a működő rendszerekhez képest, ezért további olyan újítások is kerülnek bele, amelyeknek nem feltétlen műszaki okai vannak.

Mára már itthon is egyre több cég foglalkozik flottakövetessel. Főként szállítmányozó cégek vásárolják meg a szolgáltatást, de a magán szektorban is van igény a járműveink nyomon követesére biztonsági vagy akár gyermekeink vezetési szokásainak megfigyelése céljából. Az alap minden flottakövető rendszernél ugyanaz, adott egy GPS eszköz, amit a gépjárművekre szerelnek, majd összegyűjtik és tárolják a helymeghatározás eredményeit és ezzel párhuzamosan megjelenítik azt általában egy webes felületen. Bizonyos vállalkozásoknál akár üzemanyag szintmérés és útdíj adminisztrálás is része a szolgáltatásnak, de ez nem feltétlenül része a flottakövető rendszerek szolgáltatásainak. Az országban nagyobb flottamenedzsment cégek, (például a WebEye Magyarország Kft.) további információt is közvetítenek egyszerre a vevővel felszerelt járművekről, mint például a tehergépkocsi billentési állapota vagy a helyes működéshez szükséges üzemi paraméter csomag. Az egyértelműen látszik, hogy gépjárművekkel, azon belül is inkább tehergépkocsikkal dolgozó cégek köré van kiépítve a kínálati oldal, aminek az oka egyértelmű: flottamenedzselésre nekik van legjobban szükségük és ezek a cégek a költségeket képesek fizetni, hiszen jelentős megtakarítás érhető el a gépjárművek követésével és útvonalaik optimalizálásával.

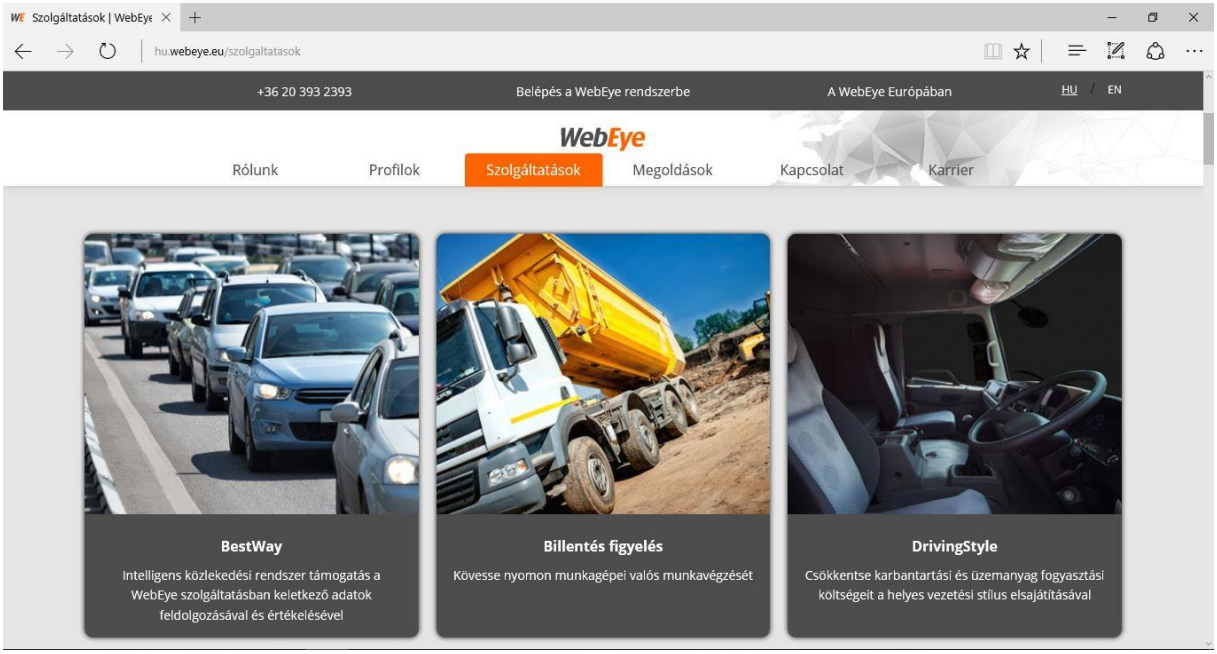
A dolgozatba bemutatott saját fejlesztésű megoldással nem csak szállítmányozási cégek piaci szükségleteit kívánom kielégíteni, hanem más területen mozgó, akár üzleti szférán kívül álló szereplőkét is. Elsősorban gondolok a magánszférán belül a kerékpár tulajdonosokra, azokra, akik napi szinten használják a biciklijüket és nem tudják azokat biztonságos helyen tartani. Ez egy nyilvánvalóan kevésbé fizetőképes csoport, mint az üzleti szférában lévő szállítmányozó cégek, viszont nyílt forráskódú alap, illetve későbbiekben részletezett fejlesztések ezt az akadályt segít leküzdeni. Másodrészt mezőgazdászokra, akiknek a mezőgazdasági gépek helymeghatározása is tud fontos lenni. Számukra fontosabb a gépvezérlés és az önvezetés, mint jogtalan eltulajdonítás elleni biztonság. Ehhez szubméteres vagy néhány centiméter pontos pozíció szükséges, amit a kódmérésen alapuló GPS helymeghatározási technika nem tud teljesíteni, így a piacon lévő jelenlegi flottakövető rendszerek nem tudnak/akarnak erre piacra belépni.

1. Flottakövető szolgáltatások Magyarországon

Magyarországon megnőtt a flottakövetéssel foglalkozó cégek száma. Az elmúlt két évtizedben a logisztika és szállítmányozás megerősödött, több versenytárs jelent meg a fuvarozói piacon, így egyre nagyobb szükség van a hatékonyabb munkavégzésre, a források teljeskörű kiaknázá-

sára. Közben új adó is bevezetésre került, ami közvetlen a tehergépjármű tulajdonosokat célozza meg. Az út rangjától függően különböző tarifák mellett lehet használni az adott szakaszt. Ennek a későbbi adminisztrálását GPS-el lehet a legegyszerűbben megtenni. Ennek megkerülése, pénzbírságot von maga után.

Az egyik legnagyobb flottakövető cég Magyarországon a Webeye. Még 11 országban jelen van Európában. Fő profiljuk a teherautók GPS alapú nyomkövetése és ezen adatok elemzése. Elsősorban útvonaltervezést és útjíd adminisztrálást végeznek az alapcsomagjukban. További feldolgozott információkból meghatározzák az optimális vezetői stílust, hogy a jövőben pénzt és időt spóroljanak a gépjárművezetők. További érzékelők beszerelésével, üzemanyagszintmérés, és dőlésmérés is egészítheti ki a szolgáltatást. Igényelni lehet ügyfélszolgálatot is a rendszerhez, a sofőröket pedig operátorok segítik munkájuk során. Természetesen a megrendelő cég saját maga is elvégezhetik a diszpécser szolgáltatásokat, ezért saját autóinak az információihoz a megrendelők böngészőn férhetnek hozzá. A szállítmányozó cég „fontos helyeket" tud megjelölni a WebEye felületén, vagy akár tiltott zónákat, ahova, ha az autó belép, riasztást ad le, hogy esetleges visszaéléseket meg lehessen előzni.



1. ÁBRA WEBEYE SZOLGÁLTATÁSOK

A többi vállalat, mint például a Multialarm, Myfleet vagy az iTrack is hasonló szolgáltatást nyújtanak, de az üzemanyag szintmérés, az eTachográf, a billentés figyelés vagy az ügyfélszolgálat nem mindenhol érhető el. Magyarországon a WebEye rendelkezik a legnagyobb funkció palettával. Minden flottakövető cégnél közös, hogy GPS-el felszerelt teherautók helymeghatározása, a pozíciók elemzése a cél, hogy profitnövekedést érjenek el a szállítmányozó vállalkozások.

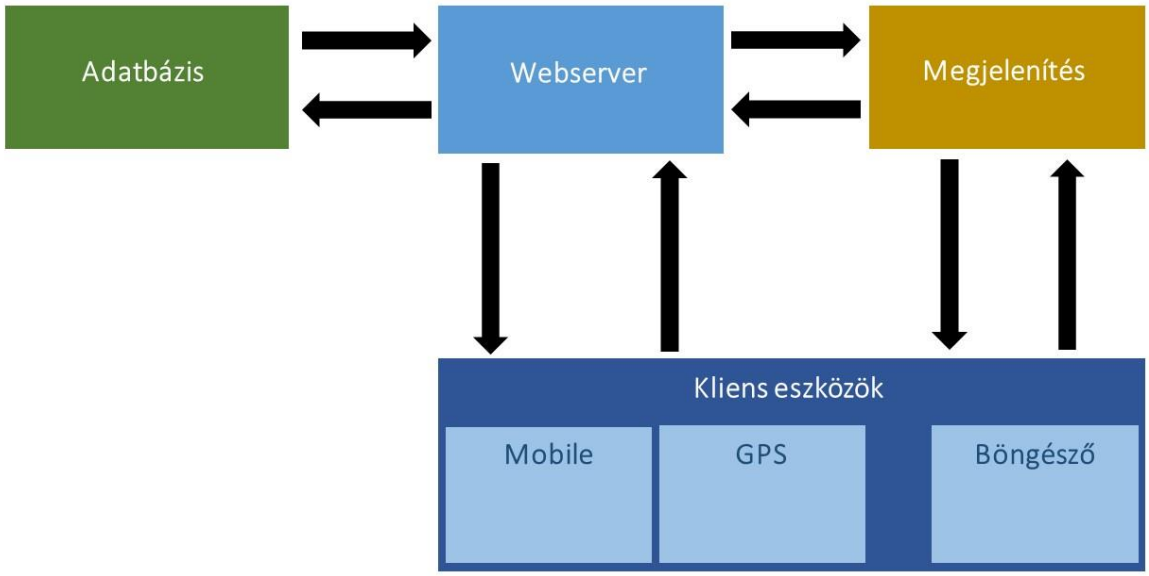
2. Rendszer általános felépítése

Az általam felépített szolgáltatást a felhasználó szemszögéből fogom bemutatni annak érdekébe, hogy a rendszereket nem ismerő olvasó számára is érthetőbbek legyenek az ok-okozati összefüggések.

Ahogy azt láthattuk a flottakövető rendszereknél a felhasználó kétféle módon tud kapcsolatba lépni a szolgáltatással. Az egyik - és egyben legfontosabb – a GPS kliens. Itt egy olyan eszközre kell gondolni, ami pozíciót tud rögzíteni és valamilyen internet kapcsolattal is rendelkezik. A helymeghatározási adatok, gyakorlatilag közvetlenül az eszközről kerülnek továbbításra a szerverre, de az eszköz a mobil internet kapcsolat kiesése esetén az adatokat saját maga is tudja tárolni. Másik kapcsolódási lehetőség a böngésző, ahol a felhasználó meg tudja nézni a rögzített pontokat egy webes térképi megjelenítő segítségével. Egyetlen egy feltétel van, hogy a felhasználónak olyan böngészőt kell használnia, ami a JavaScript nyelvet támogatja. Ez manapság abszolút nem nagy elvárás, viszont ez azt is mutatja, hogy akár okostelefonról is használható a felület.

A felhasználó eszközei által küldött adat, minden esetben a webszerverre futnak be. Ennek nevesítetten két féle módja van, de erről részletesen a későbbiekben lesz szó. Ez a webszerver teremti meg a kapcsolatot az adatbázis és a kliensek eszközök között is, továbbá gondoskodik a megjelenítésről és vezérlésről, hogy ne kelljen az felhasználónak kódokat írnia az adatok manipulálásához.

Az adatbázis tárolja a feltöltött adatokat és rendszerezi őket táblákba. Fenntart egy jogosultsági rended, ami pontosan leírja, hogy milyen felhasználónak pontosan milyen jellegű módosításhoz van joga. A gazda számítógép háttértárát használja föl az információk mentésére. Technikailag lehetne megoldás, hogy közvetlenül a felhasználó is tudjon kapcsolódni az adatbázishoz, de az nem lenne túl biztonságos. Szándékosan úgy alakítottam ki a rendszert,, hogy az adatbázis távolról ne legyen elérhető, így biztosítva van, hogy ténylegesen csak a webszerver kommunikáljon az adatbázissal.

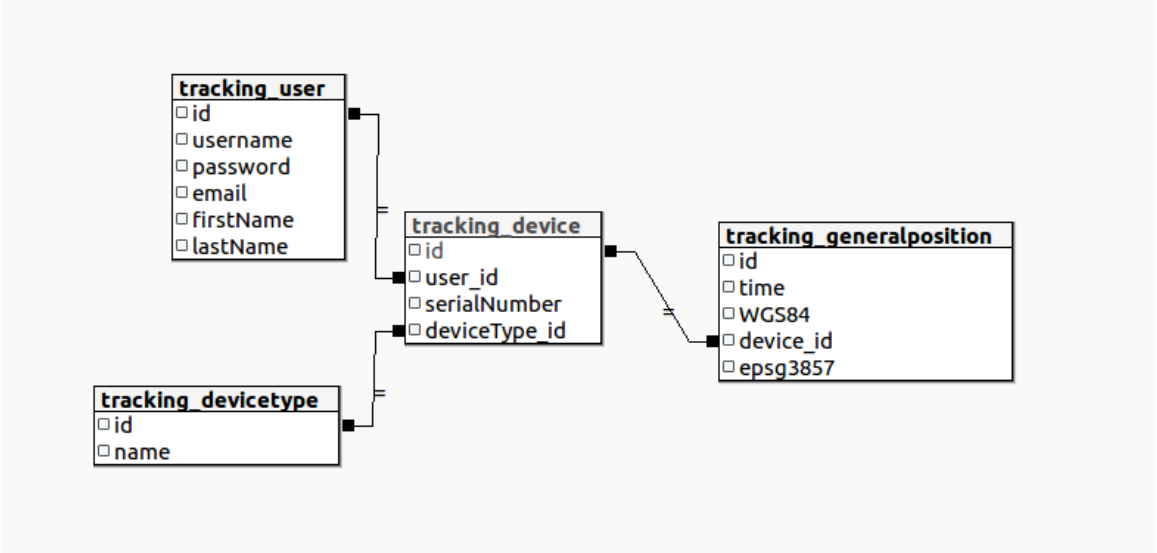


2. ÁBRA SZOLGÁLTATÁS FELÉPÍTÉSE

3. Rendszerfejlesztés

Ezúttal fejlesztői szemszögből mutatom be, a felépült szolgáltatást, hogy érthetőbb legyen a szerkezet. Ahogyan arra korábban is utaltam, bizonyos elemeket több részre szedek szét, mert különböző módon látják el a funkciókat. Ilyen például a kliens oldalon a böngésző, az okostelefon vagy a GPS.

3.1. Adatbázis



3. ÁBRA ADATBÁZIS DIAGRAMM

PostgreSQL nyílt forráskódú relációs adatbáziskezelő rendszert választottam a szolgáltatás adatainak mentéséhez. Ennek egyik oka, hogy bőven meghaladja a rendszer funkcionális igényeit, tehát bonyolultabb adatmanipulációra is képes, mint amire egyelőre szükség van. Másik fontos, oka, hogy könnyen bővíthető, például a PostGIS - el, ami egy olyan kiegészítő csomagja a PostgreSQL-nek másnéven Postgres-nek, ami téradatok kezeléséhez biztosít könnyen kezelhető felületet. A webszerver és a PostGIS helyes működéséhez szükségesek további táblák létezése, de ezek nem kapcsolódnak szorosan az általam kialakított szolgáltatás relációihoz, ezért azokat nem részletezem.

Ahogyan azt a 3. ábra mutatja, négy tábla működik a rendszer mögött. A „tracking_user" nevű tábla felel a felhasználók tárolásáért. Hat mezővel rendelkezik, az első az „id", egész számokat lehet értékül adni. Ez a mező elsődleges kulcsként működik, ami azt jelenti, hogy minden egyes értéke egyedi a táblán belül, nem lehet két külön rekordnak ugyan az az id-je. A „username", „password", „email", „firstname" és „lastname" szöveg típusú mezők, ahova száz karaktereket tudunk tárolni soron ként. Egyetlen rekord bevitele egy felhasználó regisztrálásával egyenlő.

A „tracking_devicetype" egy kis tábla, ahova a mindenkori üzemeltető tölti föl a szolgáltatás által támogatott GPS eszközöket. Szintén van egy „id" mező, ami idegenkulcs ként működik és egész számokat képes tárolni, továbbá az eszköz típusának a neve, például „adafruit Ultimate GPS".

A „tracking_device" tábla a regisztrált eszközöket tartalmazza. Négy attribútuma van, az első a már megszokott „id", ami itt is ugyanazokkal a tulajdonságokkal bír, mint korábban említett táblákban. Újdonság viszont, hogy a „tracking_device.user_id" és a „tracking_device.deviceType_id" önmagukban egész szám típusú mezők, viszont hivatkoznak más táblák rekordjaira, azon belül is az elsődleges kulcsukra, tehát az „id" attribútumokra. Ezeket az attribútumokat idegen kulcsoknak nevezzük. A „tracking_user.id", „tracking_device.user_id" között és a „tracking_devicetype.id", „tracking_device.deviceType_id" között egyaránt '1 – N' -es kapcsolat áll fönn, azaz egy eszköztípushoz és egy felhasználóhoz több eszköz is társulhat. A leggyakrabban használt tábla a „tracking_generalposition", mert minden pozíció ebbe a táblába kerül mentésre. Szintén van egy „id" nevű elsődleges kulcsa, rendelkezik egy „device_id" idegen kulcs attribútummal, ami egész szám típusú és a „tracking_device" relációra hivatkozik. '1-N' -es viszony van jelen a két tábla között, mivel egy eszköznek rengeteg pozíciója kerül mentésre. A „time" attribútum, egy Date dátum formátumot kezelni tudó típus, míg a „WGS84" és az „epsg3857" PostGIS geometria fajta mezők. Sajnos ugyanaz a pont, két különböző vetületi rendszerben is tárolásra kerül, mert a pontok transzformálása a webtérkép meghívása előtt túl kapacitás igényes lenne, így a transzformálás közvetlenül a mentés előtt történik.

3.2. Webszerver

Az előző fejezetben, mindent ismertettem az adatbázisról, ami szükséges ahhoz, hogy lássuk, hogy milyen adatok tárolódnak és milyen információk szükségesek számunkra. Arról még nem volt szó, hogy ezek az adatok hogyan áramlanak az alkalmazáson belül.

A PostgreSQL önmagában is az SQL[1] nyelv segítségével programozható, ami járható út is lehet bizonyos esetekben, viszont nagyon sok hacker támadásra önállóan kell felkészülni, ami az effektív munkából túl sokat vesz el. Éppen ezért én a Django web Framework -öt használom, mert ezek a védelmi mechanizmusok ebbe alapértelmezetten bele vannak építve. Ez a Python[2] programnyelv egy kiegészítő csomagja, ami szintén nyílt forráskódú és ingyenesen elérhető, használható. További előnye, hogy nem kell SQL nyelvet használnom, a webszerver működtetése, módosítása Python környezetben is elvégezhető. A Django az alkalmazások létrehozásához a Model – View – Template *(Modell – Nézet – Sablon)* mintát követi. Ezt a mintát én is követtem, ennek segítségével szerveztem az alkalmazást.

3.2.1. Modell

Modellen belül fogalmazom meg, az adatbázis struktúrát, pontosan azt, amit az előző fejezetben tárgyaltam. Tehát minden reláció, elsőként Python osztályként születik meg. Az osztály neve a tábla neve lesz és a mezői a reláció attribútumai lesznek. Nagy előnye van ennek, mert egy példány objektum egy sort fog reprezentálni. Ez rengeteg időt spórol meg a programozónak, mert nem kell fölöslegesen típuskonverziókkal foglalkozni. Később a Nézet*(View)* erre támaszkodva fog kapcsolatot tartani az adatbázissal.

3.2.2. Nézet

Ebben a szakaszban történik az effektív munkavégzés. Mint például adatok mentése, megjelenítése és azok esetleges manipulálása. Ezeket a cselekményeket megvalósító függvényeket minden esetben egy felhasználói kérés indítja meg. Amennyiben a felhasználó kliense segítségével kérést fogalmaz meg, a webszerver ebben a környezetben dönti el, hogy hogyan válaszoljon rá. A felhasználó kérésnek kétféle útját építettem ki, amelyek mindkét esetben http[3] kapcsolatra épülnek.

Az egyik a POST kapcsolat, ami a kérés-csomag részeként tölti fel a kliens által megfogalmazott kérés esetleges értékeit. Ide tartozik a regisztráció, amikor adatbázisba mentés történik. Amikor a felhasználó böngészőjén keresztül bejelentkezést próbál végrehajtani, az adatbázis ilyenkor adatot olvas ki a táblákból, tehát kikeresi, hogy létezik-e ilyen felhasználónévvel és jelszóval regisztrált személy és az eredmény függvény értelmében közvetít tovább információt. Amennyiben csak tájékozódik az oldalon az adott személy, szintén a böngészőjén keresztül, akkor egy irányú adatáramlás van, tehát csak letöltés zajlik az oldal nem vár semmilyen információt.

A másik pedig a GET[4] kapcsolat, ami viszont a kéréssel együtt megfogalmazott információkat önmagában az URL[5]-ben tárolja. Ide kizárólag a helymeghatározási adatok feltöltése tartozik. Saját fejlesztésű eszköz készítésekor ezt az utat választottam, hogy a pozíciók mentése, akár kézi bevitellel is könnyen végrehajtható legyen. Teszt fázisban könnyíti meg nagyon a munkát.

---

[1] Relációs adatbázis-kezelők strukturált lekérdezési nyelve

[2] Általános célú, nagyon magas szintű programozási nyelv

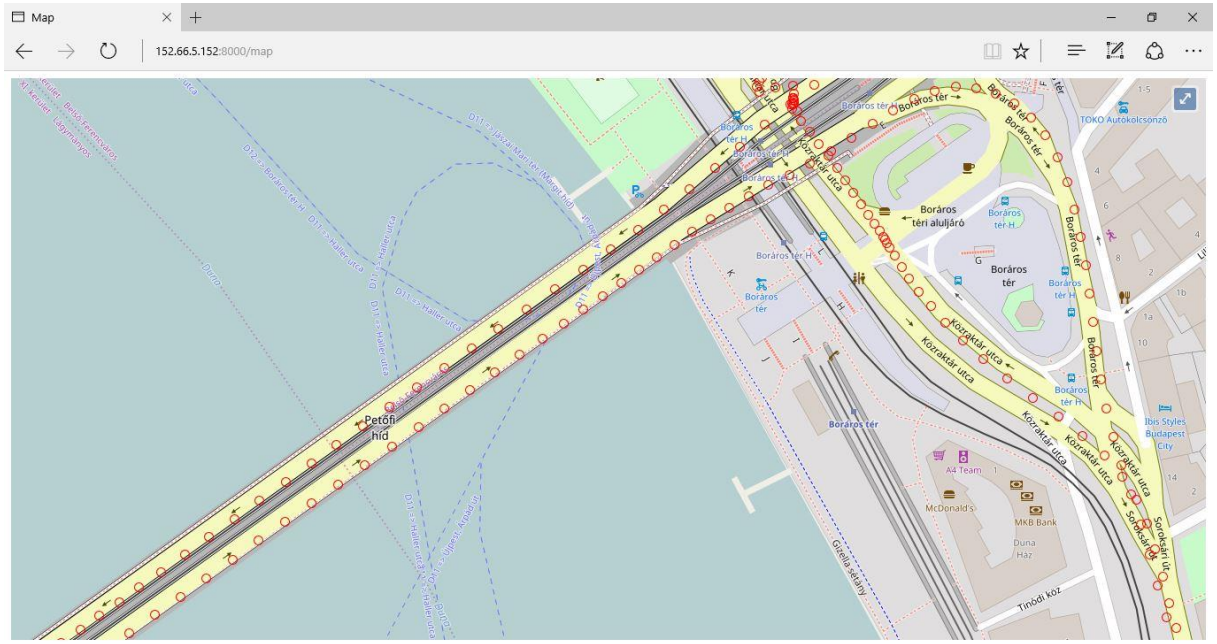
[3] (HyperText Transfer Protocol) Információátviteli protokoll elosztott információs rendszerekhez.

[4] Példa: http://www.pelda.hu/?peldaparameter=peldaertek. A példa weboldal a példaparaméternek példa értéket fog kapni.

[5] Az interneten megtalálható erőforrások szabványosított címe.

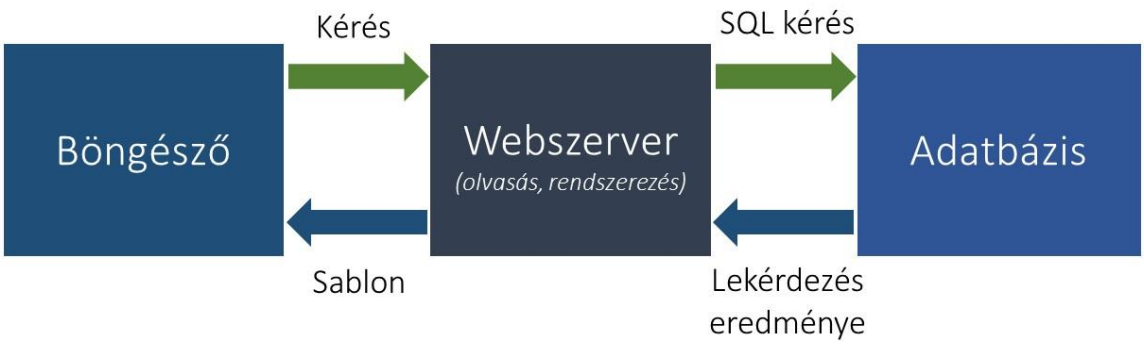
3.2.3. Sablon

Kizárólag böngésző meghívásakor van igazán értelme használni. Lényege, hogy egy statikus HTML[6] fájlt töltünk fel adattal. A sablon jelen esetben egy nagyjából megtervezett és formázott HTML oldal, ez a fájl az oldalon mindenhol ugyanúgy néz ki, például a háttér mindenhol kék színű. Viszont a bejelentkezésnél más mezők, feliratok, lehetőségek jelenhetnek meg, mint regisztrációnál. További nagy előnye, hogy a HTML fájlokat tovább tudjuk bővíteni JavaScript nyelvvel, jelen esetben a további formázási lehetőségek mellett webtérképet is biztosít.



4. ÁBRA SZOLGÁLTATÁS WEBTÉRKÉPE *(PIROS KÖRÖK A MÁR MENTETT POZÍCIÓKAT JELÖLI)*

[6] Egy leíró nyelv, melyet weboldalak készítéséhez fejlesztettek ki és mára internetes szabvánnyá vált.



5. ÁBRA A *WEBOLDAL MEGNYITÁSÁNAK FOLYAMATA*

Sorba téve az állomásokat a következő folyamat játszódik le az alkalmazásban GPS -ünk pozíciójának lekérésekor. A felhasználó kérést fogalmaz meg böngészőjén keresztül, tehát megnyitja a webtérképet az oldalon, erre a View állomáson kiolvasás történik az adatbázisból. A felhasználó eszközéhez tartozó pozíció geometriai adatok vissza áramlanak a View-hoz, ahol ezúttal már adatrendszerezés és megjelenítéshez való előkészülés történik. Utolsó lépésként a View feltölti a sablont geometriai információkkal és elküldi a böngészőnek, tényleges vizualizáció a felhasználó böngészőjében történik meg, plusz információt nem fog hozzáadni a böngésző az oldalhoz.

3.3. Kliensek

3.3.1. Böngésző

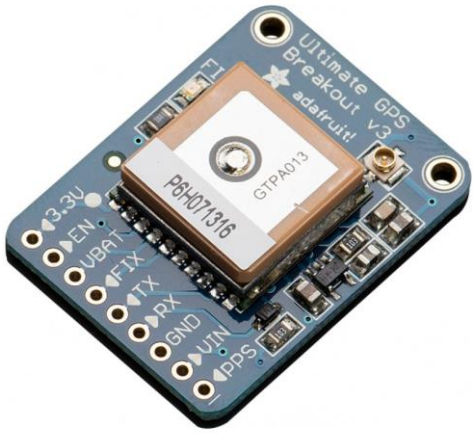
Felhasználó a böngészőn keresztül tudja legegyszerűbben elérni a szolgáltatást. Manapság már nem is biztos, hogy ez számítógépen történik. Az oldal elérhető okostelefonon, Tabletről vagy bármilyen olyan eszközről, amely weblapokat meg tud jeleníteni és rendelkezik valamilyen internet kapcsolattal. A webszervert ugyanúgy fel lehet készíteni, hogy olyan sablont küldjön az eszköznek, amely arányosan méretezett a képernyő nagysághoz.

3.3.2. Adafruit GPS helymeghatározó kliens

A helymeghatározási és adatgyűjtési feladatokat végrehajtó kliens kialakításához a Raspberry PI *(6. ábra)* miniszámítógépet választottam. A kis számítógépre lehet Linux operációs rendszert telepíteni, ami tartalmaz Python fordítót.

6. ÁBRA RASPBERRY PI

Ez még önmagában helymeghatározáshoz nem elegendő, ehhez csatlakoztattam az Adafruit által gyártott kód mérésre alkalmas GPS modult *(7. ábra)*. A GPS a Raspberry USB portjára küldi a pozíciót, NMEA[7] üzenetek formájában.



7. ÁBRA ADAFRUIT ULTIMATE GPS

NMEA üzeneteken belül én a GGA *(Global Positioning System Fix Data)* adatokat dolgozom fel. Ez az egyik legfontosabb NMEA üzenet, amely tartalmazza a helymeghatározás megoldását ellipszoidi földrajzi szélesség, hosszúság és ellipszoid feletti magasság formájában.

($GPGGA,001430.003,3907.3885,N,12102.4767,W,1,05,02.1,00545.6,M,-26.0,M,,*5F)

NMEA minta üzenet

[7] National Marine Electronics Association által kidolgozott szabvány, GPS adatokra vonatkozóan.

Minden üzenet egy dollárjellel kezdődik, majd jelölésre kerül, hogy milyen típusú NMEA üzenettel állunk szemben. A vesszös elválasztást követjük, akkor a harmadik és az ötödik tag fontos *(mintában 3907.3885 és a 12102.4767)* ezek jelölik rendre a szélességi és hosszúsági adatokat. Ellenőrzésként lehet használni a nyolcas tagot, ami a látható műholdak számát reprezentálja *(mintában 05 értékkel szerepel)*. Feldolgozás közben lehetne vizsgálni, hogy a műholdak száma több-e mint négy, ami biztosítaná a kiegyenlítés megtörténését, viszont a gyakorlat azt, hozta ki, hogy ha nem helyes az üzenet, ezáltal nem feldolgozható, akkor a pozíció sem pontos és műholdak száma sem elégséges. Ez fordítva is igaz, tehát ha a pozíció pontos, akkor az üzenet is hiánytalan, feldolgozható és több műholdat látunk, mint az elégséges műholdszám.

Az üzeneteket a már említett Python fordító segítségével dolgozom föl. Készítettem egy olyan scriptet, amely annak futtatásától kezdve, folyamatosan figyeli a „ttyUSB0" -ás portot Erre a portra küldi a GPS modul soron ként az NMEA üzenetet. Amennyiben nem NMEA formátumú, tehát, hibás vagy hiányos a kód eldobja azt a sort. Helyes üzenet rögzítésekor, feltölti egy általam írt osztály példányát, amelynek egyes attribútumai az NMEA üzenet értékeit hordozzák. A létrejött objektumot egy URL formatáló függvény várja, amit az URLlib[8] segítségével alakítottam ki. Majd a korábban ismertetett GET http kapcsolattal elküldi a szervernek.

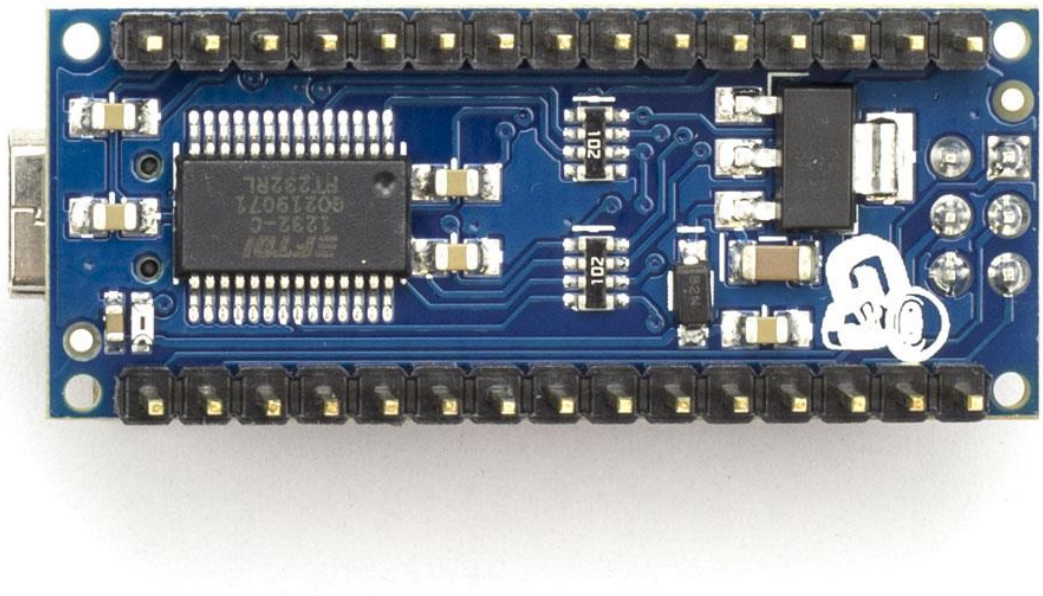
8. ÁBRA GPS ADAFRUIT POZÍCIÓ RÖGZÍTÉSI FOLYAMAT

Ugyan a rendszer kialakítása és tesztelése során a Raspberry PI ideális eszköznek bizonyult, a végleges adatgyűjtő kliens kifejlesztése során nem lesz célszerű ennek használata, a mérete és a szükségtelen funkció gazdagsága miatt. Ezért a későbbiekben az Arduino Nano mikrokontrollerre tervezek áttérni. Egy kevésbé számítógépesített eszköz, amely továbbra is programozható, gyakorlatilag közvetlenül, tehát operációs rendszer megléte nem szükséges. Ebből az eszközből

[8] Magas szintű interfészt biztosít adatok manipulálására a Web-en keresztül.

inkább célhardware -t lehet építeni, ami gazdaságosabb, mivel számunkra felesleges funkciókat nem tartalmaz.

További előnye, hogy mérete miatt, be tudjuk szerelni más járművekbe is, akár kerékpárokba is. Ezzel bővítve a szolgáltatás kínálatát, amivel egy újabb piaci csoportnak az igényeit ki lehet elégíteni, gyakorlatilag minimális plusz munkával. Ez egy olyan fejlesztés, ami nem műszakilag viszi előre a rendszert, hanem felhasználást tekintve lesz függetlenebb, mint a mostani piaci társai.



9. ÁBRA ARDUINO NANO

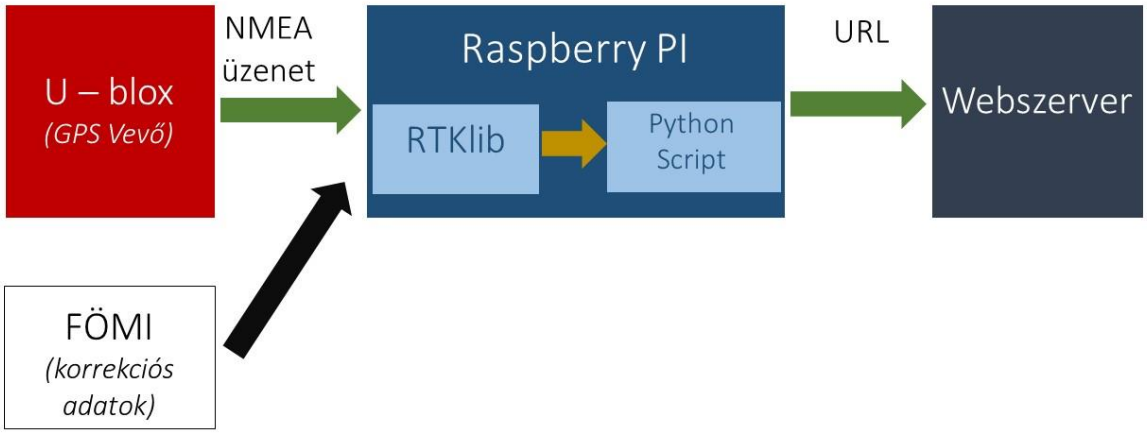
3.3.3. U – blox / Raspberry Pi adatgyűjtő kliens

A U-blox vevő egy nagy teljesítményű GPS modul. Az L1 frekvencián végzett mérések végrehajtására képes, de a kódmérés mellett a nyers fázismérésekhez is hozzáférhetünk. Így megfelelő feldolgozást követően pontosabb koordinátát tudunk előállítani, mint a kódmérésre alkalmas vevőkkel. Költséghatékony eszköz, mert egyrészt olcsón hozzá lehet jutni a piacon, másrészt az energia igénye sem magas. Számomra különösen hasznos, mivel számos csatlakozási lehetőséggel rendelkezik. USB port segítségével kapcsolódtam hozzá a Raspberry PI -vel *(6. ábra).*

Az általam kódolt script, amely az NMEA üzenet dolgozza föl és küldi tovább a webszervernek, ebben az esetben is működik a kis számítógépen. Viszont a U-blox vevő adatait elsőnek az RTKlib szoftvercsomag segítségével dolgozom föl. Az RTKlib egy olyan nyílt forráskódú program csomag, amit Linux operációs rendszeren parancssorból vagy Windowson párbeszédablakokon ke-

resztül lehet futtatni. Raspberry -n, a parancssoros módszert lehet használni. Több helymeghatározási rendszerrel tud együttműködni, mint a GPS, GLONASS vagy a GALILEO. Rengeteg formátumot támogat én továbbra is az NMEA üzenetet használom.

Tehát ezzel a módszerrel, egyfrekvenciás fázisméréssel határozom meg a földi koordinátákat. Az RTKlib segítségével, A FÖMI GNSSNet.hu aktív GNSS hálózatából kapott valós idejű korrekciós adatokkal dolgozom fel valós időben a méréseket, majd a Raspberry, az RTKlib-ben beállított portjára küldöm az adatokat. Ezután az NMEA értelmező scripttel feltöltöm az adatokat a szerverre.



10. ÁBRA U - BLOX POZÍCIÓ RÖGZÍTÉSI FOLYAMAT

Az így kialakított megoldás pontosabb helymeghatározási eredményt szolgáltat, mint a kódmérés, hiszen a fázisméréssel végzett műhold-vevő távolság meghatározás elméleti szinten néhány mm pontos. Ugyanakkor ismeretes, hogy a mérések feldolgozásához fel kell oldani a ciklustöbbértelműséget, amely némi időt vesz igénybe. A korrekciók vételét követően néhány másodpercen belül szubméteres pontosságú úgynevezett „float” megoldást kaphatunk, míg a ciklustöbbértelműségek egész számként történő feloldása több percet is igénybe vehet. Az így kapott „fixed” megoldással már akár a cm-es helymeghatározási pontosság is elérhető.

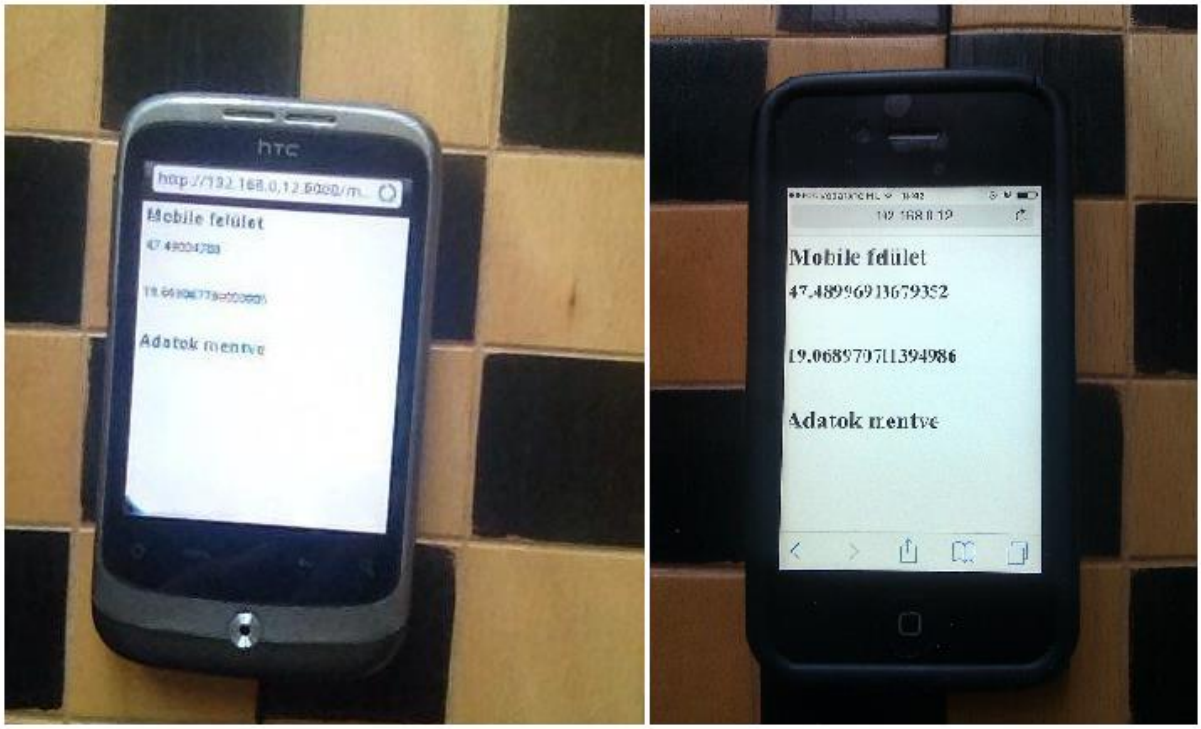
A helymeghatározási eljárás alkalmazhatóságát tesztmérésekben vizsgáltam meg, amiket a későbbiekben fogok részletesen bemutatni.

3.3.4. Mobil

Ahogyan azt már korábbi fejezetben írtam, okos készülékkel is lehet kapcsolódni a szolgáltatáshoz. Ez azért érdekes a mi szemszögünkből, mert ezek a készülékek tartalmaznak önmagukban GPS modult. Tehát ez azt jelenti, hogy ugyanúgy lehet ezeket az eszközöket helymeghatározásra használni, össze lehet kötni a szolgáltatással.

Gyakorlatban ez úgy néz ki, hogy hozzá lehet férni a készülék GPS adataihoz. Gyártótól függően, JAVA, vagy Swift nyelv segítségével lehet pozíció adatokat kérni az eszköztől. Ez egy munkás feladat, mivel kétszer kell megírni mindent, viszont későbbiekben a szolgáltatáshoz külön alkalmazások tartozhatnak, így ez nem csak flottakövető rendszer lehet, hanem akár okos eszköz figyelő is.

Egyelőre, az eszköz böngészőjén keresztül szerzem be a helymeghatározási adatokat a JavaScript nyelv „Location API[9]" -jának segítségével. Így márka függetlenül tudom használni teszteléshez a pozíciós adatokat. A 11. ábra egy ilyen próba pozíció rögzítést mutat be.

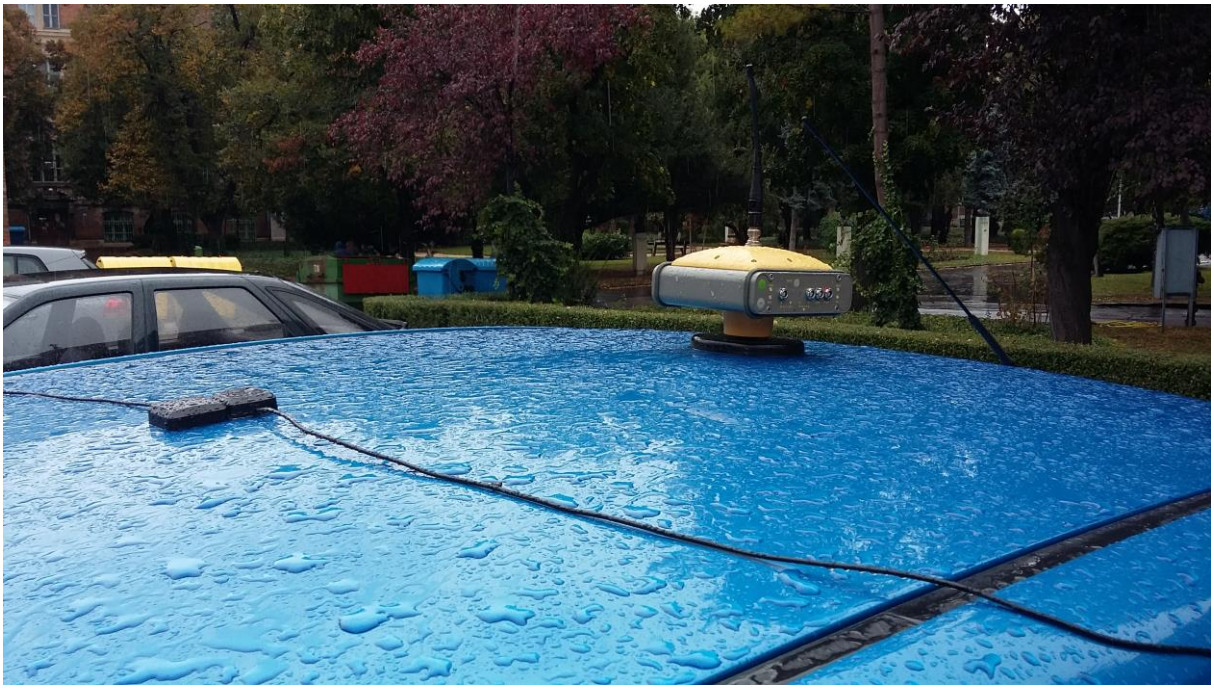


11. ÁBRA POZÍCIÓ RÖGZÍTÉSE OKOSTELEFONOKON A SZOLGÁLTATÁS FELÜLETÉN

4. Helymeghatározási technológiák

Ebben a szakaszban a mérések módjairól, felhasznált eszközökről és azok eredményeiről lesz szó. Cél a helymeghatározási technikák összehasonlítása és a szolgáltatáshoz a legmegfelelőbb mód kiválasztása. Az összehasonlítások alapját, adott útszakaszokon való tesztmérések adják. A tesztmérések időtartama az útvonalak hosszával arányos. Magát a méréseket személyautóra erősített vevők és a vevőkre csatlakoztatott antennák segítségével végeztük. Topcon Hiper Pro geodéziai pontosságú kétfrekvenciás RTK műszer, adafruit ultimate GPS és U-blox vevő vett részt a tesztmérésben. A*12. ábra* jól felismerhető a Topcon műszer antennája és további két antenna, ami az előbb felsorolt műszerekhez kapcsolódik.

[9] Egy program vagy rendszerprogram azon eljárásainak és azok használatának dokumentációja, amelyet más programok felhasználhatnak.

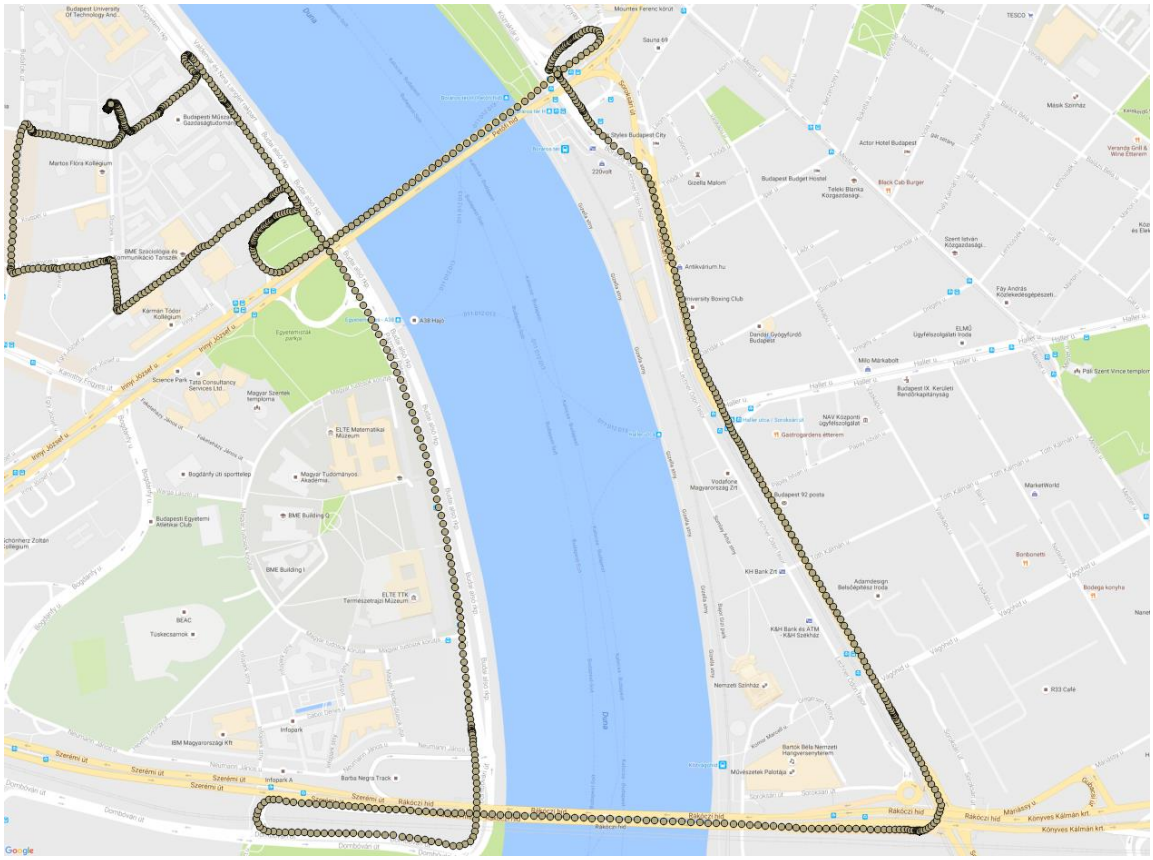


12. ÁBRA GPS ADAFRUIT ANTENNA- UBLOX VEVŐ ANTENNA- TOPCON ANTENNA

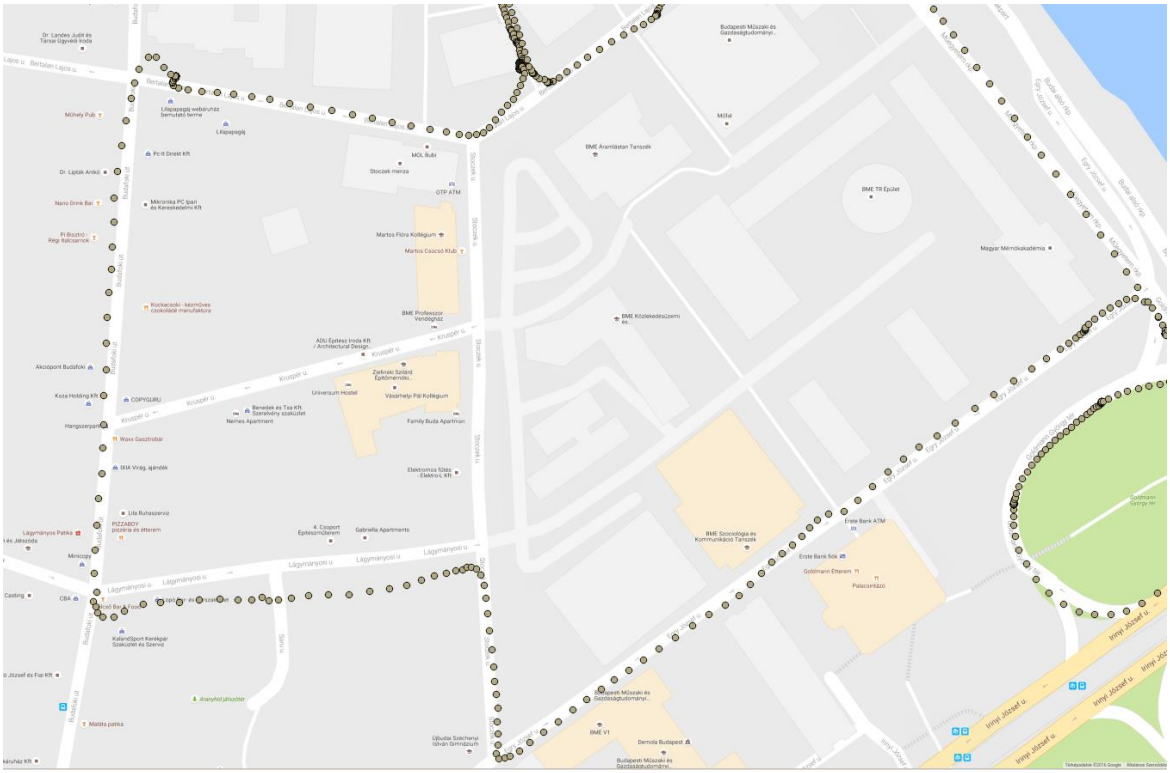
4.1. Abszolút helymeghatározás kódméréssel

A mérendő jel, egy vivőjelen lévő kódsorozat. Kód távolságokat mérünk, majd ebből számolunk földi koordinátát. Előnyei közé tartozik, hogy négy műhold szükséges a pozíció szerzéshez. Továbbá, minden egyes epochában egyértelmű megoldás van. A rendszer indítását követően tíz – húsz másodperc alatt pozíciót tud adni, de jelvesztést követően szinte azonnal helyreáll a pozíciómeghatározás a műholdak észlelését követően. Az eszköz pár ezer forintért beszerezhető. Hátránya, hogy jellemzően 5 – 7 méteres pontossággal adja meg pozíciót.

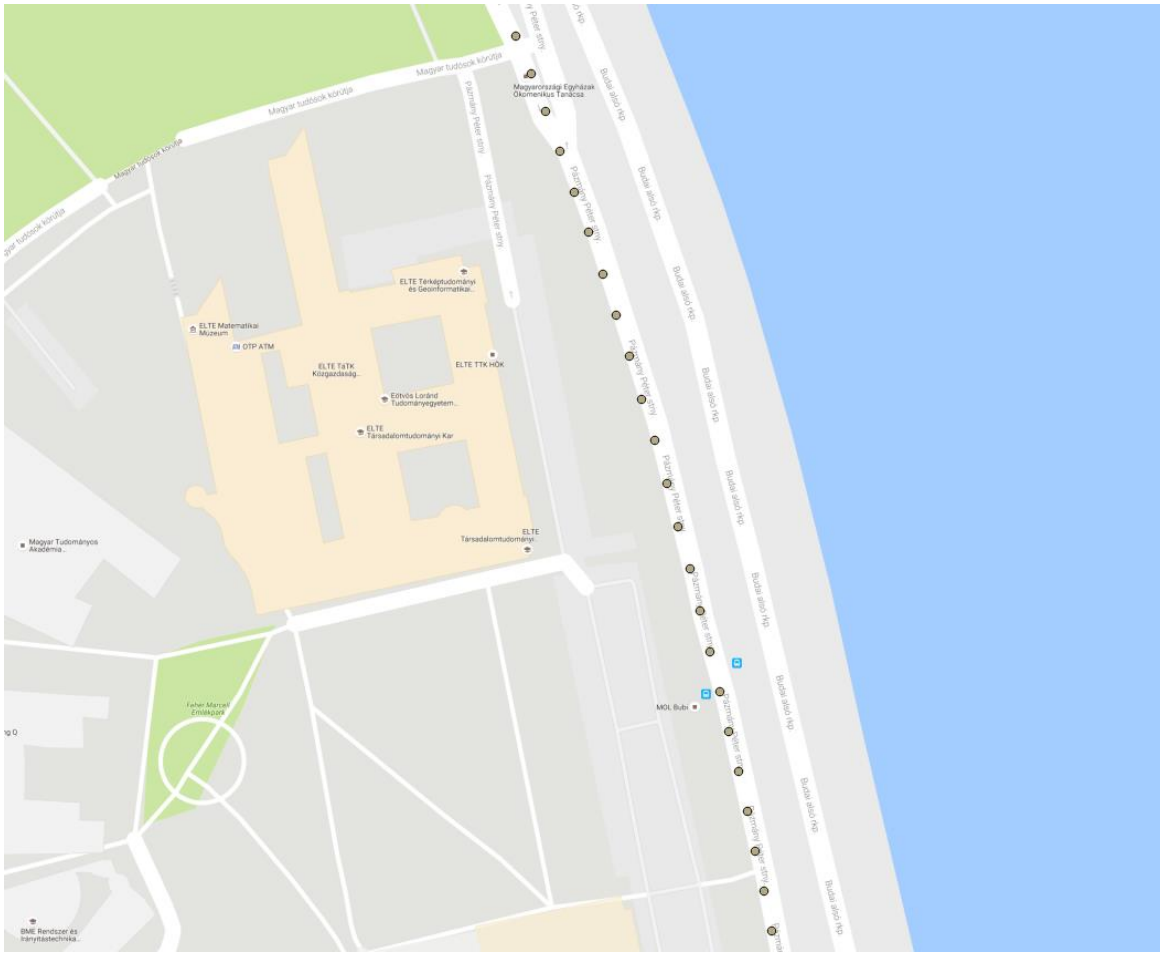
Ezt technikát a Adafruiton GPS-en keresztül használtam a mérés során. A tesztelés során előkerültek az előnyei a kódmérésnek, ahogyan azt remélni lehetett. 13. ábra mutatja, az útvonalát a tesztmérésnek, amit az Adafruit GPS modullal rögzítettem. Minden egyes másodpercben rögzítésre került egy egyértelmű pozíció, nem volt sehol sem jel vesztés. Viszont a nagyobb házak között pontatlanok a koordináták elsősorban a korlátozott műholdgeometria és a többutas terjedés miatt, ami várható is volt általánosan a helymeghatározás természetétől.



13. ÁBRA GPS ADAFRUIT TESZTMÉRÉS, TELJES SZAKASZ



14. ÁBRA GPS ADAFRUIT TESZTMÉRÉS, ÉPÜLETEK KÖZÖTT

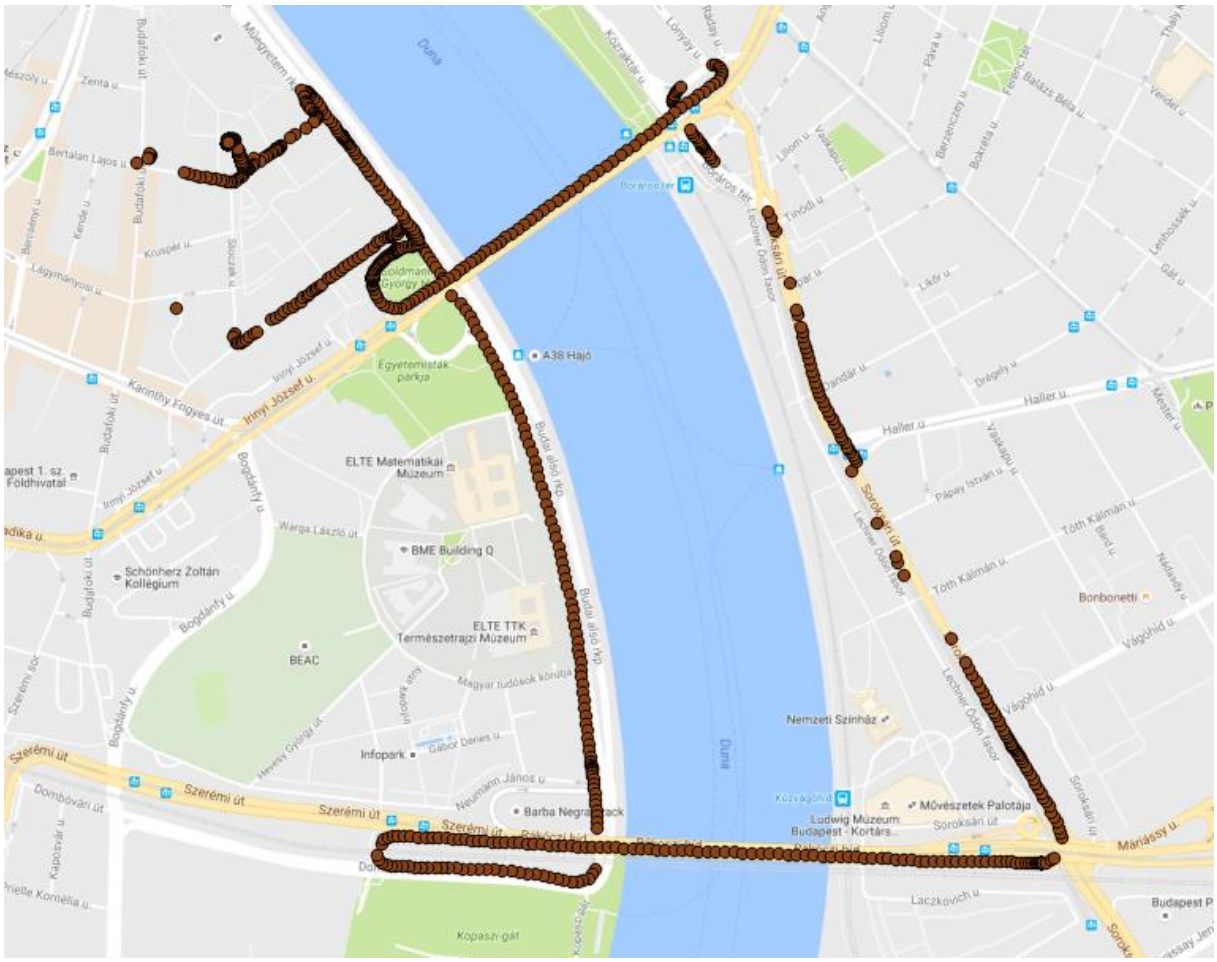


15. ÁBRA GPS ADAFRUIT TESZTMÉRÉS, NYÍLT TEREP

Érdemes megfigyelni, hogy nyílt terepen akár már méteren belüli pontosságot is elérhetünk a mérések során. Flottakövetéshez, hogy méter pontosan tudjuk, hogy az autók merre járnak ahhoz teljesen elegendő az eszköz. Kerékpár követéshez is megfelelő, sőt talán a legcélszerűbb, mert az eszköz elég pontos ahhoz, hogy biztonságban tudjuk biciklinket, viszont nem félmilliós árral rendelkezik, mint többfrekvenciás társai, hogy kerékpárosoknak ez problémát okozzon.

4.2. Relatív helymeghatározás fázismméréssel

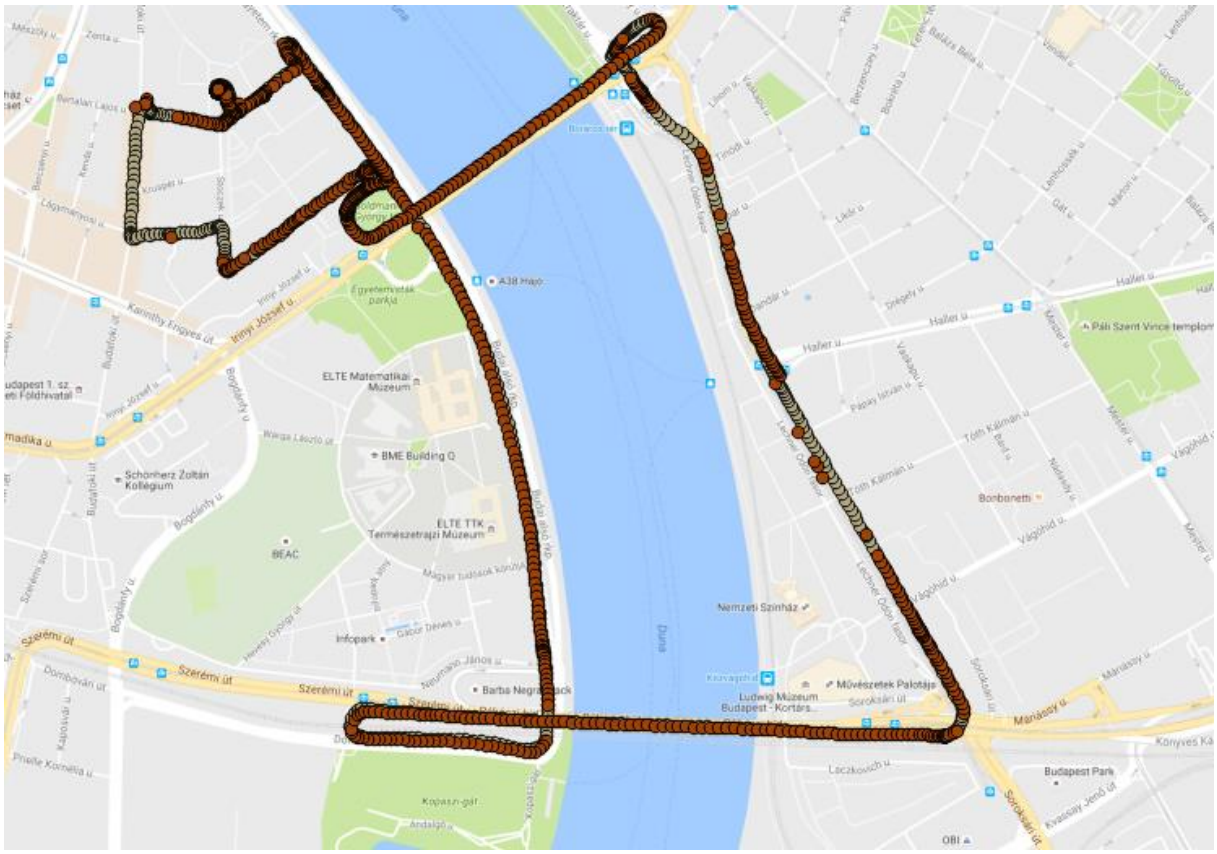
Ezt a mérési technikát a U-blox vevőn keresztül használom. Egy frekvenciával dolgozik, ciklus-többértelműség feloldásával határoz meg földi koordinátákat. Éppen ezért nem minden másodpercben van egyértelmű pozíció. Minden megoldásnál első lépésben float megoldás születik. A pozíciók pontosabbak, mint a kódmérésnél. A méteren belüli pontosságot könnyedén el lehet érni.



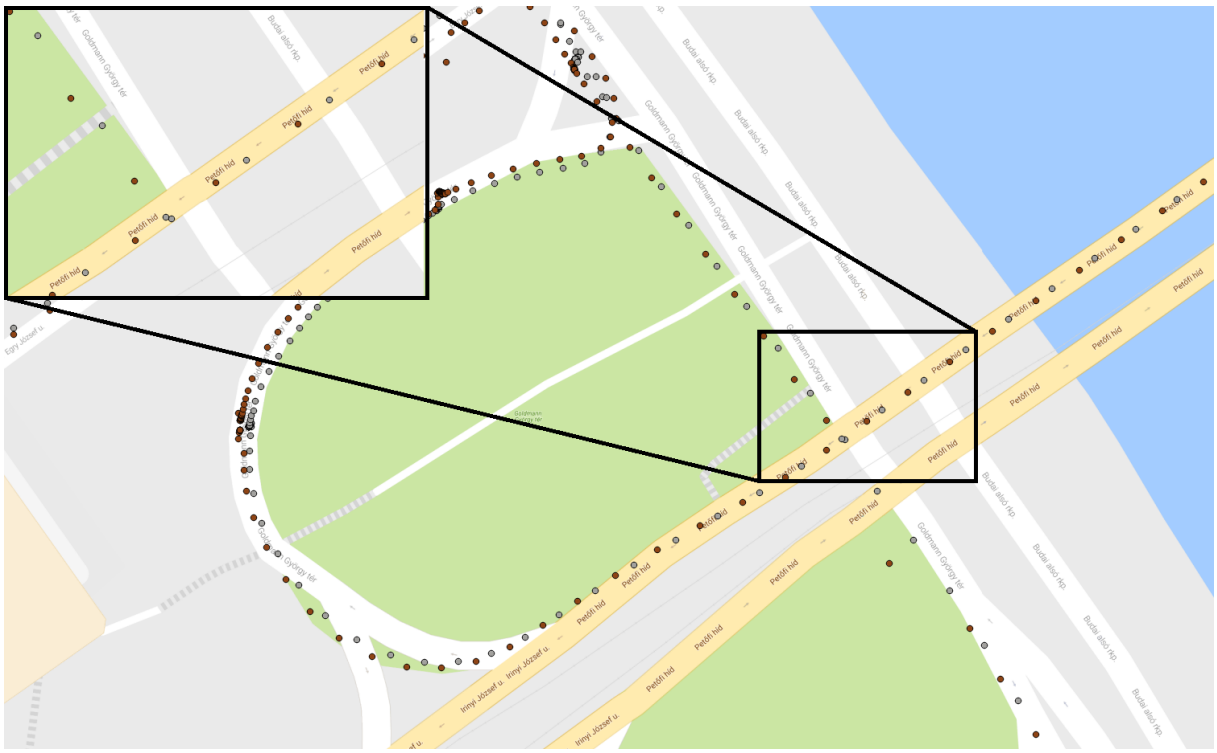
16. ÁBRA U - BLOX TESZTMÉRÉS, TELJES SZAKASZ

16. ábra mutatja a teljes útvonalat, amit a U – blox vevő rögzített. Szépen látszik, hogy nem mindenhol van pozíció. Magas épületek között, vagy felüljárok alatt, nincs földi koordináta. Nyílt területeken, mint a rakparton végig láthatunk pozíciókat. Gyorsan megtörténik a kiegyenlítés és újra van pozíció.

17. ábra a kódmérés és fázismérés teszteredmények összehasonlítását mutatja. Szürke színnel a kódmérési pozíciókat láthatjuk, barna színnel a fázismérést végző vevő koordinátáit láthatjuk. Kódmérésnél folyamatosan van koordináta, viszont fázismérés végrehajtása esetén nem minden epochában van megoldásunk.



17. ÁBRA GPS ADAFRUIT ÉS U - BLOX TESZTMÉRÉSEK ÖSSZEHASONLÍTÁSA

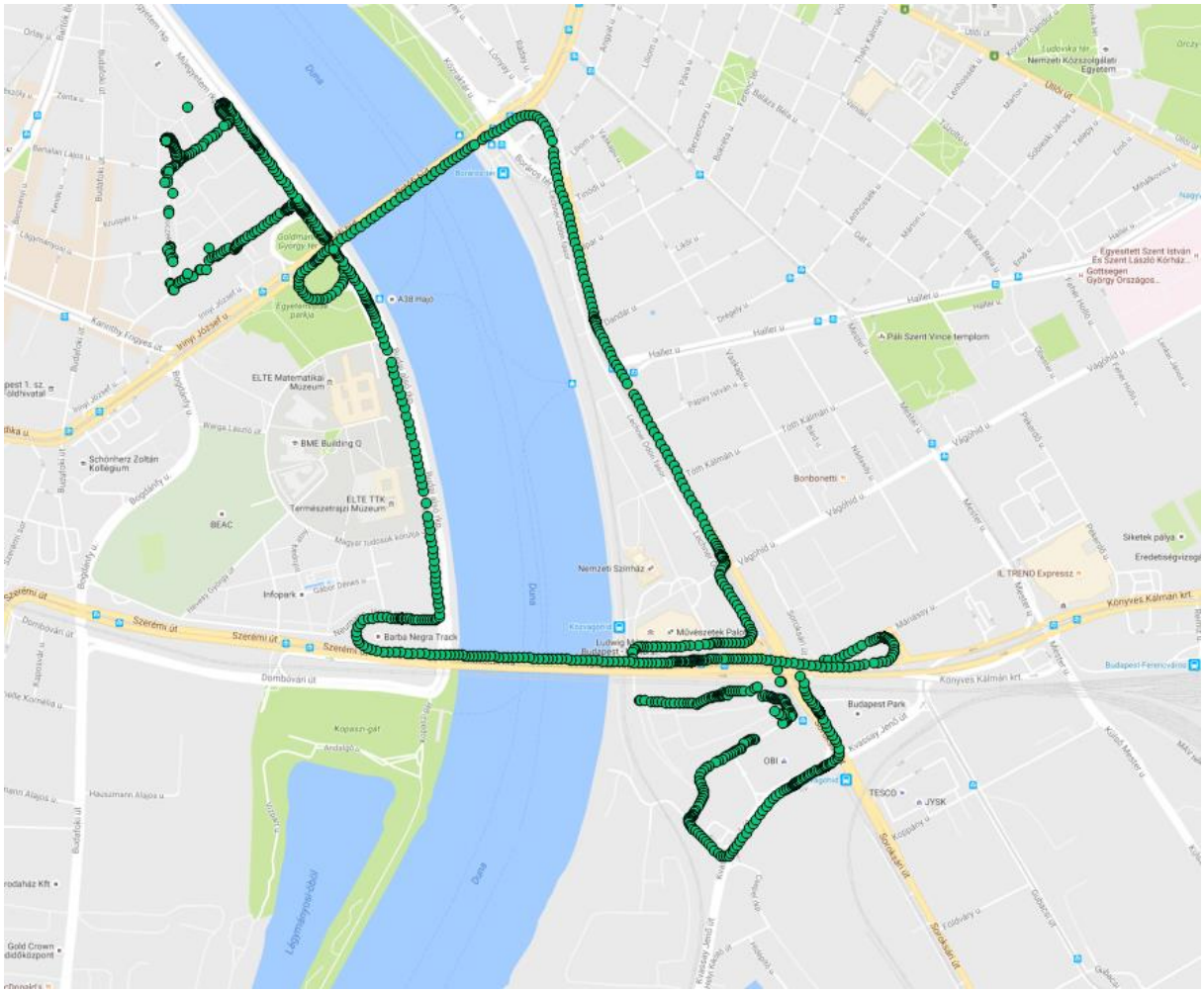


18. ÁBRA GPS ADAFRUIT ÉS U - BLOX TESZTMÉRÉSEK ÖSSZEHASONLÍTÁSA

Flottakövetéshez a technológia részben megfelel. Több helyen is van pozíció vesztés, viszont ezeket ki lehet pótolni kódméréssel. Így az eszköz a városközpontban kódmérésre váltana, ahol előnytelen a műholdgeometria, majd az autópályán újra a fázismérést alkalmazná. A hatékony flottakövetésből nem lehet kihagyni a kódmérésen alapuló technológiát, viszont a pozíció pontosságát javítani lehet, akár már egy frekvenciás eszközzel. Ezzel meg lehet alapozni a 6.2 pontban részletezett fejlesztéseket.

4.3. Relatív helymeghatározás fázisméréssel, többfrekvenciás vevővel

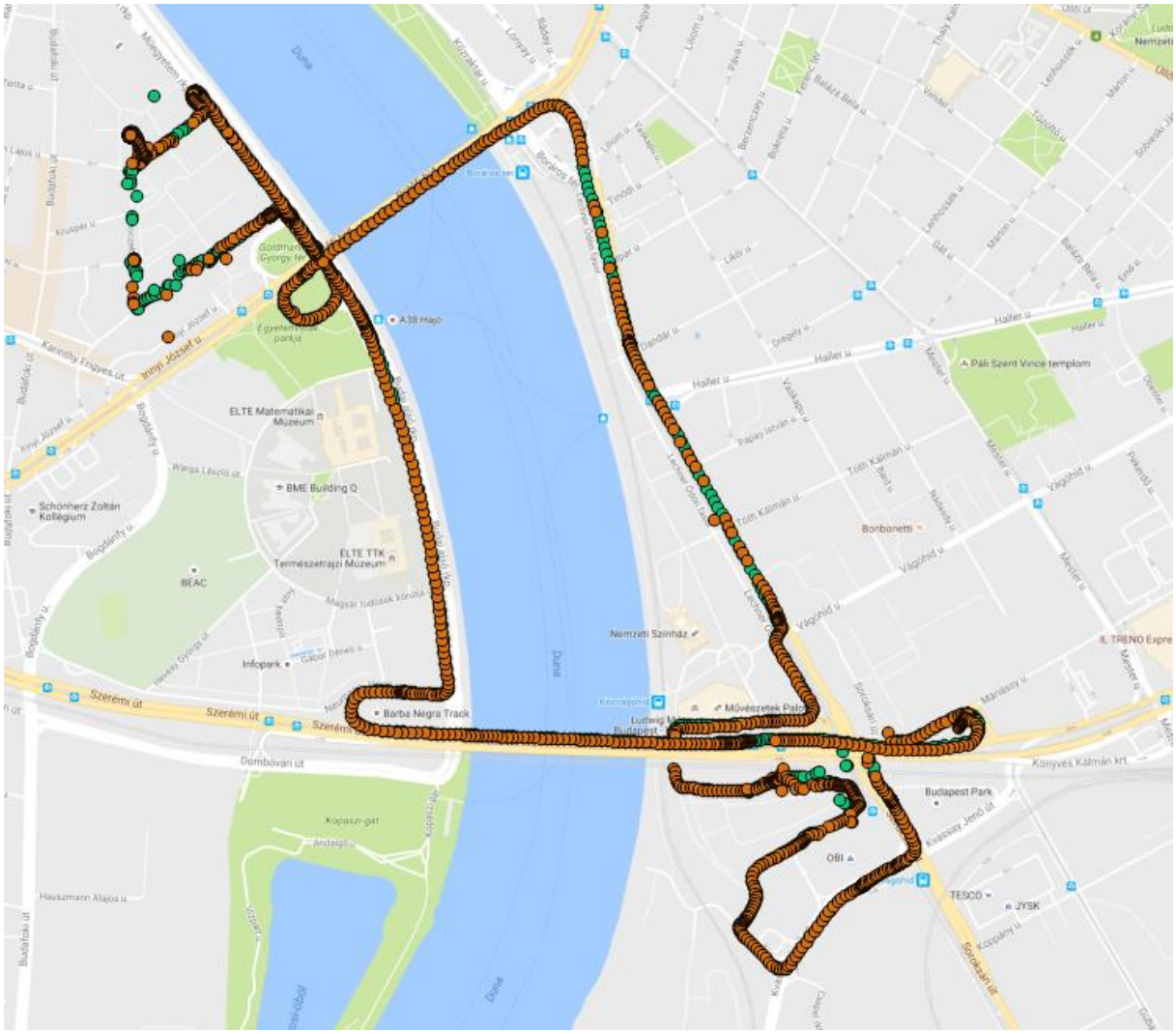
Topcon vevőn keresztül használtam ezt a mérési technikát. Célom egy olyan referencia mérés előállítása volt, amiről biztosan tudtam, hogy pontosabb, mint a korábbi eszközök és viszonyítási alapként és esetleges hibák kimutatására tudom használni. Ciklustöbbértelműség megoldása egy egész szám, ami fix pozíciót eredményez. Ez centiméteres pontosságot jelent. Az eszköz közvetlenül nem kapcsolódik a szolgáltatáshoz, a teszteredményeket utólag hasonlítottam össze. Valós idejű feldolgozással álltak elő a pozíciók, a FÖMI által szolgáltatott korrekciókat használta az eszköz.



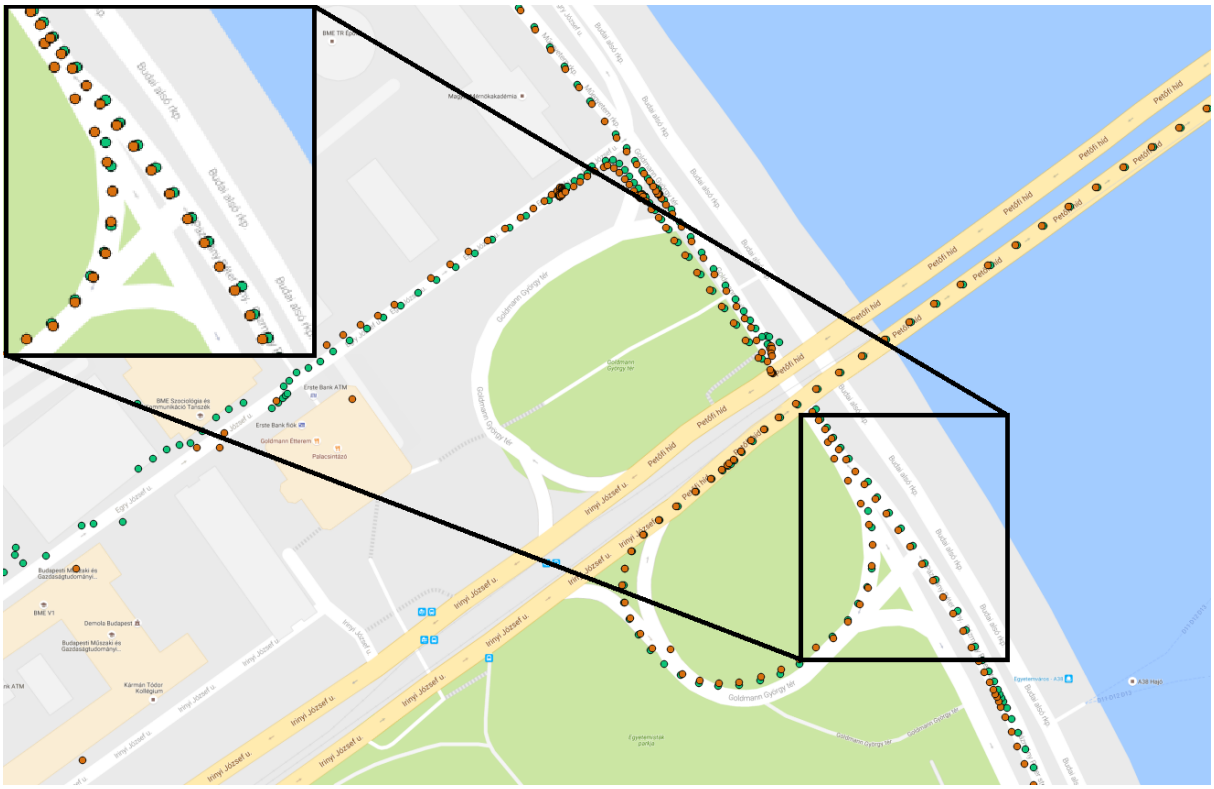
19. ÁBRA TOPCON TESZTMÉRÉS, TELJES SZAKASZ

19. ábra reprezentálja a Topcon vevő által kirajzolt útszakaszt. Itt is megfigyelhetjük, hogy magas épületek között, vagy felüljáró alatt nincsen pozíció. Nyílt terepen, hídon, rakparton pontos koordinátákat láthatunk. Az inicializálás gyorsan történik a kétfrekvenciás méréseknek és a korrekciós adatoknak köszönhetően. 20. ábra már a Topcon vevő pontjait zöld színnel jelölve, míg a U – Blox vevő pozícióait barna színnel rajzolva mutatja. Egy két koordinátával többet rögzített a Topcon vevő, de markánsan nem különböznek. A készülékek határoznak meg pozíciókat minden olyan másodpercben, amikor elegendő számú műhold kapcsolat állt fönt mindkét vevő részéről. A 21. ábra ábrázolja az előbb látott útszakasz felnagyított részét. A pontsűrűségből ki lehet tehát jelenteni, hogy a U – blox vevő ugyanolyan hatékonysággal végzi a munkát ebben a feladatkörben, mint több frekvenciás társa. Viszont ez az eszköz, nagyságrendekkel olcsóbb, mint egy geodéziai pontosságú vevő.

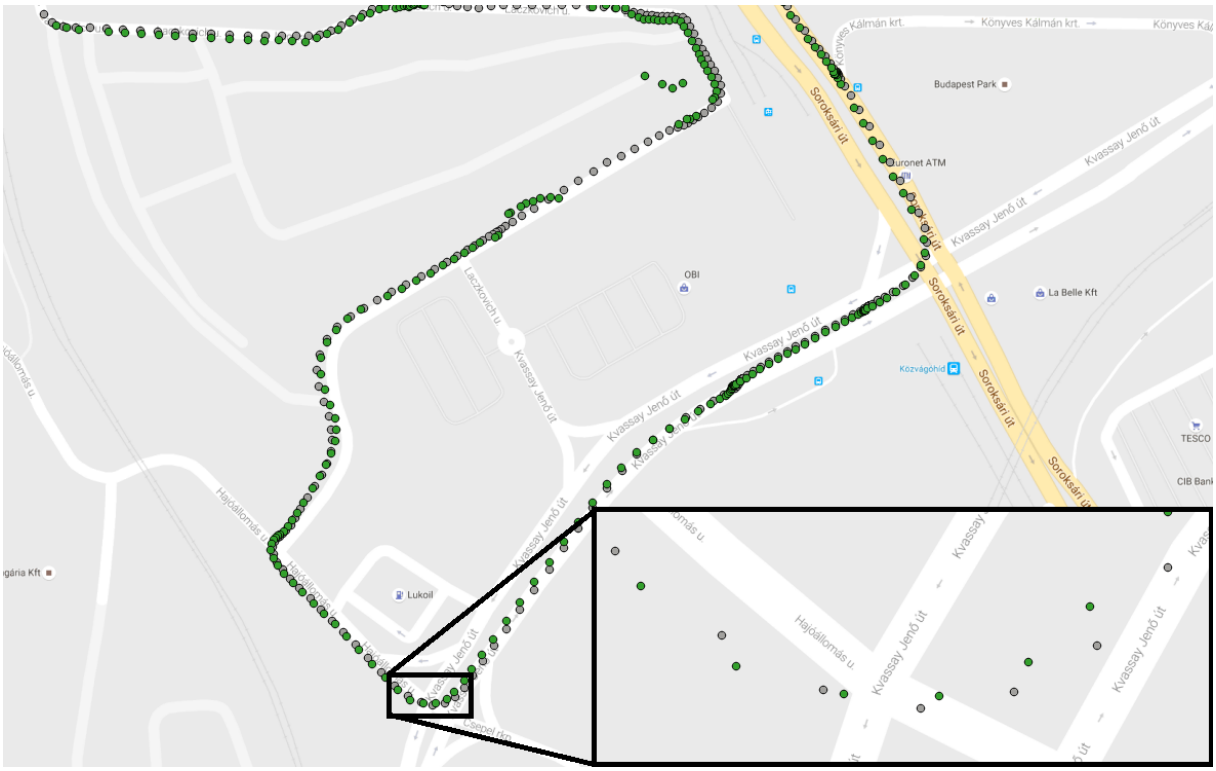
22. ábra ábrázolja a kódmérés és a relatív többfrekvenciás fázismérés összehasonlítását. Zöld színnel láthatjuk a Topcon vevő eredményeit, míg a szürke színnel az Adafruit GPS pozícióit. Tesztmérésből látszik, hogy az épületek között a FIX pozíció gyorsan elveszik, viszont a kódmérés változatlanul szolgáltat pozíciókat. A szolgáltatás szempontjából ez előnyös, mivel kisebb arra az esély, hogy a gépjárművet teljesen szem elől veszítenénk. Amennyiben közelebbről vizsgáljuk az ábrát, akkor kiderül, hogy átlagosan 1-2 méteres különbség van a referencia méréshez képest. Geodéziai célokra ez nem megfelelő, viszont a szolgáltatás igényeit bőven kielégíti, gyakorlatilag, ha már azt tudja az operátor, hogy melyik utcában halad a gépjármű, az már teljesen elegendő a számára.



20. ÁBRA TOPCON ÉS U-BLOX TESZTMÉRÉS, ÖSSZEHASONLÍTÁSA

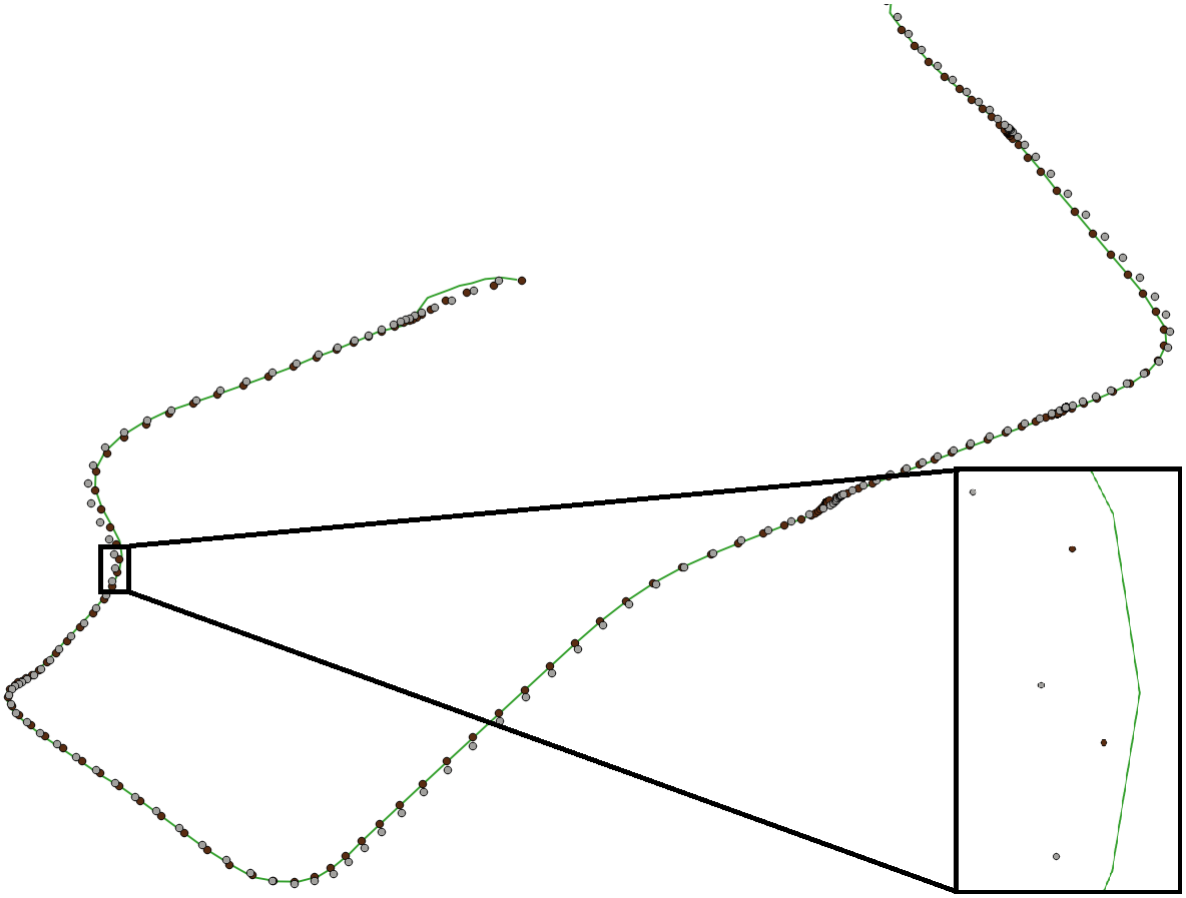


21. ÁBRA TOPCON ÉS U-BLOX TESZTMÉRÉS, ÖSSZEHASONLÍTÁSA



22. ÁBRA TOPCON ÉS ADAFRUIT GPS ÖSSZEHASONLÍTÁSA

4.4. Statisztika a tesztmérésekből



23. ÁBRA TESZTMÉRÉS SORÁN FELHASZNÁLT ESZKÖZÖK ÖSSZEHASONLÍTÁSA

Ebben a fejezetben a tesztmérések statisztikai feldolgozásáról lesz szó. A többfrekvenciás vevő mérési eredményeit referenciaként használtam föl, az általa készített pontokból vonalláncot hoztam létre. A létrejött nyomvonaltól való a különböző technikákkal megmért pontok távolságait vizsgáltam. 23. ábra mutatja a mérésekből kiválasztott szakaszt. A mintavétel kicsivel több mint 250 pontot tartalmaz. Zöld vonallánc a referencia mérést a nyomvonalat ábrázolja, zöld színnel láthatjuk a kódméréssel készült földi koordinátákat, míg a barna színű pontok az egy frekvenciás mérést jelölik. Teljes szakasz vizsgálata rontotta volna a statisztikát a jelvesztések miatt. A mintavétel földrajzi helyét a 22. ábra segítségével könnyen meg lehet határozni.

	Egy frekvenciás fázismérés [m]	Kódmérés [m]
Maximális eltérés	3.91	5.443
Minimális eltérés	0.0002	0.04
Átlagos eltérés	0.81	1.22

A maximális eltérés a durva hibára enged következtetni, míg az átlagos eltérés a szabályos hibára. Kimondhatjuk, hogy az egy frekvenciás fázismméréssel pontosabban tudunk pozíciót

meghatározni, mint kódméréssel, viszont gyakran szubméteres pontosságot is eltudjuk érni. A statisztika a várt eredményeket igazolja és további fejlesztési lehetőségeket vezet be, amit későbbiekben részletezek.

5. Összegzés

A szolgáltatás tehát felépül egy kódmérés technológiát használó GPS -ből, az Adafruit Ultimate GPS modulból, ami a 3.3.2 pontban részletezett módon tölti föl az adatokat a webszervernek, majd azok mentésre kerülnek az adatbázisba, amelynek módját részletesen taglaltam a 3.2 fejezetben. Az így összeállt nyílt forráskódú alapokon nyugvó szolgáltatás képes kielégíteni azokat a szükségleteket, amivel a szállítmányozással foglalkozó cégek rendelkeznek, legalább annyira, mint a már piacon lévő hasonló célra szánt rendszerek.

Továbbá kiderült a tesztmérések a során, amit az 4. pontban fejtettem ki, hogy kódmérés önmagában is elegendő a flottakövetéshez, mivel a méteres pozíció bőven elegendő a tehergépjárművek helymeghatározásához. Hiába pontatlanabb frekvenciamérésre alkalmas társainál. A tesztelés kimutatta, hogy a kódmérés eredménye a referencia méréshez képesti különbség közel elhanyagolható, a szolgáltatás pontossági igényei miatt.

6. Célkitűzés, tovább fejlesztés

Általános célkitűzés a szolgáltatás éles üzemben történő alakalmazásra történő felkészítése. Elsősorban a szerver oldalon van fejlesztésre szükség, hogy minél több felhasználót tudjon kiszolgálni a rendszer, ne lehessen túlterhelni. Biztonság megerősítése esetleges támadások ellen. Formai alakításokra is szükség van, hogy a GPS kompakt legyen, minél könnyebben be lehessen szerelni egy gépjárműbe. Továbbá lehet javítani az internet kapcsolaton, illetve antennázási módon. Ezek az előre lépések nem elméleti szinten adnak hozzá a szolgáltatáshoz, inkább a gyakorlati hasznosságukat növelik.

6.1. További érzékelők beépítése

Ezt is inkább célkitűzésnek mondanám, mint fejlesztésnek, mivel a piacon már vannak ilyen jellegű kiegészítő szolgáltatások. Mint például üzemanyag szint regisztrálás, billenés figyelés, vezetői stílus monitoring vagy munkaóra követés. Ezekhez a rész funkciókhoz a kialakított infrastruktúra elegendő, csak bővíteni kell az adatbázist, hogy ilyen jellegű adatokat is tárolni tudjon, illetve a webszervert, hogy tudja fogadni ezeket az információkat.

6.2. Mezőgazdasági gépek

U – blox vevő nyílt területen való használata során, olyan pontosságú pozíciókat tud küldeni, ami akár elég lehet egy nagyobb földművelésre szánt gép önvezető rendszerének támogatásához. A szolgáltatás jelen állapotára erre a felhasználásra félig van felkészítve, mivel az adatokat tudja tárolni, de koordinációs információkat - ami a valós idejű irányításhoz szükséges - nem tud visszaküldeni. Akár ez is része lehet a teljes szolgáltatásnak.

6.3. Kerékpárkövetés és SpyBike

Piacon igény van a kerékpárjaink követésére is, megfigyelésére, nyilván ennek más oka van, de a rendszer szemszögéből nincs különbség. Tehát a már kialakított infrastruktúra elegendő egy kerékpár monitoring szolgáltatás hozzáadásához. A kódmérés helymeghatározáshoz ebben az esetben teljesen elegendő, ami előnyös, mert önmagában ezek a készülékek nem drágák, ami fontos egy kerékpártulajdonos számára.

Hasonló termék van a piacon, „SpyBike” névre hallgat. Szándékom az is, hogy ezt a terméket integráljam a rendszerbe. Ehhez a készülék által küldött jelek feldolgozására van szükség. Nem szabványos NMEA üzenetet használ, illetve nem http protocol segítségivel kapcsolódik a kijelölt szerverhez, ami egyelőre még korlátozza a beépítés megvalósulását.

6.4. Okos eszköz figyelés

Láthattuk a 3.3.4 pontban, hogy elérhető az okostelefonok helymeghatározó modulja és ki lehet olvasni a pozíciókat, amiket ebben az esetben feltöltöttem a rendszerbe. Olyan alkalmazások fejlesztése a cél, androidra, IOS -re és Windows Phone -ra, ami a háttérben működik folyamatosan és szokatlan használatkor pozíciókat tölt fel a szerverre. Például helytelen PIN kód megadása után, elküldi koordinátáit a rendszernek, így a tulajdonos gyorsabban tud reagálni a problémára.

7. Ábrajegyzék