Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

Bevíz Péter

Akusztikus távolságmérésen alapuló lokalizáció szenzorhálózatban

KONZULENS

Orosz György

BUDAPEST, 2011

Tartalomjegyzék

HALLGATÓI NYILATKOZAT

Alulírott **Bevíz Péter**, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2011. 05. 13.

.....
Bevíz Péter

Összefoglaló

Szakdolgozatomban egy olyan akusztikus távolságmérésen alapuló lokalizációs rendszert terveztem meg, mely segítségével meghatározható egy vezetéknélküli hálózatban az egyes szenzorok egymáshoz viszonyított távolsága. A távolságmérés a szakirodalomban TDOA (Time Difference of Arrival) néven ismert elven alapszik, mely során az egyidőben kiadott rádiós és akusztikus jel beérkezési időkülönbségének felhasználásával történik a mérés.

A kiválasztott lokalizációs algoritmust Berkley micaz típusú mote-okon implementáltam. Létrehoztam egy PC-s szoftverkörnyezetet is, mely segítségével a szenzorhálózat lokalizációs tevékenysége vezérelhető, és a mérési adatok feldolgozhatóak.

Mivel a mérési eredmények gyakran pontatlanok, így munkám során különös figyelmet fordítottam a potenciális hibalehetőségek kiküszöbölésére és a mérési hibák hatásának enyhítésére.

A dolgozatban bemutatom a megvalósított rendszeren végzett mérések eredményeit. Ezek alapján elmondható, hogy a rendszer 1-2 m-en belül körülbelül 10-20 cm pontosságú mérést tesz lehetővé, míg ezen távolságon kívül a mérés pontossága jelentősen csökken.

Abstract

This thesis report describes a wireless sensor network (WSN), which uses an acoustic distance measurement to determine the distance between the sensor nodes. The system uses the method known as TDOA (Time Difference of Arrival) for distance measurement. The principle of the operation is that a sensor node transmits an acoustic and radio signal at the same time, meanwhile an observer node measures the arrival time difference between these two signals.

The localization algorithm was implemented on Berkley micaz motes. Some software modules were also developed in MATLAB in order to control the operation of the sensor network and process the measurement data on PC.

Since the measurements are often perturbed by errors, special emphasize was put on the identification and elimination of the potential error sources as far as possible.

The measurements carried out on the system show that the distance measurement can be performed with precision 10-20 cm or better within the distance 1-2 m, and the precision degrades considerably above this distance. In this case, the system can be used to obtain qualitative information, e.g., whether a sensor node is in the vicinity of an other node or not.

1. Bevezetés

A szenzorhálózatok manapság egyre több területen elterjedő, intelligens egységekből felépülő mérőrendszerek. Mivel a szenzorhálózatokban a kommunikáció általában vezetéknélküli csatornán keresztül történik, így a rendszer nagyfokú szabadságot biztosít a szenzorok elhelyezésével kapcsolatban. Ennek köszönhetően a vezetéknélküli szenzorhálózatokban az egyes szenzorok pozíciója sok esetben bizonytalan, vagy egyáltalán nem ismert. Bizonyos mérési adatok értelmezéséhez azonban a térbeli pozíció ismerete is szükséges. Ezen térbeli pozíciók automatizált meghatározása úgynevezett lokalizációs eljárásokat alkalmazhatunk.

A lokalizációs feladatok általában két alapvető lépésre bonthatóak. Első lépésben valamilyen elsődleges információt nyerünk a szenzorok elhelyezkedéséről, például megmérjük a szenzorok közötti távolságot, és aztán az így kapott mérésekből valamilyen magasabb szintű információt állítunk elő, például a hálózat geometriai felépítésére vonatkozóan. A dolgozatom célja a lokalizációs feladat első fázisának, jelen esetben a hálózati csomópontok közötti távolságok mérésének megoldása.

A feladat megoldásához a Crossbow cég által gyártott Berkley micaz mote-okat használtam fel. A lokalizációs technika kiválasztásában a mote család tulajdonságait is figyelembe vettük. A micaz mote alapkártyája rendelkezik rádiófrekvenciás tartományban kommunikáló áramkörrel, továbbá a mote-okra csatlakoztatható szenzorkártya tartalmaz egy hang kiadására alkalmas buzzer egységet, valamint a hang detektálására alkalmas mikrofont is. A távolságméréshez használt módszer az úgynevezett érkezési idők különbségén alapuló távolságmérési technika, mely a szakirodalomban TDOA (Time Difference o Arrival) néven ismert.

A dolgozatban jelentős szerepet kapnak a mérési eredmények feldolgozásának módszerei. A szenzornode-ok által szolgáltatott mérések ugyanis alapvetően sok hibával terheltek, így elkerülhetetlen egy vagy akár több előfeldolgozási lépés megvalósítása.

A 2. és a 3. fejezetben bemutatom a szenzorhálózatok fogalmát, és az alapvető mérési és lokalizációs technikákat. A már gyakorlatban is használt rendszerek bemutatása után leírom a helymeghatározó rendszerek legjellemzőbb tulajdonságait.

A 4. fejezetben a felhasznált hardvereket, a Berkley micaz mote-okat mutatom be. Ismertetem a szenzorhálózatban használt TinyOS operációs rendszert és a NesC

programnyelvet, amiben a szenzorokon futó programokat implementáltam. A szenzorhálózatok esetén az adatfeldolgozás általában egy nagy teljesítményű központi egységen történik, esetemben ez a PC. A PC-n az adatok feldolgozásához használt MATLAB környezetet is bemutatom.

Az 5. fejezet az általam tervezett rendszerben alkalmazott mérési elvről, a felmerülő hibalehetőségekről szól, valamint bemutatja a szenzorok közti kommunikációban alkalmazott üzenettípusokat, és a szenzorhálózat általános működési mechanizmusát.

A 6. fejezet a hibás mérési eredmények feldolgozásával, hibák kiszűrésével és a megbízható működéshez elengedhetetlen kalibrációról szól.

A 7. fejezetben az implementációs eredményeket mutatom be. Bemutatásra kerül a szenzorokon futó és a PC-n az adatok feldolgozását és a mérés vezérlését végző programok.

A 8. fejezetben mérési eredményeket ismertetek. Több mérési távolság és kalibrációs tartomány esetében bemutatom a távolságmérő rendszer működését és annak pontosságát.

A 9. fejezet ismerteti az összefoglalást és a továbbfejlesztési lehetőségeket.

2. Szenzorhálózatok

2.1. Általános bemutatás

A szenzorhálózatok a környezettel szorosan kapcsolatban álló, körülbelül gyufásdoboz méretű, intelligens, esetleg mobil eszközök százainak szoros együttműködése, melyek próbálják kényelmesebbé tenni világunkat. Ezeket az eszközöket nevezzük intelligens pornak, vagy idegen kifejezéssel *smart dust*-nak. Az apró eszközök "telepítése" abból áll, hogy megfelelően nagy számban kihelyezzük őket, majd a szenzorok autonóm módon hálózatba szerveződnek. A miniatűr, MEMS (mikro-elekro-mechanikai) technológiával készített szenzorok és beavatkozók ma már egyre apróbbak, de igazán porszemnyi eszközök még nem léteznek. A mai szenzorhálózatok a smart dust jövőképének működő előhírnökei.

A szenzorhálózatok sok intelligens érzékelő egységből felépített, önálló működésre képes elosztott számítógépes hálózatok. Ezek a szenzorok a tér különböző pontjain méréseket esetleg beavatkozó tevékenységeket is végezhetnek. Az alapvető adatgyűjtő funkción kívül az érzékelő hálózat képes adatfeldolgozás és analízis jellegű feladatok elvégzésére is. Az alacsony áruknak és a kis méretüknek köszönhetően a szenzorok a megfigyelt térrészben vagy annak közelében viszonylag sűrűn és nagy számban helyezkedhetnek el.

Minden szenzor tartalmaz:

- saját érzékelő- és esetleg beavatkozó egységet
- saját számítási egységet (mikrokontrollert),
- saját kommunikációs egységet (általában rádiót), valamint
- saját tápellátást (tipikusan szárazelemet vagy fényelemet).

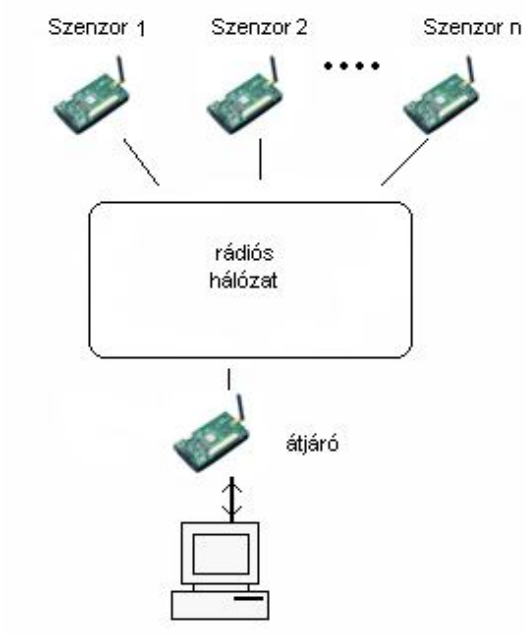
Az alacsony ár és a kis méret következtében a szenzorok teljesítőképessége erősen korlátozott mind a számítási kapacitás, a memória, a kommunikációs képességek és a tápellátás tekintetében.

Például az általam használt Berkeley micaZ mote egy 7,3721 MHz-es, 8 bites mikrokontrollert használ 128 kB program- és 4 kB adatmemóriával. A rádió 250 kbps sebességgel képes maximum 100 méter távolságra kommunikálni, mely azonban zárt

térben jelentősen csökkenhet. A tápellátást két ceruzaelem biztosítja, amellyel az egység folyamatosan néhány napig, "altatva" pedig néhány hónapig képes üzemelni.

A mai tipikus szenzorhálózatok elemszáma a néhány tucattól a több százig terjed, de teszteltek már több ezer szenzort tartalmazó hálózatokat is. A kísérleti eszközök előállítási költsége kb.10 ezer Ft/db. Az egységek méretét ma elsősorban a tápellátás határozza meg, így a gyufásdoboznyi (ceruzaelemek), vagy nagyobb gombelemek elemek általánosak, de létezik már intelligens szenzor 2.5mm x 2.5mm lapkára integrálva is, olvashatjuk az [1] irodalomban.

A szenzorhálózatok általános felépítésére jellemző, hogy az adatgyűjtő szenzoregységek hálózatba rendeződnek, és a szenzornode-ok esetleg még kisebb feldolgozó műveleteket is végeznek az érzékelt adatokkal. Ezek a szenzorok az adatokat továbbítják egy átjáróként működő szenzorhoz, amely a mért adatokat továbbítja egy feldolgozó egységhez. Ez lehet PC, PLC vagy akármilyen, általában nagy teljesítményű adatfeldolgozó egység. A 2.1. ábrán a szenzorhálózatok egy általános felépítése látható.



2.1. ábra. A szenzorhálózatok általános felépítése

2.2. Követelmények

A szenzorhálózatok mérete a felhasznált szenzorok számának növekedésével nagyságrendileg változhat. Ezt a megváltozást a hálózatnak jelentősebb teljesítménybeli, illetve a működés minőségének számottevő romlása nélkül tolerálnia kell. Ezt nevezzük skálázhatóságnak. A lokalizációs módszer kialakítást nagyban befolyásoló tényezők a költség, valamint a használhatósági korlátok, mivel egyes rendszerek nem működnek bizonyos körülmények között. A GPS rendszerek például nagy hatótávolsággal rendelkeznek, de drágák. A rádiós adások térerőssége alapján történő lokalizáció egyszerű, de pontatlan.

A hálózatnak önállóan is képes kell lennie tolerálni a változásokat, el kell tudnia indítani a rendszer működését, mivel a nagy elemszám miatt az operátori beavatkozás igen nehézkes. Ez az önszerveződés igénye.

A szenzorhálózatok működését, köszönhetően a kis energiaigények, hosszabb időre akár több évre is tervezik jelentősebb szervizelés átalakítás nélkül.

Minél összetettebb tulajdonságú, pontosabb működésre képes szenzorok használata esetén nő az energiafelvétel, valamint az eszközök gyártásának költsége is megemelkedik. Ezért kompromisszumokat kell kötni a hálózatban alkalmazott érzékelők mérésének megbízhatósága és az ár között.

Amennyiben az hálózat mérete olyannyira megnő, hogy egy központi egység már nem képes az adatok fogadására és kezelésére az adott sávszélességen, akkor itt hálózaton belüli elosztott feldolgozás, analízis, illetve aggregáció szükséges.

2.3. Szenzorok közti kommunikációs csatornák fajtái

A szenzorhálózatokban az átvitel módja lehet rádiós, infravörös tartományba tartozó vagy ultrahangos. Mindegyik lehetőségnek megvannak az előnyei és a hátrányai is.

Az **infravörös** tartományba tartozó hullámok nem sokkal a látható frekvenciaspektrum alatt helyezkednek el. A hőt kibocsátó testek, berendezések, fénycsövek megzavarhatják a megbízható működésben. A sebessége körülbelül megegyezik a rádióhullámok sebességével, tehát $3 \cdot 10^8$ m/s. A hatótávolsága hozzávetőleg 5 méter. Falakon nem képes áthatolni.

Az **ultrahangos** érzékelés előnye az alacsony ár. Viszonylag kis frekvencián, 40 kHz-en üzemel. A terjedési sebesség a hangéval megegyező 343 m/s. A legtöbb beltéri akadályról visszaverődik a hatótávolság maximum 10 méter, valamint ez a módszer érzékeny a környezet hőmérsékletének megváltozására is.

A **rádiófrekvenciás** kommunikáció esetében érhető el a legnagyobb távolság, mivel a legtöbb akadályon és falakon képes áthatolni a rádióhullám, valamint több frekvenciasávon képes üzemelni. A szenzorhálózatokban alkalmazott tipikus rádiósávok az ún. ISM (industrial, scientific and medical) sávokban találhatók, például 433 MHz, 868 MHz, 915 MHz, 2,4 GHz a működési frekvenciák. A jelterjedési sebesség itt is nagyjából megegyezik az elektromágneses hullám vákuumbeli terjedési sebességével, amely $3 \cdot 10^8$ m/s.

A vezetéknélküli szenzorhálózatok kommunikációs sávszélessége általában viszonylag alacsony: néhány tíz vagy száz kbps (kilobit per second) tartományban található. Ennek oka energiatakarékossági megfontolásokban keresendő. Mivel az általam használt, a később bemutatásra kerülő módszer segítségével az egyes szenzorpárok lokálisan határozzák meg az egymáshoz viszonyított távolságukat, így nem igényli nagy mennyiségű adat továbbítását. Ez a kommunikációs csatorna kihasználásának szempontjából kedvező, nem igényel nagy adatátviteli sebességet.

2.4. Tipikus alkalmazások

A vezeték nélküli szenzorhálózatok használata, fejlesztése eleinte a hadiiparban, hadászatban figyelhető meg a 1970-es évektől. Azonban a vezeték nélküli szenzorhálózatok (Wirless sensor networks: WSNs) gyorsan elterjedtek a civil felhasználók körében, és egyre jelentősebb szerepet töltenek be a mindennapi élet számos területén. Köszönhető ez a szenzorok árának rohamos csökkenésének, alacsony energia-felvételnek, valamint a kisméretű, de egyre bonyolultabb funkciók ellátására képes érzékelőknek. Az olcsó, intelligens szenzorok tömegének hálózata soha nem látott lehetőségeket kínálnak háztartások, városok és a környezet megfigyelésére és irányítására. Ezen túlmenően a hálózati érzékelők széles spektrumát alkalmazzák lakóépületek, gyárterületek őrzésére, illetéktelen behatolók jelzésére, mint riasztórendszerek érzékelői, valamint új területe a szenzorhálózatok használatának a

felderítési és felügyeleti feladatok ellátása is. A következő felsorolásban néhány tipikus alkalmazási terület olvasható.

Egészségügy: kórházi menedzsment, katasztrófa-elhárítás (pozíció- és állapotszolgáltatás, vészriasztás), otthoni betegellátás (tevékenység-monitorozás, riasztás)
Gyártás, raktározás: gyártósor monitorozás, készletnyilvántartás (intelligens raktárak, vasúti szerelvények, tartálykocsik, tankhajók)
Környezetvédelem: élőhely-monitorozás (állatok viselkedése, növény-társulások mikroklímája), katasztrófa-előrejelzés (árvíz, vulkánkitörés)
Mezőgazdaság: „precíziós mezőgazdaság" (szőlőtermesztés: öntözés, növényvédelem, szüret igény szerint)
Mérnöki alkalmazások: épületek, műtárgyak monitorozása (hidak statikai ellenőrzése), közlekedésfelügyelet, intelligens épületek
Védelmi alkalmazások: megfigyelés, követés, detektálás (elektronikus kerítés), lokalizálás (akusztikus lövés-lokalizáció)
Űrkutatás: Mars-szondák

A dolgozat szempontjából különös figyelmet érdemelnek azon rendszerek, ahol a szenzorhálózatokat valamilyen lokalizációs feladatra használják. Példaként említhető az orvlövészek helyének azonosítása a lövés által keltett hanghullámok (a durranás, vagy hangtompítós szuperszonikus fegyverek esetén a lökéshullám) alapján [2], vagy különböző állatok vonulási szokásainak megfigyelése a nyakukra erősített kis szenzorral.

3. Helymeghatározás

Az ön-lokalizációs képesség fontos követelménye a vezeték nélküli szenzorhálózatoknak. Környezetmonitorozási feladatokban, mint például tűzjelző berendezéseknél, vízminőség mérésénél és precíziós mezőgazdasági feladatoknál elengedhetetlen, hogy tudjuk hol mértük az adott eredményeket, a pontos helymeghatározás nélkül értelmét veszítik a mérések. Azonban lehet olyan eset is, ahol nem vagyunk kíváncsiak a pontos koordinátákra, hanem elég, ha csak szimbolikusan ismerjük a pozíciókat, például azt akarjuk megtudni, hogy mozog-e az adott szenzor esetleg távolodik, közeledik.

Azokat a szenzorokat, amelyeknek ismert a pozíciója anchor-oknak [5] nevezzük (magyarul: vasmacska, horgony). Ezen egységek helyzete meghatározható például globális helymeghatározó rendszerrel (GPS), vagy már a telepítési koordinátái ismertek (pl. a telepítést végző személy manuálisan megadja). A GPS-t használó rendszerek esetében ezeknek az anchorok-nak a helyzete a globális koordinátarendszerben lesz meghatározva. Azokban az esetekben ahol elégséges egy helyi koordinátarendszer (például intelligens otthonok), ezek a pozícióval rendelkező szenzorok határozzák meg a koordinátarendszert és a többi érzékelő ezekhez a rögzített pozícióval rendelkező „vasmacskákhoz" képest határozza meg a pozícióját. Az ár és a szenzorméret korlátai, energiaellátás nagysága miatt a GPS rendszer nem alkalmazható minden esetben, valamint a véletlenszerűen telepített szenzorhálózatok esetében a legtöbb szenzornak nem ismerjük az elhelyezkedését. Ezeket az ismeretlen pozíciójú eszközöket non-anchor-oknak hívjuk. Ezen csomópontok koordinátáját a szenzorhálózat lokalizációs algoritmusa alapján lehet meghatározni.

Az irodalomjegyzékben lokalizációs technikákat mutat be a [4] irodalomban. Ezekben a hangsúly a mobil hálózatok, a vezeték nélküli helyi hálózatok (WLAN) és a jelfeldolgozás szempontjából fontos lokalizációs technikákon van.

A szenzorhálózatok jelentősen eltérnek a hagyományos mobil hálózatoktól és a WLAN rendszertől, mivel nagyszámú, apró és olcsó csomópontot tartalmaz. Ezek a tulajdonságok egyedi kihívást jelentenek a szenzorhálózatokban történő lokalizáció során. Az [5] cikkben összefoglalást találhatunk a szenzorhálózatokban leggyakrabban alkalmazott lokalizációs algoritmusokról. A legtöbb mérési módszer, amely 2-

dimenzióban alkalmazható, az kiterjeszthető 3-dimenzióra is. A szenzorhálózatokban használt lokalizációs technikák a mérési elv alapján három kategóriába sorolhatóak: beesési szög (Angle of Arrival: AOA) mérések, távolsághoz kapcsolódó mérések és vett jel erősségén (Received Signal Strength: RSS) alapuló technikák.

3.1. Lokalizáció lépései

A lokalizációs információ lehet fizikai, mely egy konkrét elhelyezkedést meghatározó adat; ilyen például egy épület pontos elhelyezkedését megadó helymeghatározás. A másik lehetőség a szimbolikus lokalizáció, mely már általában kevésbé pontos. Ebben az esetben valamilyen információt/leírást rendelünk az adott egység helyzetéhez, például csak azt határozzuk meg, hogy az adott szenzor a háznak például melyik szobájában található, vagy esetleg járművek esetén a haladási irány detektálása történhet. Ezek a lokalizációs módszerek akár együtt is használhatóak.

A pozíció, amit meghatároztunk, lehet abszolút vagy relatív pozíció [10]. Abszolút pozíciót szolgáltató rendszerekben közös referenciát (gridet) használnak; ilyen például a GPS által szolgáltatott földrajzi szélesség, hosszúság, magasság. A relatív esetben akár objektumonként más-más lehet a viszonyítási keret, például, minden szenzor saját magához képest lokalizál. Az abszolút pozíció könnyen átalakítható relatívvá, de ez fordítva is érvényes megállapítás, ha ismerjük a megfelelő adatokat és referenciapontokat.

Egy szenzorhálózatból álló rendszerben nem elég tudni azt, hogy a szenzorok milyen távolságban helyezkednek el egymástól. A hálózat geometriájának, topológiai felépítésének meghatározása is fontos feladat. Ennek a topológiai rekonstrukciónak a megvalósítására olvashatunk algoritmusokat az [5] irodalomban. A dolgozatban ezekkel az eljárásokkal nem foglalkozok részletesen, mivel a kitűzött cél alapvetően a lokalizációhoz felhasznált távolságmérési eljárás kifejlesztése.

3.2. Kültéri helymeghatározás

Manapság a legkorszerűbb helymeghatározó rendszerek műholdak segítségével határozzák meg egy adott objektum elhelyezkedését. A legelterjedtebb a GPS rendszer. A **GPS** (*Global Positioning System* azaz Globális Helymeghatározó Rendszer) az

Amerikai Egyesült Államok Védelmi Minisztériuma (Department of Defense) által elsődlegesen katonai célokra kifejlesztett és üzemeltetett, Navstar névre keresztelt műholdas helymeghatározó rendszer. A rendszerhez folyamatosan zárkózik fel az Orosz-Indiai GLONASSZ, az Európai Galileo és a Kínai Beidou-2 műholdas rendszer, kiegészítve, pontosítva azt.

A GPS rendszer a nap 24 órájában felhasználható lokalizációs, valós idejű rendszer. A pontosság is figyelemre méltó, mivel méteres hibával képes a Földön lévő ember, épület, tárgy stb. koordinátáinak megadására. További javulást érhetünk el differenciális mérési módszerek segítségével, melyek akár cm vagy mm pontossággal is szolgáltathatnak adatot. A helymeghatározás 24 Föld körül keringő műhold segítségével történik, amelyek a Föld felszíne felett 20200 méteres magasságban keringenek, és az egyenlítővel a pályájuk körülbelül 55°-os szöget zár be. A GPS háromdimenziós lokalizációra is képes. A síkban történő helymeghatározáshoz három, míg a térbeli elhelyezkedés megadásához további egy műhold szükséges [6].

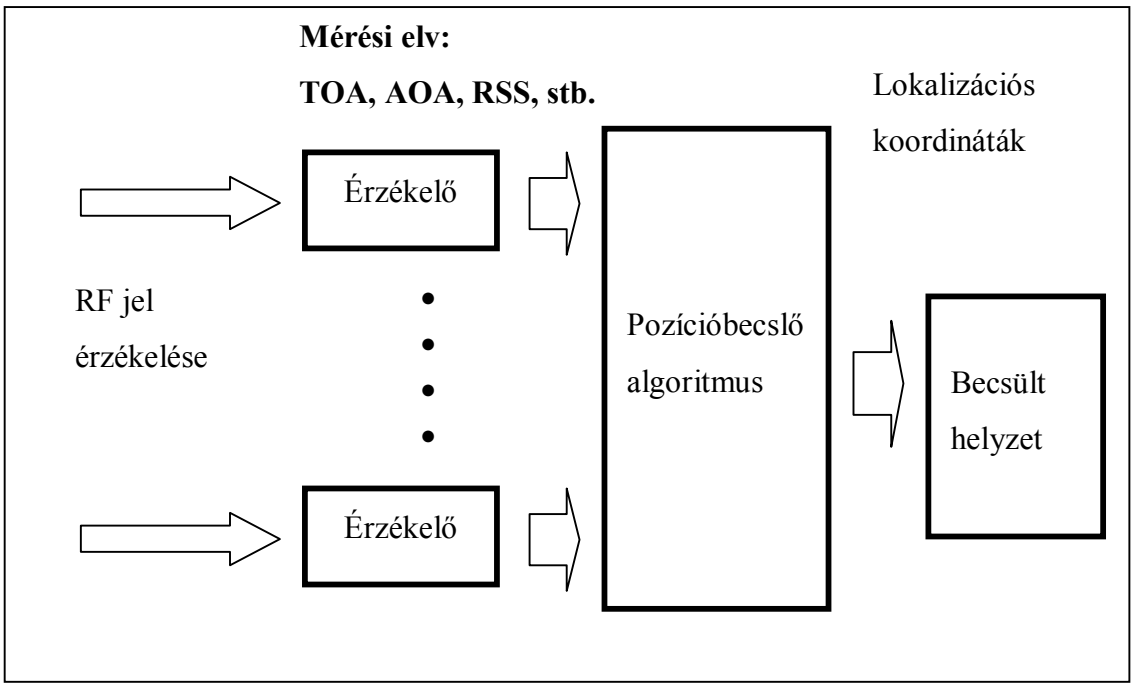
A rendszer legjelentősebb hátránya az, hogy zárt térben, vagy fedett helyen, ahol a műholdaknak nincs rálátása az objektumra, ott nem lehetséges a hely meghatározása, tehát épületek belsejében, alagutakban nem tudunk a globális helymeghatározó rendszer segítségére támaszkodni. Emiatt van szükség beltéri helymeghatározó rendszerekre is, amikkel zárt térben is lehetséges a lokalizáció.

Manapság a GPS rendszer már mindenki számára elérhető és használható, például a Google Earth alkalmazás segítségével a számítógépünkön vagy akár már fejlett mobiltelefonokon keresztül is. A GPS jelentős szerepet játszik még a navigációs rendszerekben is, a legoptimálisabb útvonal megtervezése is a műholdak segítségével történik. A mindennapok már elképzelhetetlenek lennének nélküle.

3.3. Beltéri helymeghatározási rendszerek

Mivel a GPS hatóköre beltérben korlátozott, valamint sok alkalmazásban nem elfogadható a költsége sem, így ezen hátránynak a kiküszöbölésére különböző kutatások irányulnak a beltéri helymeghatározó rendszerek kifejlesztésére is. Ezeknek a lokalizációs módszereknek több alapvető nehézséget át kell hidalniuk. Elsősorban ezeket a problémákat a zárt épületben lévő akadályok okozzák, mint például a falak, térelválasztók, bútorok.

A [7] irodalomban olvashatunk ezekről a kihívásokról és az azokra adott megoldásokról. A cikk írója egy rendszertervet is felvázol, amely blokkdiagram a 3.1-ábrán látható. Ebben a rendszerben fix pozíciójú eszközök figyelik egy mobil eszköz által kiadott jeleket (pl. rádiófrekvenciás jel). A rádiófrekvenciás jel érzékelése után a szenzorok az adott mérési elv alapján távolságot határoznak meg. Az érzékelt jel természetesen lehet a már említett infravörös vagy hangfrekvenciás tartományba eső szignál is. A mérési elv alapja lehet az időmérés (Time of Arrival: TOA), beérkezési szögmérés (Angle of Arrival: AOA), vagy a vett jel erősségének megváltozása (Received Signal Stergth: RSS). Ezeket az elveket később részletesen be fogom mutatni a szakdolgozatomban. Miután a szenzorhálózat meghatározta a távolságokat, pozícióbecslő algoritmusok segítségével rekonstruálható az elrendezés. Attól függően, hogy milyen koordinátarendszert alkalmazunk, a rendszer meghatározza a lokális vagy globális koordinátákat, így megkapjuk az adó vagy vevőállomás becsült helyzetét.



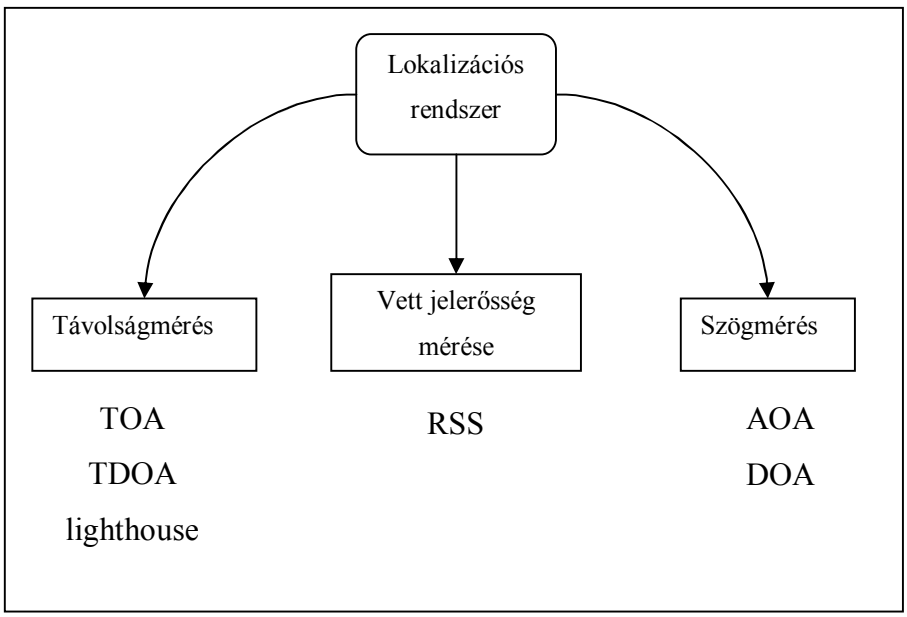
3.1. ábra. A helyzetmeghatározó rendszer blokkdiagramja [7]

Manapság egyre inkább elterjed a WLAN (Wireless Local Area Networks). Ezek a vezeték nélküli hálózatok, általában egyéni felhasználók, vállatok építik ki otthonaikban vagy munkahelyeiken. Mivel ennek a rendszernek a struktúrája már előre kiépített, és a hálózatra kapcsolódó eszközök rádiófrekvenciás jelek segítségével kommunikálnak a hálózattal, ebből adódik, hogy alkalmazható helyzetinformációk

meghatározására. Ez jelentős költségcsökkenést jelenthet a lokalizációs szenzorhálózatok nulláról történő kiépítéshez képest, ezért ez a lehetőség a jövőben jelentős sikereket érhet majd el.

3.3.1. Alapvető távolság és helyzetmeghatározó technikák

A vezeték nélküli szenzorhálózatokban a manapság a távolságmérési technikákat három főbb csoportra lehet osztani. Ahogy a 3.2.-es ábra is mutatja, a mérés történhet beérkezési szög mérésével (AOA: Angle Of Arrival), vett jel erősségének meghatározása alapján (RSS: Received Signal Strength), vagy távolságmérésen alapuló módszerrel. A távolságmérésen alapuló módszeren belül további három csoportot mutatok be a következőkben: a jel megérkezésének ideje (TOA: Time Of Arrival), különböző jelek megérkezésének időkülönbsége (TDOA: Time Difference of Arrival) alapján mért távolság, illetve a lighthouse (világítótorony) módszerrel történő távolságmérés. Természetesen a helymeghatározáshoz a fentebb említett módszerek kombinációja is felhasználható.



3.2. ábra Lokalizációs technikák osztályozása

Távolságmérésen alapuló technikák

A távolságmérésen alapuló technikák keretében három alapvető eljárást mutatok be: TOA, TDOA és lighthouse (világítótorony) módszerek.

A felsorolt módszerek közül a két legelterjedtebb az egymással szoros kapcsolatban álló TOA és TDOA módszerek. Ezek a mérési technikák különböző, az egyes egységek által kiadott jelek terjedési idejének mérésén alapulnak. A TOA és TDOA technikák között az eltérés annyi, hogy az előbbi módszer estén a rendszer egy bizonyos jel kiadásával valamint annak terjedési idejéből és sebességéből számítja a távolságokat. A TDOA-t használó szenzorhálózatokban két szignál egyidejű kiadása esetén, ha a két jel eltérő terjedési tulajdonságokkal rendelkezik, akkor a vevő által mért beérkezési időkülönbségekből is számíthatóak távolsági adatok.

Az érkezési idők mérése esetén nagy hangsúlyt kell fektetni az időzítés pontos beállítására, valamint kis mérési távolságok esetén nagyon lecsökkenhet a mért idő felbontása. Ennek következtében elengedhetetlen nagypontosságú órák használata. A TOA módszer megkívánja az adó és vevő órájának pontos szinkronizálását, mivel a jelkibocsátás és a jel megérkezésének időkülönbsége ezen két óra alapján történik. A bonyolult szinkronizációs algoritmusok alkalmazásának kiküszöbölése céljából gyakran alkalmaznak úgynevezett round-trip, vagy más szóval oda-vissza terjedésű jeleket a szenzorhálózatokban. Ennél a módszernél viszont az a hátrány, hogy számolni kell a szenzorok késleltetésével, ami a jelfeldolgozásból és a „vissza” jel elküldéséből adódik. Megfelelő módszerekkel azonban ez a hiba is kiküszöbölhető.

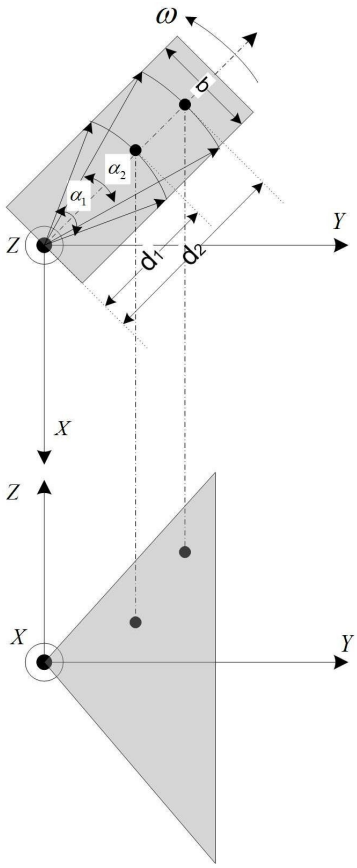
A TOA esetén felmerülő szinkronizációs és késleltetési problémák csökkenthetőek a TDOA elv alkalmazásával. Ebben az esetben egy adó eszköz két különböző terjedési sebességű jelet sugároz ki egy időben, majd egy vevő eszköz ezen jelek érkezési idejeinek különbségét méri. Mivel egyazon eszközön történik az idő mérése, így időszinkronizáció nem szükséges az eszközök között. A két egyszerre kiadott jel lehet például egy rádiófrekvenciás és egy akusztikus, azaz hangfrekvenciás szignál. Mivel az elektromágneses jelek terjedési sebessége több nagyságrenddel nagyobb, mint a hang sebessége, ebből adódóan a rádiós jel terjedési ideje elhanyagolható. Mivel az általam megvalósított rendszer ezt a módszert használja, így ezt az elvet részletesebben majd az 5.1. alfejezetben fogom bemutatni.

A TOA és TDOA módszereken kívül érdemes megemlíteni a „világítótorony” megközelítést (Lighthouse approach) [8] a távolság meghatározásához.

A *világítótorony módszer* alapja a fényérzékelés. Egy optikai érzékelővel párhuzamosan helyezkedik el egy ismeretlen szögsebességgel forgó fénysugár, ahogy a 3.3. ábrán is látható. A vevő a XY síkban helyezkedik el d_1 távolságra az adótól, a fénysugár pedig a Z tengelykörül forog. Az ismeretlen ω szögsebesség kiszámítható az alapján, hogy az optikai vevő milyen időközönként érzékeli a fényt. A d_1 távolság levezethető a t_1 időtartam ismeretében, ami azt adja meg, mennyi ideig tartózkodik a szenzor a fénysugárban [8].

$$d_1 \approx \frac{b}{2\sin(\alpha_1/2)} = \frac{b}{2\sin(\omega t_1/2)}. \tag{3.1}$$

A legnagyobb előnye ennek a módszernek az, hogy a vevő lehet nagyon kicsi méretű, ez alátámasztja a „smart dust” elképzelést. Ugyanakkor az adó viszont lehet nagy. Bővebben erről a módszerről a [8] cikkben lehet olvasni.

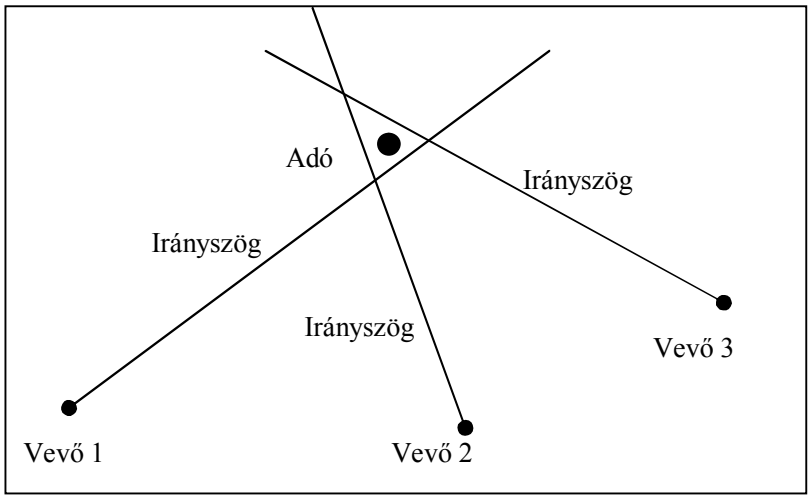


3.3. ábra. Világító torony módszer elmélete [5]

Szögmérésen alapuló technikák

A beérkezési szög (AOA) vagy a beérkezési irány (DOA: Direction Of Arrival) segítségével is meghatározható a hálózat topológiája. Ebben az esetben több, legalább

három, de ajánlott inkább négy referenciapont felvétele. Ezek alapján a beérkezési szög mérése segítségével szerkesztett egyenesek metszéspontjai megadják az adó vélt pozícióját. Ennek illusztrációja látható a 3.4.-es ábrán.



3.4. ábra. AOA alapján történő lokalizáció

Vett jel erősségén alapuló helymeghatározási módszer

Ez a módszer a vett jel erősségét (RSS) veszi a lokalizációs mérés alapjául. A módszer hátránya az, hogy részletes ismeret szükséges a tartomány pontjainak jelerősségéről. Mivel beltéri kommunikáció esetén a többutas terjedés miatt egy adott pontban mérhető elektromágneses térerősség nem feltétlenül monoton függvénye a távolságnak, így a térerősség nem származtatható egyszerű formulák segítségével a távolságból, hanem méréssel kell meghatározni a közöttük lévő összefüggést. Ezeket a mérési eredményeket egy adatbázisban kell eltárolni. A mért térerősség-értékek az adott hely úgynevezett fingerprint értékei. A feltérképezésnek köszönhetően nagyon időigényes a rendszer telepítése és kalibrálása. Az eljárás előnye viszont, hogy nem igényeli a legalább három bázisállomás meglétét, mivel a szenzor által mért jelerősség értéket összehasonlítja az adatbázisban szereplő értékekkel és a hozzá legközelebb eső alapján kiválasztja a vevő koordinátáját.

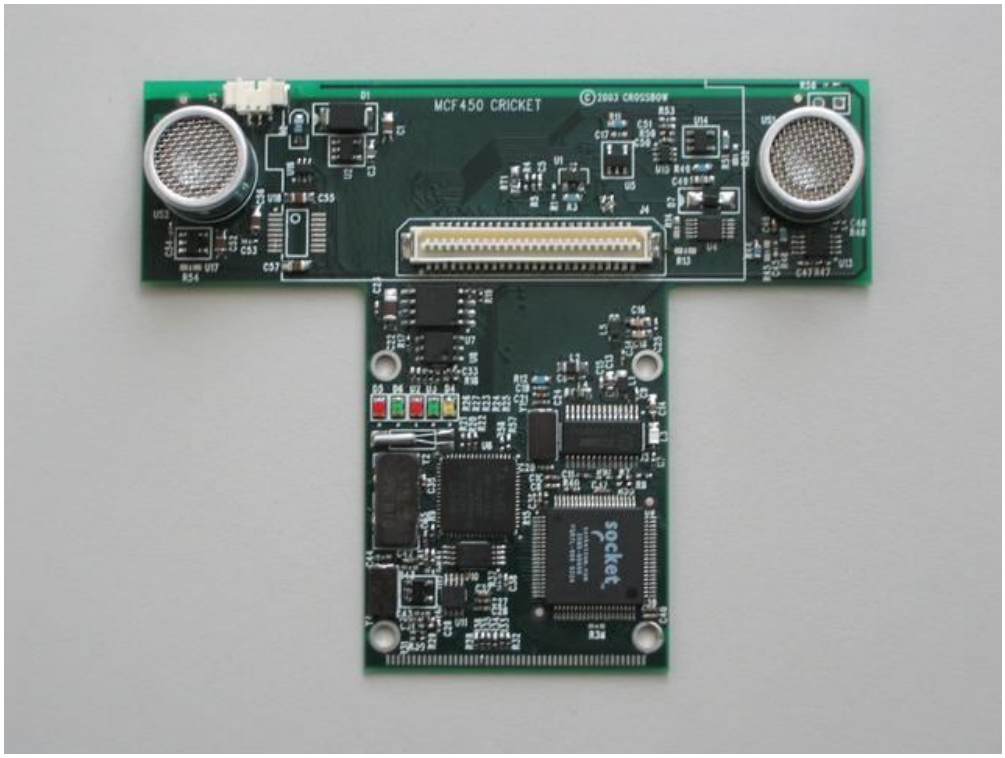
3.4. Napjainkban használt beltéri lokalizációs rendszerek

A legtöbb manapság gyakorlatban is alkalmazott beltéri helymeghatározó rendszer a fentebb tárgyalt módszerek segítségével működik. Ezekről a technikákról olvashatunk a [9],[10],[11] irodalmakban.

Cricket rendszer

A Cricket rendszerben [9] távolságmérés segítségével történik a lokalizáció. A hálózat nem tartalmaz központi szervert, ezért minden számítás az egyes állomásokon történik. Ennek előnye, hogy könnyen alakítható a hálózat. A távolságmérés TDOA, tehát a beérkezett jelek detektálásának időkülönbségéből származtatható. A Cricket rendszer esetében minden helységben található egy rádiófrekvenciás adó, amely egy egyedi azonosítójú jelet sugároz, ezzel egyidőben pedig kiad egy ultrahang tartományba eső szignált is. A vevők érzékelik mindkét jelet, és a beérkezési időkülönbségből távolságot becsülnek. A rádiós jel alapján történik az egységek szinkronizációja, így a rádióüzenet megérkezése előtt bejövő ultrahangos jelet nem veszi figyelembe, ezzel is javítva a zavarérzékenységet.

A rendszer hátránya, hogy a rádiófrekvenciájú jelek interferálhatnak, ezért például két szomszédos szoba között felcserélődhetnek a szinkronizáló jelek. A helymeghatározás pontossága is kisebb, de a rendszer előnye az, hogy jó adatvédelmi tulajdonságokkal rendelkezik, könnyen átalakítható és viszonylag alacsonyak a költségei. A 3.5. ábrán egy Cricket rendszer vevőállomása látható.



3.5. *ábra. Cricket rendszerben alkalmazott szenzor*

SpotOn rendszer

A SpotOn [12] rendszer egy ad-hoc lokalizációs rendszer. Az ad-hoc lokalizációs rendszereknél mindegyik szenzor, eszköz ugyanazzal a tulajdonságokkal rendelkezik. Nincs kitüntetett eleme a rendszernek, nincs bázis elem, mindegyik érzékelő egyforma programmal működik.

Ennél az alkalmazásnál a jelerősség csillapodásának mértékéből számítják a távolsági adatokat. Ezekben a rendszerekben az elemek egymással együttműködve üzemelnek, megosztják a mért adatokat, így próbálják csökkenteni a mérési hibákat. A szenzorok egymástól mért távolságokat mérnek, nem egy rögzített bázistól, mint más mérési technikák esetén. Az eszközök a mérési eredmények közötti korrelációt használják ki a pontosság javításához.

Ennek a nem fixen telepített rendszernek az előnye, hogy könnyebben átalakítható a szenzorhálózat topológiája, és telepítése nem vesz hosszú időt igénybe. A költségek is viszonylag alacsonyak ennél a helymeghatározó rendszernél.

Active Badge

Az egyik legkorábban kifejlesztett beltéri helymeghatározó rendszer, az *Olivetti* és *AT&T* újítása volt. Infravörös (IR) jeladókat használó cellás (proximity) rendszer. A

jeladó periodikusan, legtöbbször tíz másodpercenként egy egyedi azonosítót ID-t sugároz, a jelet pedig egy infravörös tartományú jelekre érzékeny szenzor veszi, és egy központi egységhez továbbítja a vett szignált. A rendszert legtöbbször úgy építik ki, hogy szobánként alkalmaznak egy vevő egységet, így csak szimbolikus adatokat szolgáltat, például, hogy a szobában van-e az akin vagy amin az adó megtalálható. Tehát a rendszer hatótávolsága korlátozott, és az infravörös jelekből adódó hibajenségek is befolyásolják az eredményességet, mint például a napfénnyel, vagy a fluoreszkáló fénnyel való interferencia

3.6. ábra. Active Badge jeladó és bázisállomás

Active Bat

Az *AT&T* 1999-ben bemutatott újítása volt, az infravörös tartományba eső jelek helyett ultrahangot alkalmaztak. Ez a rendszer már jóval pontosabb, a mérések 95%-ban 9 cm-es körön belül van a pozícionálás pontossága.

Az IR jeladók helyett alkalmazott ultrahangos jeladókat a felhasználók viselik magukon, a vevők pedig egy úgynevezett grid (rács) rendszert alkotva vannak elhelyezve a plafonon. A távolságmérés itt is az ultrahang terjedési idejének és a sebességének a szorzatából adódik.

Első lépésben egy központi kontroller kiad egy rádiós üzenetet, amire a megfelelő Bat jeladó egy ultrahangos jellel válaszol, valamint ezzel egyidőben a kontroller a plafonra szerelt egységeknek is küld egy *reset* jelet, amelyekkel vezetékes kapcsolatban van. A kiadott ultrahangos szignál érzékelése után az érzékelők

visszaküldik a mért adatokból számított távolságot a központnak, ahol megtörténik a lokalizáció.

A rendszer hátránya az, hogy a plafonra meglehetősen sok érzékelő felszerelése szükséges, és a pontos elhelyezés is fontos a rendszer megfelelő működéséhez. Ennek köszönhetően a telepítés meglehetősen nehéz, és a sok szenzor miatt az ár is magasabb.

3.7. ábra. Active Bat jeladó

Radar

A RADAR rendszert a Microsoft fejlesztette ki. Ez egy vett jel erőssége alapján működő rendszer. A manapság egyre jobban elterjedt WLAN hálózat segítségével határozza meg a pozíciót. Az IEEE 802.11 szabvány alapján működő WLAN-alapú rendszer. Minden WLAN bázisállomásban mérik a vett jel erősségét és a jel-zaj viszonyt, ezen értékek alapján az épületen belül meghatározható egy síkbeli elrendezés. A legközelebbi szomszéd algoritmus alkalmazásával 38%-os pontosság érhető el 2 m-en belül.

A rendszer előnye, hogy a már legtöbb helyen kiépített WLAN hálózatokra épül, így olcsó a kiépítése. Hátránya viszont az, hogy IEEE 802.11 szabványt alkalmazó eszköz kell, a legtöbb szenzor viszont ezt nem támogatja.

3.5. A helymeghatározó rendszerek általános tulajdonságai

A helymeghatározó rendszerek egyik alapvető tulajdonsága az, hogy milyen lokalizációs információval szolgál. Ez lehet fizikai információ, ebben az esetben pontos koordinátákat ad meg a rendszer a kimenetén, vagy lehet szimbolikus, ahol nem

határozunk meg konkrét távolságadatot, hanem valamilyen kvalitatív információt szolgáltat a rendszer a pozícióról, például egy adott térrészben vagy azon kívül található egy szenzor, vagy éppen távolodik/közeledik, stb.

A pozíció megadását is többféleképpen végezhetjük. Az abszolút pozíciós rendszert alkalmazó hálózatokban közös referencia grid-et használnak, míg a relatív pozíció esetében akár objektumonként más-más lehet a referencia. Az abszolút pozíció könnyen átalakítható egy másik ponthoz képest relatív információvá, és ez visszafelé is működik. Kivételt képez az az eset, amikor a referenciapont mozog, így pozíciója nem ismert minden időpontban (például mobiltelefon). A lokalizációs algoritmusok a számítási műveletek elvégzésének szempontjából lehetnek centralizáltak vagy elosztottak. A centralizált esetben a szenzoroktól begyűjtött globális adatokat egy központi, úgynevezett bázis egységnek továbbítják, és ott történik meg a pozíció becslése. Ilyen például az Active Bat rendszer is. Az elosztott rendszerek esetében minden node ugyanolyan műveletek végrehajtására képes, és a szomszéd eszközökkel való kommunikáció után saját maguk oldják meg a lokalizációs feladatot. Ilyenek például az ad-hoc telepítésű SpotOn rendszerek. A lokalizáció történhet referenciapontok segítségével, vagy távolságmérés alapján.

A beltéri helymeghatározó rendszerek további fontos paramétere annak költsége. A költségek alatt a telepítési, kiépítési, működtetési, ellenőrzési és karbantartási kiadásokat értjük. Ezen felül fontosak az időparaméterek is; itt a rendszer telepítési ideje és a pozícióbecslés ideje a két legfontosabb tényező. Továbbá előnyös, ha a rendszer mobilis. A rendszerben a bázis és/vagy szenzorok, a követendő objektum mozoghat. Napjainkban a biztonság és az adatvédelem is elengedhetetlen. A jogosulatlan hozzáférések megakadályozása, a felhasználók anonimitásának biztosítása is elengedhetetlen feladat. Az I. Táblázatban ezek a tulajdonságok vannak összefoglalva [15].

I. Táblázat. *Helymeghatározó rendszerek általános tulajdonságai*

Lokalizációs információ	Fizikai Szimbolikus
Abszolút vagy relatív pozíció	Abszolút Relatív
Lokalizációs technikák	Centralizált Elosztott
Lokalizációs megoldás	Referenciapontok Távolságmérés
Költségek	Telepítési, kiépítési, működtetési, karbantartási
Mobilitás	Mozoghat: bázis és/vagy szenzor, az események, követendő objektum
Idő	Telepítési, pozícióbecslés ideje
Biztonság	Jogosulatlan hozzáférések kivédése Felhasználók anonimitása

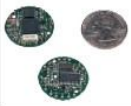
4. Felhasznált hardver és szoftvereszközök

A lokalizációs rendszer megvalósítására rendelkezésre álló platformot a Crossbow cég által árusított Berkeley micaz mote-ok szolgáltatták. Ezen eszközök tipikus szenzorhálózati alkalmazásokhoz készített, mikrokontroller alapú, rádiós kommunikációval és szenzoregységekkel rendelkező eszközök. Mivel a feladat megoldásához szükséges ezen eszközök bizonyos szintű ismerete, így a fejezet következő részében bemutatom a mote-ok hardverfelépítését és szoftveres támogatottságát. A szenzorhálózat által mért adatok feldolgozása általában valamilyen nagyobb teljesítményű eszközön, tipikusan PC-n történik, így röviden bemutatom a szenzorhálózattal való kapcsolat kialakítását lehetővé tevő PC-s eszközöket is.

4.1. Szenzorhálózat hardverelemei

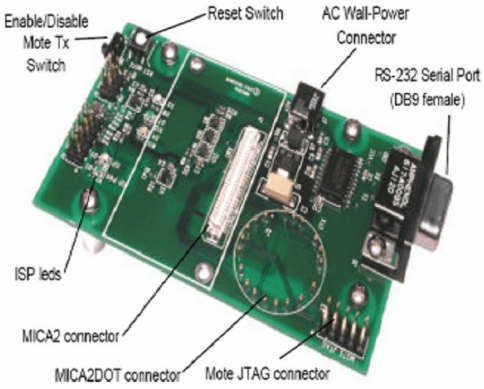
A Crossbow cég több mote családot is gyárt. Elterjed típusok a XM2110, M2110, MPR2400, MPR400. Ezek közös tulajdonsága, hogy mindegyik az Atmel ATmega 128-as mikrovezérlőre épül, és rendelkeznek rádiós adóvevővel. A rádiós IC-k lehetnek típustól függően Atmel RF230, TI CC2420, TI CC1000. A flash memória 512 kB-os és a RAM lehet 4 illetve 8 kB.

Én a Berkley mote család MPR2400-as alapkártyáját használtam fel a szenzorhálózatban. A 4.1.-es ábrán a Berkley mote család tagjai láthatóak.

Photo	Crossbow Part ID	Commonly Used Name	Frequency Range	Processor	Radio Transceiver	Nonvolatile Memory
	MPR300 (discontinued) MPR310 (discontinued)	MICA (sometimes refered to as MICA1)	902 to 928 MHz 433.1 to 434.8 MHz	Atmel ATMega128L	RFM TR1000	Atmel AT45DB041B (512 kB)
	MPR400 MPR410 MPR420	MICA2	868 to 870; 902 to 928 MHz 433.1 to 434.8 MHz 313.9 to 316.1 MHz	Atmel ATMega128L	Chipcon CC1000	Atmel AT45DB041B (512 kB)
	MPR500 MPR510 MPR520	MICA2DOT	868 to 870; 902 to 928 MHz 433.1 to 434.8 MHz 313.9 to 316.1 MHz	Atmel ATMega128L	Chipcon CC1000	Atmel AT45DB041B (512 kB)
	MPR2400	MICAz	2400 to 2483.5 MHz	Atmel ATMega128L	Chipcon CC2420 (802.15.4)	Atmel AT45DB041B (512 kB)
	MCS400	Cricket	433.1 to 434.8 MHz	Atmel ATMega128L	Chipcon CC1000	Atmel AT45DB041B (512 kB)

4.1. ábra. Berkley mote család

A feladatomhoz felhasznált Berkley micaz mote-ok moduláris felépítésű eszközök. A mote tartalmaz egy alapkártyát (MPR2400), amelyre különböző kiegészítő kártyák (szenzor- és interface kártya) csatlakoztathatóak [13].

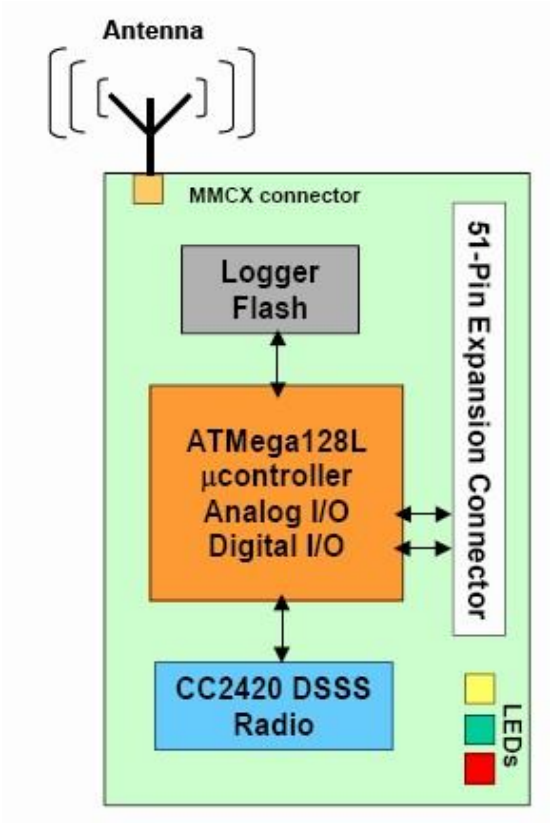


4.2. ábra. Alapkártya és szenzorkártya (bal), programozó kártya (jobb)

A 4.2.. ábra bal oldalán látható a mote alapkártyája és a rácsatlakoztatott szenzorkártya. Jobboldalon egy interface kártya látható, mely segítségével programozhatjuk az eszközöket, és lehetővé teszi a PC-vel a kommunikációs kapcsolat kialakítását soros port segítségével.

Alapkártya

Az alapkártya az az egység, amelyen található egy, a feladat szempontjából fontos rádiós IC valamint egy mikrovezérlő. Az alapkártya főbb részei a 4.3. ábrán láthatóak.



4.3. ábra. A micaz mote alapstruktúrája

Csatlakoztatási lehetőségek: Az alapkártyán található 51-Pin-es csatalakozó segítségével lehet az alapkártyához (MPR2400) csatlakoztatni a szenzorkártyát és programozás esetén az interface kártyát. Ezen a konnektoron találhatóak a megszakításkezelő kivezetései, az AD átalakító csatlakozói, a soros kommunikációs lehetőségek és általános I/O és programozható bemenetek. A tápellátás két darab AA elem segítségével vagy az interface kártyán keresztül tápegységből történhet.

Az alapkártyán találhatóak még egyéb kiegészítők is. Az egyik legfontosabb ilyen kiegészítők a LED-ek, egy piros, zöld és sárga LED, melyek debugolási célra használhatóak. További kiegészítők még egy flash memória és egy azonosító memória is [14].

Mikrokontroller: A micaz mote alapkártyáján egy ATmega128 típusú mikrokontroller található, amely egy egyszerű feladatok végrehajtására képes 8 bites, RISC architektúrájú mikrovezérlő. A legtöbb Crossbow által gyártott mote-on ez a mikrokontroller található meg.

A mikrokontroller órajelét programozáskor lehet kiválasztani öt lehetőség közül. A mikrokontroller tartalmaz reset áramkört is, aminek fontos szerepe van a determinisztikus indulás miatt. A biztonságos működést egy watchdog timer biztosítja. A watchdog timer feladata az, hogy reseteli a mikrokontrollert, ha nem jelzi a program elég gyakorisággal a működését, tehát nincs visszajelzés a működésről (például végtelen ciklusba került a program).

További hasznos funkció még az energia menedzsment, aminek a segítségével tilthatóak a perifériák, így növeli a vezérlőt működtető elemek élettartamát. Az üzemmód meghívható a *SLEEP utasítás + engedélyez + módválasztás* segítségével. Hatféle üzemmódot támogat:

- *Idle:* CPU áll (clkCPU, clkFLASH)=0, perifériák, megszakítások működnek
- *ADC Noise Reduction:* (clkI/O, clkCPU, clkFLASH)=0
- *Power-down*: aszinkron modulok működnek, amihez nem kell belső órajel
- *Power-save*: hasonló az előzőhöz, kivétel Timer0
- *Standby*
- *Extended Standby*

Az ATmega128 Háromféle belső memóriával rendelkezik. Egy 4 kB-os SRAM, ami bővíthető, egy 128 kB-os program flash, amiben a programkód tárolódik és 4 kB-os EEPROM a hosszabb tárolásra szánt adatoknak.
A vezérlő tulajdonságai röviden összefoglalva:

- 8-bites ALU
- Az általunk használt mote órajelfrekvenciája 7.3728 MHz
- 128 kbyte on-chip flash
- 4 kbyte on-chip SRAM
- 4 kbyte on-chip EEPROM
- Szinkron és aszinkron soros interface

- 4 db számláló/időzítő amelyek PWM módban is használhatóak
- Analóg komparátor
- 8 csatornás multiplexelt 10 bites AD-átalakító
- Többféle energiatakarékos üzemmód

Rádiós IC: Az alapkártya tartalmaz egy CC2420 típusú integrált adó-vevő egységet, amelyek segítségével a mote-ok között kialakulhat egy vezetéknélküli kapcsolat. Ezen keresztül az eszközök egymásnak adatokat tudnak továbbítani, valamint fogadni. Ez egy ZigBee szabványnak megfelelő rádió, amely 2.4 GHz-es ISM (Industrial, Scientific and Medical - ipari, tudományos és orvosi célokra szabadon felhasználható) rádiófrekvenciás sávban működik, az adatátviteli sebessége 250 kbps. Ez a rádió az IEEE 802.15.4 szabvány szerint definiált fizikai és MAC réteget tartalmazó integrált áramkör. Ezt a szabványt kifejezetten lokális, kisméretű helyi hálózatokhoz fejlesztették ki. Támogatja a nagy csomóponti számmal rendelkező hálózatokat, ezért nagy címtartománnyal rendelkezik. Fontos szempont még a kis fogyasztás, amelyet a CC2420 támogat célirányos moduljaival, így egyes műveletek nem veszik igénybe az ATmega128-as mikrovezérlőt, ezért nem foglalják annak erőforrásait.

Az IC tulajdonságai :

- 2400 - 2483.5 MHz rádiófrekvenciás adó-vevő:
  – Direct Sequence Spread Spectrum (DSSS) adó-vevő
  – 250 kbps adatátviteli sebesség
  – O-QPSK (ofszet kvadratúra-fázis) moduláció
  – alacsony áramfogyasztás (Rx: 8.8 mA, TX: 17.4 mA)
  – magas érzékenységi szint: -95 dBm
  – alacsony tápfeszültség (2.1 V - 3.6 V)
  – 4 vezetékes SPI interfész
- FIFO az adás és vétel számára:
  – 128 byte küldött adat FIFO
  – 128 byte vett adat FIFO
- kevés külső komponens:
  – referencia oszcillátor és minimális számú passzív alkatrész
- 802.15.4 MAC hardware támogatás:

– automatikus preamble generátor
– szinkronizáló szó beszúrása/detektálása
– CRC-16 számítás
– jelerősség érzékelés – digitális RSSI érték
– szabad csatorna figyelése

802.15.4 szabvány szerinti MAC hardveres biztonság:

CTR titkosítás - visszafejtés
CBC-MAC hitelesítés
CCM titkosítás - visszafejtés
AES titkosítás

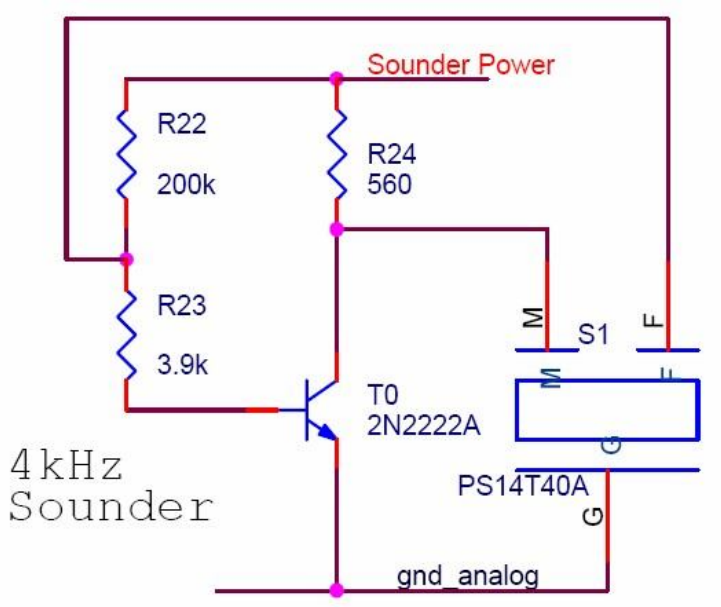
Flash memória: Ez egy AT45DB041B típusú memória, mely 512 kByte tárolására alkalmas hosszú távon, így a mérési adatok tárolhatóak, ha nem szükséges az azonnali továbbításuk. Soros elérési lehetőséggel rendelkezik, 13 MHz-es órajellel működik. Fogyasztása 4 mA / 2 uA (aktív és standby állapotban). Hierarchikus felépítésű, blokkos adatkezelést alkalmaz. Az olvasás történhet elemenként, vagy folytonos olvasás formájában. Az írás viszont csak blokkos formában lehetséges.

A szenzorkártya főbb részei

A szenzorkártya egy MTS310-es típusú kártya, amelyen találhatóak kifejezetten akusztikus lokalizációhoz szükséges egységek: a hang kiadására szolgáló piezo buzzer és a hang érzékelésére szolgáló mikrofon. A szenzorkártya az alapkártyába az 51-pin-es csatlakozó segítségével kapcsolódik. A hang kiadásán és érzékelésén kívül képes a fényerősség és a hőmérséklet mérésére is. A fényerősséget egy fotoszenzor, míg a hőmérsékletet egy termisztor érzékeli. Található még a szenzorkártyán egy gyorsulásmérő (ADXL202e), aminek a szerepe majd a továbbfejlesztés során lehet jelentős, például a lokalizációs mérés indítása esetén, ha akkor szeretnénk mérni, amennyiben azt érzékeljük, hogy valamelyik szenzor megmozdult (gyorsulása megnő). A szenzorkártyán található mágneses szenzort nem alkalmaztam a fejlesztés során.

Akusztikus jel kiadása: A lokalizációhoz szükséges akusztikus jel előállítása egy buzzer segítségével történik, amely szinuszos hang kiadására képes, piezoelektromos elven működő eszköz. A szinuszos jel frekvenciája körülbelül 4 kHz. Ez egy keskenysávú, jól

megkülönböztethető hang. A buzzer a mikrokontroller egyik digitális kimenete segítségével kapcsolható be és ki.



4.4. ábra. Piezo Buzzer

Akusztikus jel érzékelése: A lokalizáció során használt akusztikus jel vételére alkalmas mikrofon valamint az ehhez tartozó jelkondicionáló egységek is megtalálhatóak a szenzorkártyán. A buzzer által kiadott jel érzékelése a mikrofonegység kimenetére csatlakoztatott PLL áramkör segítségével történik. A PLL sávközépi frekvenciája a buzzer frekvenciájára van hangolva. A PLL behúzását, vagyis a buzzer által kiadott hang érzékelését a PLL áramkör egy digitális kimeneten jelzi, tehát ez a jelzés használható fel a lokalizáció során az akusztikus jel érkezésének észlelésére. A feladat megoldása során természetesen figyelembe kell venni a PLL beállási idejét, mely ofszet jellegű hibát okozhat. A kártyán lehetőség van még a mikrofonhoz tartozó erősítő erősítési tényezőjének beállítására, amely a kártyán található, I^2C porton keresztül programozható ellenállás segítségével valósítható meg.

Az interface kártya

Az általunk használt programozói kártya típusa MIB510 (lásd 4.2. ábra), melynek segítségével tudjuk csatlakoztatni a mote-kat a PC-hez. A számítógépen a fordító előállítja a forráskódból a gépi kódú forráskódot, mely a megfelelő szoftver segítségével a programozói kártyához csatlakoztatott mote programmemóriájába tölthető. A betöltött

program a következő újraprogramozásig a mote memóriájában marad. A programozás során meg kell adni a mote hálózati azonosítóját. A kártyára lehetséges tápfeszültség csatlakoztatása, amely a mote-nak is biztosítja a működéshez szükséges feszültséget, valamint a mote tápfeszültségéről is üzemelhet a programozói kártya. A PC-hez RS232-es porton keresztül csatlakozik az egység, így bármely olyan eszközhöz csatlakoztatható, ami rendelkezik RS232-es interface-szel. Az interface kártya segítségével könnyen kialakítható egy olyan bázisállomás, amelyik képes a hálózat adatainak a feldolgozó egységhez való továbbítására.

4.2. Szenzorhálózat szoftverelemei

A Berkeley mote-ok programozása TinyOS operációs rendszer segítségével történt, a programozási nyelv pedig az ehhez kapcsolódó NesC nyelv. A NesC nyelven megírt kódból az előfordító egy C nyelvű kódot állít elő, majd a fordító egy mikrokontrollerre tölthető kódot generál [16].

A TinyOS operációs rendszer kifejezetten a szenzorhálózatokhoz kifejlesztett beágyazott operációs rendszer. Ezen felül támogatja a vezetéknélküli technikát, nyílt kódú (open source) és ingyenes, bárki által felhasználható. A rendszert NesC nyelven írták meg, így ez jól illeszkedik a fejlesztői környezethez, amelyben ugyanezen a nyelven fejleszthetünk.

A NesC nyelv szintaktikája hasonlít a hagyományos C nyelvhez. Ez egy komponens alapú programnyelv, tehát a program komponensekből épül fel, ez a szemlélet segíti a programtervezést és az elkészült program tesztelését. Így az egymástól nagymértékben függetleníthető komponensek csökkentik a hibalehetőségeket is. Kétféle komponensből áll össze a rendszer: egyik a modul a másik pedig a konfiguráció elnevezésű komponens. A komponens szolgáltatást nyújthat (*provides*) és használhat (*uses*). Minden komponensnek meg kell hívnia az alkomponensei megfelelő függvényeit a sajátjában.

A komponensek összekötése interfészeken keresztül történik, amiket a modulok implementálnak. Az interface-ek kétirányúak és a providert valamint a usert kapcsolják össze. A provider parancsokat (*commands*) definiál, a user pedig eseményeket (*events*) generál. A parancsok segítségével hajtatunk végre bizonyos feladatokat a komponensekkel, míg az események visszajelzésként szolgálnak az elvégzett műveletek

után, illetve külső események bekövetkezését jelzik. Az interfészek paraméterezhetőek, mely azt jelenti, hogy egy adott komponenshez több ugyanolyan típusú, de egy szám segítségével azonosítható interface kapcsolódik. Ennek akkor van szerepe, amikor egy adott komponenst a felhasználói program több folyamata is használja. Például a rádiós egységhez külön interface-en keresztül kapcsolódhat egy, a lokalizációval kapcsolatos információkat továbbító, és egy mérési eredményeket továbbító task, és a két task lekezelése a rádiós modulban történik a kölcsönös kizárás megfelelő lekezelésével. Ennek segítségével a rádió több folyamat számára egyszerűen elérhető anélkül, hogy a felhasználói programban gondoskodnunk kellene a közös erőforrás megfelelő kezeléséről.

A NesC által támogatott platformok: mica, mica2, mica2dot, mica128 a Crossbow termékek közül, valamint támogatja még az avrmote-ot és PC-t is.

A TinyOS ütemezése az alábbi entitások alapján történik:

- **hadver megszakítás:** Ez a legmagasabb szintű megszakítás, bármilyen kódrészletet megszakíthat, ha nincs bekapcsolva a megszakítások tiltása.
- **aszinkron command és aszinkron event:** olyan parancsok illetve események, amelyeket megszakításrutinokban generálunk.
- **taszkok:** a taszkok olyan műveletek, amelyeket bármely programrészletből bármikor elindíthatunk. A taszkok közötti versenyhelyzet elkerülése érdekében a taszkok egy FIFO-ba kerülnek, ennek megfelelő sorrendben hajtódnak végre. Egy taszk futása csak akkor indul el, ha már az előző taszk végrehajtódott, befejezte a futást.
- **szinkron command és szinkron event:** ezek olyan parancsok illetve események, amelyek a taszkokban generálódnak.

A kölcsönös kizárás a TinyOS operációs rendszeren belül kétféle módon oldható meg. Egyik módszer esetén csupán taszkokon keresztül történik az erőforrások kezelése, mivel a taszkok egymást nem tudják megszakítani. Másik lehetőség az atomic mód, mely atomic kulcsszóval aktiválható. Ekkor a megszakítások tiltva vannak, így biztosítva van a kölcsönös kizárás.

A TinyOS honlapjáról letöltött programozói környezetben több alapvető komponens már rendelkezésre áll. Ezek előre elkészített, ingyenesen használható, átdolgozható és tesztelt kódok a mote-ok perifériájának és a rádió kezelésére, így ezek a

kódrészletek felhasználhatóak a fejlesztés során, és ha szükséges, akkor a hiányzó funkciók beépíthetőek a rendszerbe.

4.3. PC-s szoftverkörnyezet

A szenzorhálózat irányítása, valamint a mért értékek feldolgozása számítógépen történik. A bázis mote kommunikál a kifejlesztett PC-s környezettel. A 4.2. alfejezetben olvasható, hogy a szenzorhálózat mote-ja is a számítógépen keresztül programozhatóak. Az MPR2400-as alapkártya egy interfész egység (MIB510) segítségével kommunikál a PC-vel. A kapcsolat soros kommunikáción keresztül valósul meg. A 7.2. alfejezetben részletesebben kifejtem majd az RS232-n történő kommunikációt. Itt a PC-n használt programokról szeretnék írni.

A mote-ok programozása a Cygwin nevű program felületen keresztül történt. A Cygwin egy olyan DLL (dynamic link library), melynek segítségével UNIX környezetet emulálhatunk Windows alatt, így akár Linux, Unix alkalmazásokat portolhatunk Win32 környezetbe. A Cygwin alatt megtalálható az összes fontos Unix környezetben használt fejlesztőeszköz, például a gcc, binutils, gdb, make, és számos egyéb ismert segédprogram.

A soros porton történő adatok fogadása és a mérés indítása a MATLAB programrendszer segítségével valósul meg. A MATLAB egy speciális programrendszer, amely numerikus számítások elvégzésére lett kifejlesztve. A The MathWorks által kifejlesztett programrendszer képes mátrixszámítások elvégzésére, függvények és adatok ábrázolására, algoritmusok implementációjára és felhasználói interfészek kialakítására.

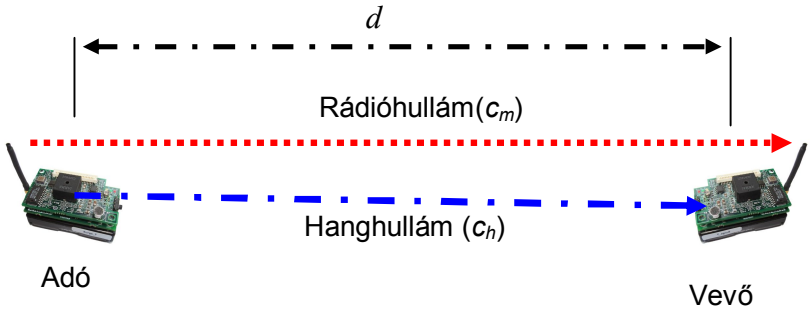
A MATLAB sokoldalú grafikus lehetőségei, valamint az általa nyújtott numerikus számításokat lehetővé tevő könyvtárak miatt ezt a környezetet választottam a PC-n történő adatfeldolgozásra. A PC-n végrehajtható feladatok például az adatok soros porton történő fogadása, adatok előfeldolgozása, megjelenítése és tárolása, kalibráció elvégzése, a mérési hibák kiküszöbölésére és a mért adatok korrigálása.

5. A kiválasztott távolságmérési eljárás alapelve és megvalósítási kérdései

Ebben a fejezetben a megtervezett akusztikus távolságmérést használó szenzorhálózat általános leírása található. Bemutatásra kerülnek a működési diagramok, a használt üzenettípusok.

5.1. Akusztikus távolságmérés és megvalósítás kérdései

Az akusztikus lokalizáció során egyazon időben kiadott rádiófrekvenciás jel és akusztikus jel érzékelésének időkülönbségéből a távolságot egyszerűen meg lehet határozni, mivel a hang és az elektromágneses hullám terjedési sebessége eltérő. Tudjuk. Hogy a fénysebesség körülbelül $3 \cdot 10^8$ m/s, míg a hangé ehhez képest jóval kisebb, körülbelül 340 m/s, de függ a környezeti paraméterektől is (pl. hőmérséklet). A 5.1. ábrán látható egy egyszerű elrendezés a lokalizációs módszer szemléltetéséhez.



5.1. ábra. Akusztikus lokalizáció illusztrációja

A két szenzor között lévő távolság egyszerűen megadható: vagy a hang terjedési sebességének és a terjedés idejének, vagy az elektromágneses jel sebességének és terjedési idejének a szorzataként:

$$d = c_h T_h = c_m T_m \tag{5.1}$$

$$T_h = \frac{d}{c_h}, T_m = \frac{d}{c_m}. \tag{5.2}$$

Ahol a c_h és T_h a hang terjedési sebessége és ideje, a c_m és T_m pedig az elektromágneses jelé. Az időkülönbség kiszámítható a két terjedési idő különbségeként:

$$\Delta T_t = T_h - T_m = \frac{d}{c_h} - \frac{d}{c_m} = d\left(\frac{1}{c_h} - \frac{1}{c_m}\right). \tag{5.3}$$

Mivel a $c_m >> c_h$, tehát a fény terjedési sebessége jóval nagyobb a hangénál, ezért:

$$\Delta T \approx d\frac{1}{c_h} \tag{5.4}$$

A fény terjedési idejének elhanyagolása a következő hibát okozza:

$$h_{\Delta T} = \frac{\Delta T - \Delta T_t}{\Delta T_t} = \frac{d\frac{1}{c_h} - d\frac{1}{c_h} + d\frac{1}{c_m}}{d\frac{1}{c_h} - d\frac{1}{c_m}} = \frac{1}{\frac{c_m}{c_h} - 1} \tag{5.5}$$

$$h_{\Delta T} \approx \frac{c_h}{c_m} = \frac{340\frac{m}{s}}{3\cdot 10^8 \frac{m}{s}} = 1{,}13\cdot 10^{-6} \tag{5.6}$$

A hiba tehát láthatóan elenyészően kicsi, ha elhanyagoljuk a több nagyságrenddel gyorsabban terjedő elektromágneses hullám terjedési idejét.

5.2. Hibaanalízis

A hibaanalízis, vagyis a hibalehetőségek felmérése fontos feladat volt a szenzorhálózat tervezésénél, mert a lokalizáció megvalósításánál ismerni kellett az előfordulható hibalehetőségeket és azok forrásit. Törekedni kell minden determinisztikus viselkedésű hiba detektálására, mivel ezen eltérések korrigálhatóak. Az 5.1.-es alfejezetben beláttuk, hogy az elektromágneses és a hallható hang terjedési idejéből származó hiba elhanyagolható, ezt nem is tekintjük hibalehetőségnek.

A jelgenerálásnál és a mérésnél előforduló időzítési hibák (Δt) az eredményekben távolsághibát okoznak:

$$\Delta t \rightarrow \Delta s = \Delta t \cdot c_h \tag{5.7}$$

Ezek a hibák sokszor nem kerülhetőek el, ezért kalibráció szükséges. Az általam használt kalibrációs eljárást majd egy későbbi pontban részletezem.

További hibalehetőségek származnak a hardver eszközök késleltetéséből és a szoftveres időzítési hibákból.

Hardver késleltetési és időzítési hibái

A szenzorkártyán található piezo buzzer rendelkezik egy úgynevezett feléledési idővel, ami ahhoz szükséges idő, hogy az eszközre kiadott engedélyező jel hatására a buzzer megszólaljon, kiadja a méréshez szükséges akusztikus szignált. Másik elkerülhetetlen hibalehetőség a mikrofonhoz kapcsolódó, adott frekvenciára hangolt (kiadott akusztikus jel frekvenciájára) PLL beállási ideje. A PLL-es detektor beállási ideje függ a hangerősségtől is.

Egy másik hibalehetőség, ami a hardver nem megfelelő működéséből ered az úgynevezett outlierek, vagyis a mért adatsorból feltűnően kilógó mérési eredmények. Ezeket az adatokat nem szabad figyelembe venni, mert a távolságmérés pontosságát nagyban torzíthatják. A mérési eredmények című fejezetben az outlierek eltávolításának az általam használt algoritmusát is be fogom mutatni.

A jel-zaj viszonyt (SNR) is a hardver hibáihoz kell sorolni, mivel a hang kiadását és detektálását is hardver végzi. A külső zajok jel-zaj viszony torzítása ellen nem sokat tehetünk. A mérés pontosságát rontja a zajos jel. A jelszint a mérni kívánt távolság növelésével jelentősen csökken, így pontatlanabbak a távolsági adatok, és nő a mérések szóra is. A csökkenő jelszint érzékeléséhez a mikrofon erősítésének az értékét kell megfelelően beállítani, azonban a túl nagy erősítés hatására nagy zavarérzékenységet tapasztalhatunk, ami már az előbb említett okok miatt rontja a lokalizációs méréseket. Kénytelenek vagyunk kompromisszumot kötni az erősítés pozitív hatása és az erősítésből származó a zavar értéke között, és optimalizálás szükséges. Az optimalizálás különböző erősítési értékek melletti mérés elvégzésével történt figyelve a mérések zajviszonyait.

Szoftver késleltetési és időzítési hibái

A szoftveres hibák adódhatnak a MAC layer késleltetéséből, ami azt jelenti, hogy rádiós üzenet indításának kérési időpontja nem egyezik meg az adás fizikai elindulásával. Ezt okozhatja a hardver feléledési ideje és a CSMA (Carrier Sense Multiple Access) mechanizmus, ami a 802.15.4 szabvány a közeghozzáférés vezérlésére használ. Ez a mechanizmus úgy működik, hogy az adó addig nem kezdheti meg az adást amíg a csatorna foglalt, így a rádiós csatornába „belehallgatva” annak foglaltságát érzékeli. A CSMA mechanizmus akár *msec* nagyságrendnyi késleltetést is okoz. Ennek a problémának a megoldását a program implementációja során végeztem. Az adás

elindítása után a program megvárja a rádiós üzenet elküldéséről a visszaigazolást, és ennek a visszaigazoló jelnek a hatására történik a hang kiadása, így a CSMA mechanizmus késleltetéséből származó hibákat kiküszöböltem. Azzal a feltételezéssel is éltem a tervezésnél, hogy a programfutási időzítési értékek elhanyagolhatóan kicsik a MAC késleltetéséhez képest, és az kevésbé ingadozik, ha a program nem használja ki nagymértékben a processzor erőforrásait.

Az időméréssel kapcsolatban fontos megvizsgálni a mote általunk felhasznált 16 bites időzítőjét, annak érdekében, hogy jó felbontást tudjuk elérni vele. Megfontolva, hogy a mérni kívánt potenciális távolságok maximum néhány méter nagyságúak lehetnek, ebből következően a mért idők körülbelül néhány *msec* vagy 10 *msec* nagyságrendbe fognak esni. Az időzítő órajele a mikrokontroller 7.37 MHz-es órajeléből, annak leosztásával áll elő. A leosztást nem szabad túl kicsire választani, mert ekkor ugyan nagy felbontással tudunk időt mérni, de a számláló túlcsordulhat. A túlcsordulás elkerüléséhez a leosztást minél nagyobbra érdemes választani, ami viszont rontja a felbontást. Ezen megfontolások alapján számítható (természetesen megfelelő tartalékkal) a leosztás, amit 64-es értékben határoztam meg. A programomban ez a következőképpen néz ki:

```
uint8_t  prescaler = 0x3 ;
call Clock3.setScale(prescaler); //timer 64-es osztója
```

A felbontás, így

$$t_{felb} = 64 \cdot \frac{1}{7{,}38 \cdot 10^6 \frac{1}{s}} = 8{,}7\,\mu\text{s} \tag{5.8}$$

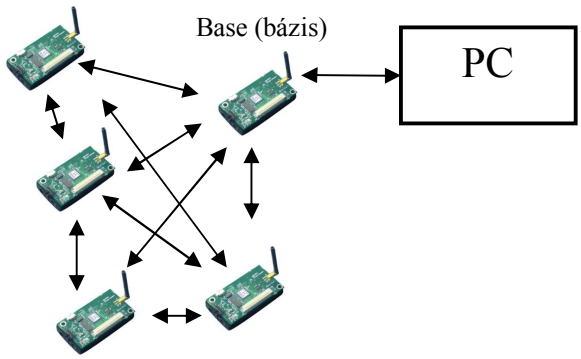
Az így kapott 8.7 μs felbontás megfelelőnek mondható, ez ugyanis körülbelül 8.7 μs·340 m/s = 3 mm-es hibát okoz a mérésben, amely tapasztalataink alapján elhanyagolható a többi hibataghoz képest.

Az időmérésnél fontos szempont volt még az időkülönbségek lekérdezési módja. A rádió és hangdetektor jeleinek folyamatos lekérdezése (polling) ebben az esetben nem célravezető, mert a megkívánt felbontáshoz olyan sűrűnek kellene lenni a lekérdezéseknek, mely már jelentősen leterhelhetné a processzort. A az időméréshez ezen ok miatt megszakítást (IT, interrupt) használtam, mely megfelelő megoldás, mivel

az érzékelni kívánt esemény elég rövid időn belül érvényre jut a felhasználói programban. A TinyOS operációs rendszerben az interruptokhoz kötődő események, úgynevezett eventek (események) formájában jutnak el a felhasználói programhoz. A rádiós üzenet és hangjel megérkezését jelző eventek során rendre elindítom és lekérdezem az erre a célra használt időzítő értékét. Az időzítő indítása nulláról történik, így az általa tartalmazott érték a (5.8) egyenletben adott felbontással szorozva kiadja a rádiós és akusztikus jel érkezése között mért időkülönbséget.

5.3. Általános működési leírás

A már a 2. fejezetben kifejtett szenzorhálózatok általános tulajdonságai az általam tervezett szenzorhálózatra is igazak. Az általam megtervezett rendszer célja, hogy képes legyen a hálózat mote-jainak egymáshoz viszonyított távolságait megmérni, tehát relatív adatokkal szolgál. Azonban, ha a bázis mote-nak, amely PC-hez csatlakozik, meg tudjuk adni a GPS koordinátáit abban az esetben a többi mote koordinátája is megadható a programban való változtatások után. A rendszer centralizált, mivel a távolsági adatok számítása egy központi egységen, ebben az esetben egy számítógépen történik, ami egy úgynevezett bázis mote segítségével csatlakozik a hálózatba. A bázis mote szerepét bármelyik szenzor betöltheti, mivel azonos program fut mindegyik érzékelőn, de csak az viselkedik bázisként, amelyik érzékeli, hogy PC-vel is képes kommunikálni. A lokalizációs megoldás távolságmérésen alapul a jelek beérkezési időkülönbségének felhasználásával, az 5.1. alfejezetben már kifejtett megállapításokat figyelembe véve. Mobilitás szempontjából is előnyös tulajdonságokkal rendelkezik a szenzorhálózat, mivel a PC-hez kapcsolódó szenzor kivételével az összes mote az érzékelési tartományon belül szabadon mozgatható. Képes érzékelni a pozícióváltásból következő távolsági adatok megváltozását. A telepítési idő a többi ad-hoc hálózathoz hasonlóan igen rövid, nem igényli különösebb infrastruktúra kiépítését. Az 5.2. ábrán látható a szenzorhálózat sematikus felépítése.



5.2. ábra. Rendszer sematikus felépítése

Alapvető funkciók és kikötések

A mote-ok a hálózatban betöltött szerepük alapján két-két, egymástól nagyjából független csoportba sorolhatóak. Egyik szempontrendszer alapján egy node lehet *sender* vagy *observer*. Ez a csoportosítás a lokalizáció során betöltött szerepére utal egy mote-nak. Másik csoportosítási szempont, hogy egy mote *bázis* vagy *általános működésű*. Ez a csoportosítás a mote-ok kommunikációban betöltött szerepére utal. A következő felsorolásban az egyes típusok jelentése található részletesen kifejtve:

- Sender: ha a szenzor küldő, tehát sender üzemmódban van, akkor az ő feladata a távolságmérésre szolgáló akusztikus és a rádiófrekvenciás jel egyidejű kiadása. Az üzemmódját a bázis által kiküldött kezdeményező üzenetből olvassa ki. Az általa kiadott, lokalizációs célt szolgáló rádiós üzenetben megtalálható annak az érzékelőnek az azonosítója, amelyikkel a távolságmérést el szeretné végezni.
- Observer: observer üzemmódban a szenzor felkészül a rádiós és akusztikus jel fogadására és beérkezési idők különbségének mérésére. A mért értéket továbbítja a PC felé (vagy közvetve egy bázisállomás segítségével, vagy ha egyben bázisállomás is, akkor UART-on keresztül).
- Base: a bázis mote csak annyiban különbözik egy általános mote-tól, hogy soros porton (UART-on) keresztül kapcsolódik a PC-hez. Egy mote úgy értesül róla, hogy a bázisállomás funkciót el kell látnia, hogy soros porton üzenetet kap a PC-től. A bázisállomás a lokalizációs folyamat során lehet sender, vagy observer módban is, annak ellenére, hogy alapvetően bázis funkciót lát el.
- General: az általános mote-ok azok, amelyek az adott mérési ciklusban nem játszanak szerepet, érzékelik az üzeneteket, de passzívan vesznek részt a mérésben. A bázis kivételével bármely szenzor lehet general.

A mote-ok funkciói mérési ciklusonként változhatnak, köszönhetően annak, hogy mindegyik ugyanazzal a programmal üzemel. A lokalizációért felelős szoftveregység egyetlen általános komponensként van megvalósítva, melyben implementálva van minden üzemmódnak megfelelő viselkedés.

A szenzorhálózatban a távolságmérés kezdeményezése minden esetben PC-ről történik a MATLAB-ban kialakított vezérlő felületen keresztül. A következő lépesben a PC-ről érkező adatok feldolgozása után a base mote kiküld egy kezdeményező rádiós üzenetet, melynek a címzettje a sender mote. Ezen üzenet hatására a sender akusztikus és rádiójeleket generál a lokalizációs mérés számára. A base kezdeményező üzenetében szerepel az observer mote azonosítója is. Ez a fogadó mote érzékeli a sender által küldött szignálokat, és leméri a két jel közötti beérkezési időkülönbséget, majd továbbítja az eredményeket a base felé, ami az időkülönbségeket PC-nek adja tovább. A PC-n történik az adatokból a távolság meghatározása.

Ez az architektúra azért előnyös, mert a PC nagyteljesítményű erőforrás, segítségével egyszerűen megoldható az adatok feldolgozása, bonyolult működési lépések valósíthatóak meg. A központilag feldolgozott adatokat tárolni tudjuk, és akár innen újra elküldhetőek bárki részére.

Egyetlen távolságadat megmérése egymás után több rádió és hangüzenet kiadásával történik, ezzel csökkentve a zaj és hamis mérések hatását. A következőkben egy adott távolságadathoz tartozó mérések sorozatát *mérési ciklusnak* nevezem. Minél több mérés történik egy ciklus alatt, várhatóan annál pontosabb a ciklus végén az egyes mérések együttes feldolgozásából származtatott mérési eredmény.

5.4. Üzenettípusok megadása

A felhasznált rádiós és soros porton továbbított üzenetek fontos szerepet játszanak a szenzorhálózat működésében. A következőben az alkalmazott üzenettípusokat fogom bemutatni.

- LOCALIZATION_FORCE_START_BASE

PC küldi a bázisnak UART-on keresztül. Az üzenetnek kettős szerepe van. Egyrészt ezen üzenet hatására vált egy általános mote bázis üzemmódba. Majd, ha a bázis nem

sender az adott mérési ciklusban, akkor rádión kiküld egy mérést kezdeményező üzenetet; ez a localization_start_request üzenet (lásd következő üzenettípus).

Az üzenet fő tartalma:

localizSenderMoteID: annak a mote-nak az azonosítója, aki a hangot és a rádiós üzenetet kiadja.

localizObserverMoteID: annak a mote-nak az azonosítója, aki a hangot és a rádiós üzenetet fogadja.

extra paraméterek:

deltaSound: ez határozza meg, hogy cikluson belül milyen hosszú legyen két csipogás közötti idő

maxLocNumWithinCycle: azt jelenti, hogy mekkora hosszúságú legyen egy ciklus, hány mérési eredmény alapján akarunk távolságot számítani.

- LOCALIZATION_START_REQUEST

A bázis küldi a rádión, amennyiben nem ő az aktuális sender. Ezzel egy lokalizációs ciklust kezdeményez.

Az üzenet fő tartalma:

localizSenderMoteID: annak a mote-nak az azonosítója, aki a hangot és a rádiós üzenetet kiadja.

localizObserverMoteID: annak a mote-nak az azonosítója, aki a hangot és a rádiós üzenetet fogadja.

deltaSound: ez határozza meg, hogy cikluson belül milyen hosszú legyen két csipogás közötti idő

maxLocNumWithinCycle: jelenti, hogy mekkora hosszúságú legyen egy ciklus, hány mérési eredmény alapján akarunk távolságot számítani.

- LOCALIZATION_SEND_RADIO_BEACON

sender mote küldi az observer mote felé egyidejűleg a hang kiadásával.

Az üzenet fő tartalma:

localizSenderMoteID: annak a mote-nak az azonosítója, aki a hangot és a rádiós üzenetet kiadja.

localizObserverMoteID: annak a mote-nak az azonosítója, aki a hangot és a rádiós üzenetet fogadja.

locCycle: a lokalizációs ciklus azonosítója, megadja, hogy hányadik lokalizációs ciklust végeztük el.

locNumWithinCycle: az adott cikluson belüli lokalizációs üzenet sorszáma.

maxLocNumWithinCycle: hány lokalizációs üzenet van egy cikluson belül.

deltaSound: ez határozza meg, hogy cikluson belül milyen hosszú legyen két csipogás közötti idő.

- LOCALIZATION_TIME_DIFF_RESULT

Ez az üzenet mind rádión, mind UART-on keresztül küldhető. Egy lokalizációs ciklusban minden egyes rádiós és akusztikus jelpár megérkezése után kerül kiküldésre. Ez az üzenet tartalmazza a mért időkülönbségeket. Az observer küldi ki ezt az üzenettípust rádión amennyiben nem bázisállomás, és UART-on, ha ő a bázisállomás. Ha a bázisállomás egy ilyen üzenetet kap egy observertől, akkor tartalmát továbbítja a PC felé. Emiatt kell tudni, hogy bázisállomás vagy általános mote-e az adott egység. A PC ezeket az üzeneteket feldolgozza.

Az üzenet fő tartalma:

localizSenderMoteID: annak a mote-nak az azonosítója, aki a hangot és a rádiós üzenetet kiadta.

localizObserverMoteID: annak a mote-nak az azonosítója, aki a hangot és a rádiós üzenetet fogadta.

timeDifference: az akusztikus és a rádiós jel beérkezésének időkülönbsége.

locCycle: a lokalizációs ciklus azonosítója. Megadja, hogy hányadik lokalizációs ciklust végeztük el.

locNumWithinCycle: az adott cikluson belüli lokalizációs üzenet sorszáma.

maxLocNumWithinCycle: hány lokalizációs üzenet van egy cikluson belül.

deltaSound: ez határozza meg, hogy cikluson belül milyen hosszú legyen két csipogás közötti idő.

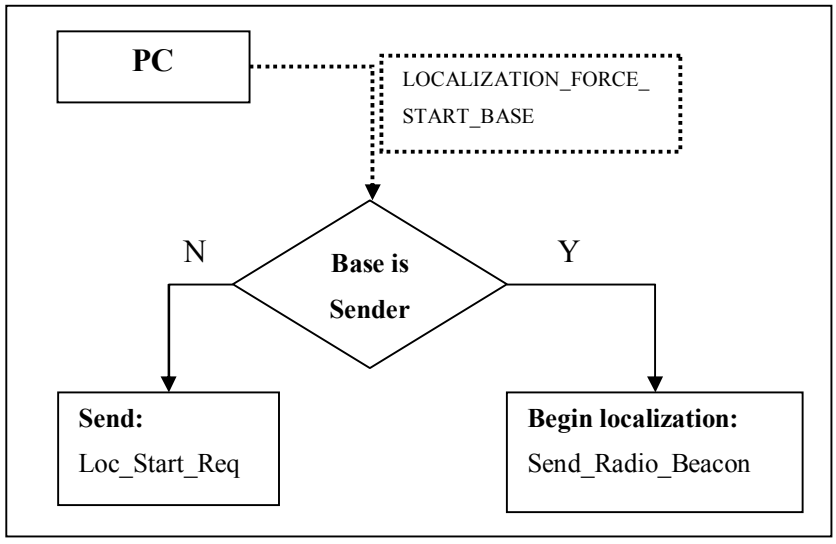
5.5. Feladatkörök szerinti működés bemutatása

Ebben az alfejezetben összefoglalom a mote-ok működését különböző üzemmódokban. Bemutatom a base és a general típusú szenzorok programjának folyamatábráját különböző alesetekben, ezt követően a fejezet végén található időzítési diagramokon látható a hálózat teljes működése a lehetséges üzemmódokban.

A működés feladatkörökre történő lebontásban:

Base feladatai:

1. *Localization_force_start_base* érkezése esetén megvizsgálja, hogy az üzenetben megadott sender ID megegyezik-e a saját azonosítójával. Ha nem egyezik meg, akkor rádión küld a hálózatba egy *localization_start_request,* tehát új mérési ciklust kezdeményező üzenetet. Ha a base a sender, akkor elkezd egy mérési ciklust, tehát kiadja a *localization_send_radio_beacon* és akusztikus jelek sorozatát. Az *5.3*. ábrán ez a működési diagram látható.

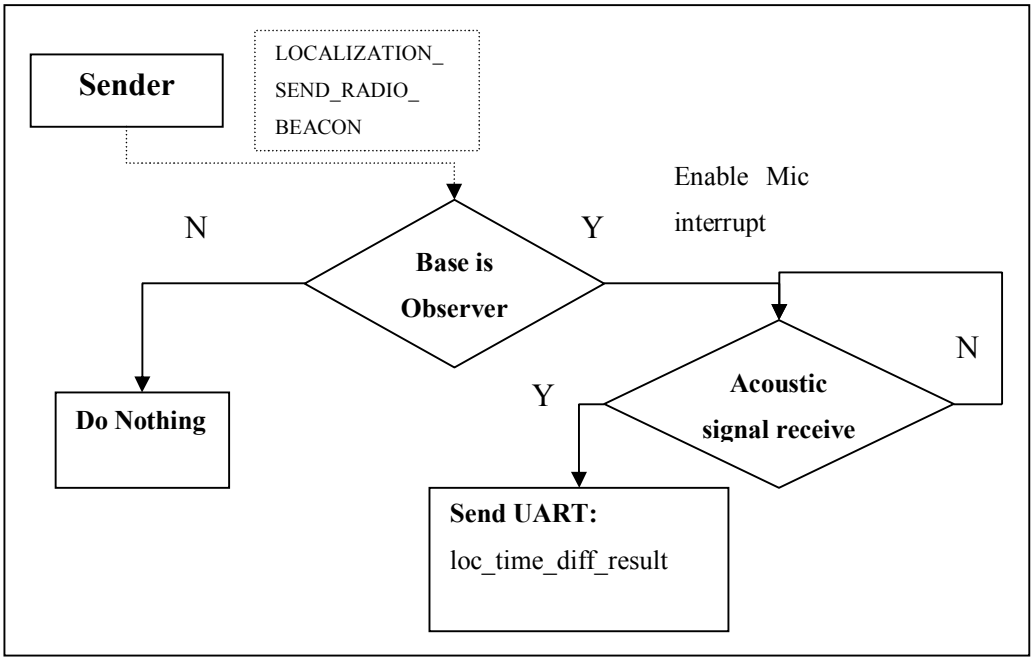


5.3. ábra. Base programja *localization_force_start_base* érkezése esetén

2. *Localization_send_radio_beacon* üzenet érzékelése esetén a bázis megvizsgálja, hogy observer-ként meg van-e címezve. Abban az esetben, ha nem, akkor nem tesz semmit, ha igen, akkor engedélyezi a mikrofon megszakítását, tehát felkészül az akusztikus jel fogadására. A lokalizációs ciklus során minden akusztikus jel beérkezését követően UART-on keresztül küld egy

localization_time_diff_result üzenetet a PC felé a rádiós és akusztikus jel érkezési idejének különbségéről. Az 5.4. ábra ezt a működést szemlélteti.

3. Ha a base mote *localization_time_diff_result* üzenetet kap, akkor továbbítja ugyanezt az üzenetet az UART-on keresztül a PC-nek.
4. Abban az esetben ha *localization_start_request* üzenetet érzékelne a base mote, ami megfelelő működés esetén nem fordulhat elő, akkor ezt az üzenetet eldobja a program. Azért nem kaphat ilyen üzenetet, mert csak a bázisállomás küldhet ilyen üzenetet.



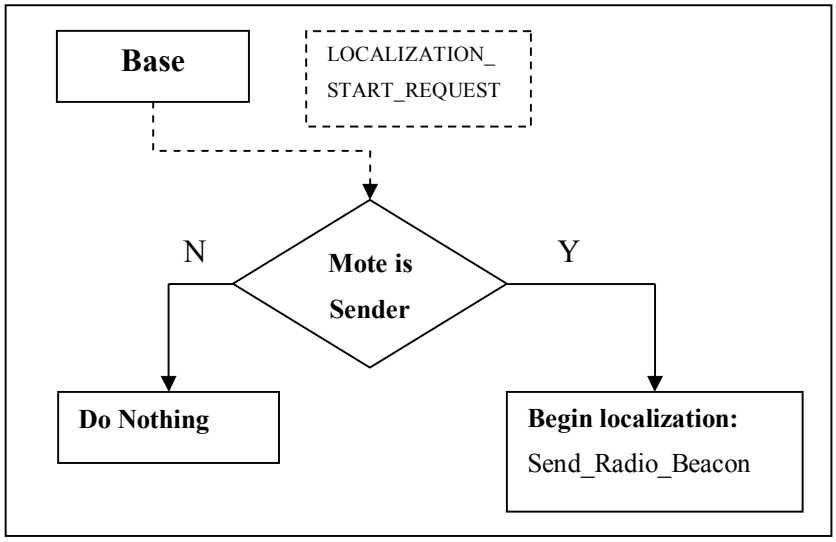
5.4. ábra. Base localization_send_radio_beacon *üzenet érzékelése esetén*

General mote feladati:

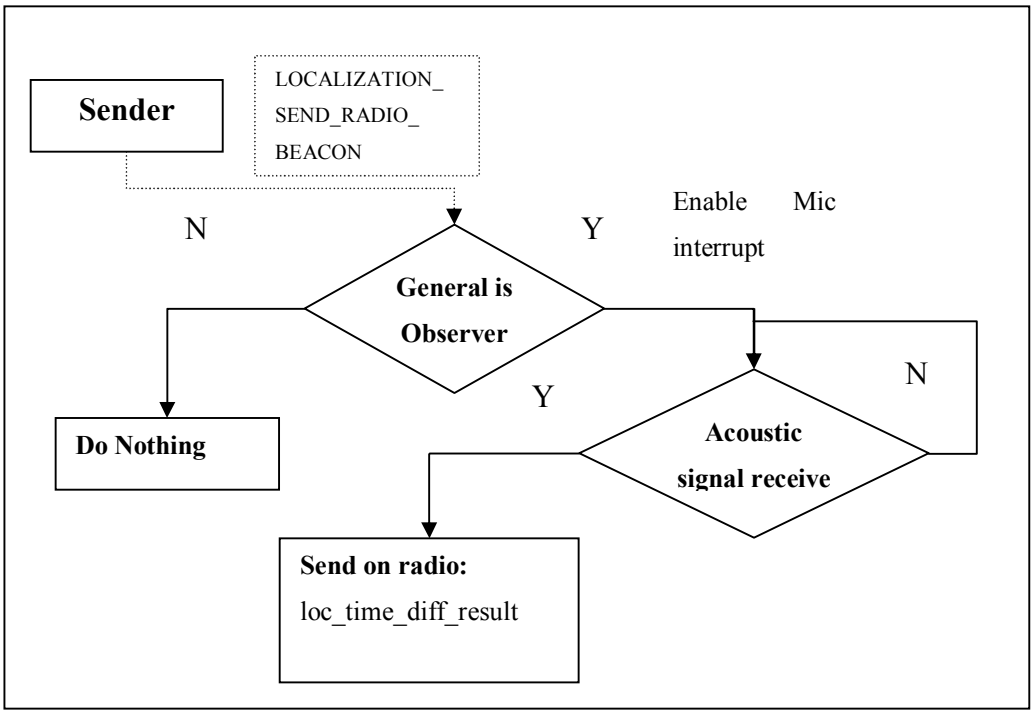
1. *Localization_force_start_base* Alapvetően minden mote general mote üzemmódban indul, de az első ilyen üzenet beérkezése után bázisként kell viselkednie. Ezek után a viselkedése az előzőekben, a bázisállomásnál megadottak alapján írható le.
2. *Localization_start_request* észlelése esetén, ha az üzenetben megadott senderMoteID-val megegyezik az adott mote azonosítója, akkor elkezd egy távolságmérő ciklust a megfelelő observer mote-tal. Az 5.5. ábra ezt a működést ábrázolja.
3. *Localization_send_radio_beacon* üzenet érzékelése esetén a szenzor megvizsgálja, hogy az azonosítója megegyezik-e a beacon üzenetben kiadott

observerMoteID-val. Ha nem ő a megcímzett observer akkor nincs teendője. Ha igen, akkor engedélyezi a mikrofon megszakítását, és várja az akusztikus jelet. A lokalizációs ciklus során minden akusztikus jel beérkezését követően rádión keresztül küld egy *localization_time_diff_result* üzenetet a bázisállomás felé a lokalizációhoz felhasznált rádiós és akusztikus jel érkezési idejének különbségéről. A bázisállomás ezeket az üzeneteket továbbítja a PC felé a bázisállomás működését megadó leírások alapján. Az 5.6. ábra mutatja ezt a programrészletet.

4. *localization_time_diff_result* üzeneteket figyelmen kívül hagyja, mivel azok lekezelése a bázisállomás feladata.



5.5. ábra. General mote Localization_start_request *észlelése esetén*



5.6. ábra. General mote Localization_send_radio_beacon *üzenet érzékelése esetén*

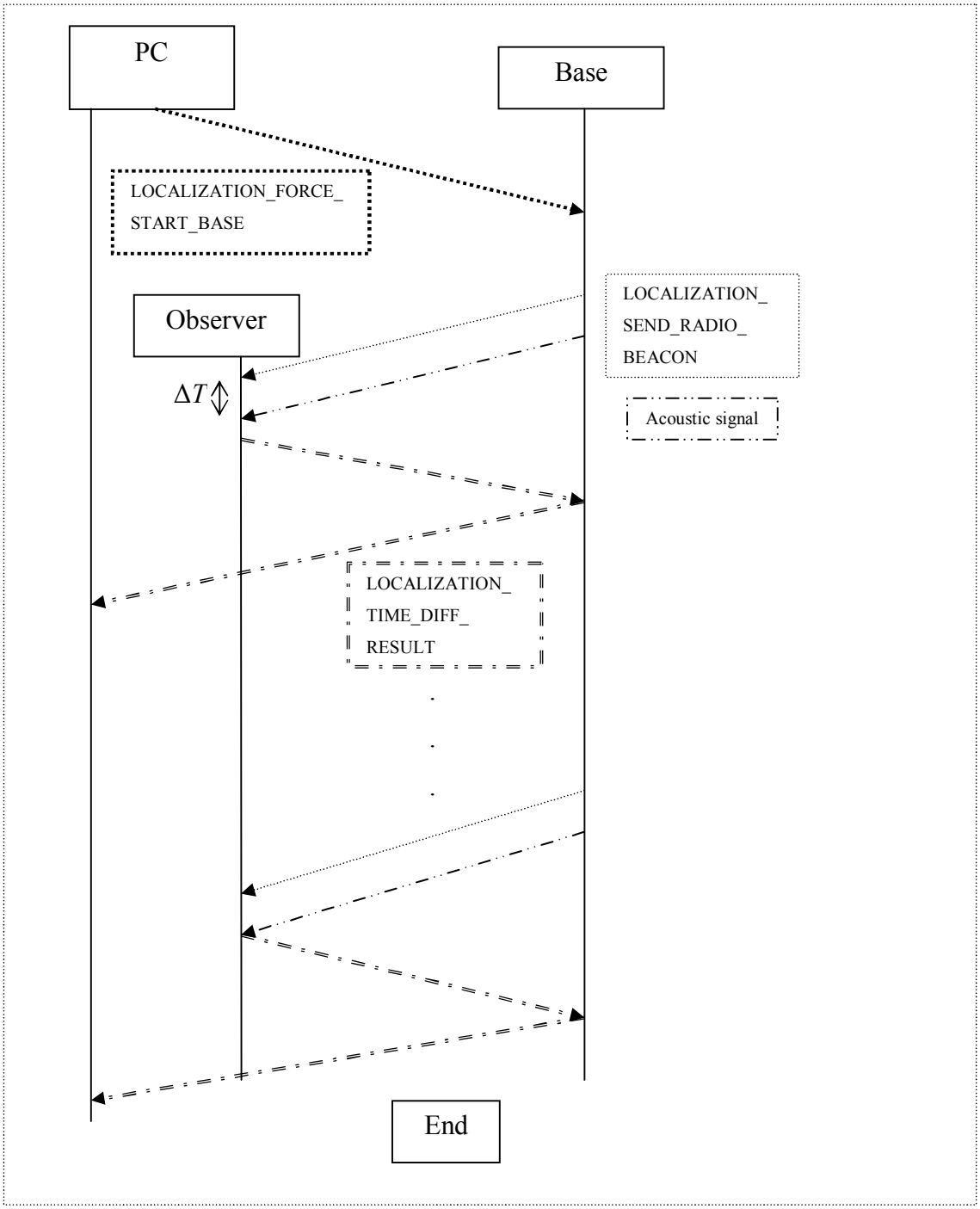
A következőkben a szenzorhálózat teljes működésének idődiagramjait mutatom be. A ábrákon a különböző nyilak jelölik az eltérő üzenettípusokat. A vastag pontozott álló nyíl jelöli a PC-től a base felé küldött Localization_force_start_base üzenetet. A szaggatott vonalakból álló nyíl a Localization_start_request-et jelöli. A vékony nyíl a Localization_send_radio_beacon, a kétfajta hosszúságú szakaszokból álló szaggatott stílusú nyíl a kiadott Akusztikus szignált szimbolizálja. Végül a mért értéket tartalmazó Localization_time_diff_result üzenetet a kettős szaggatott nyíl jelöli.

Az egyes aleseteket röviden összefoglalva:

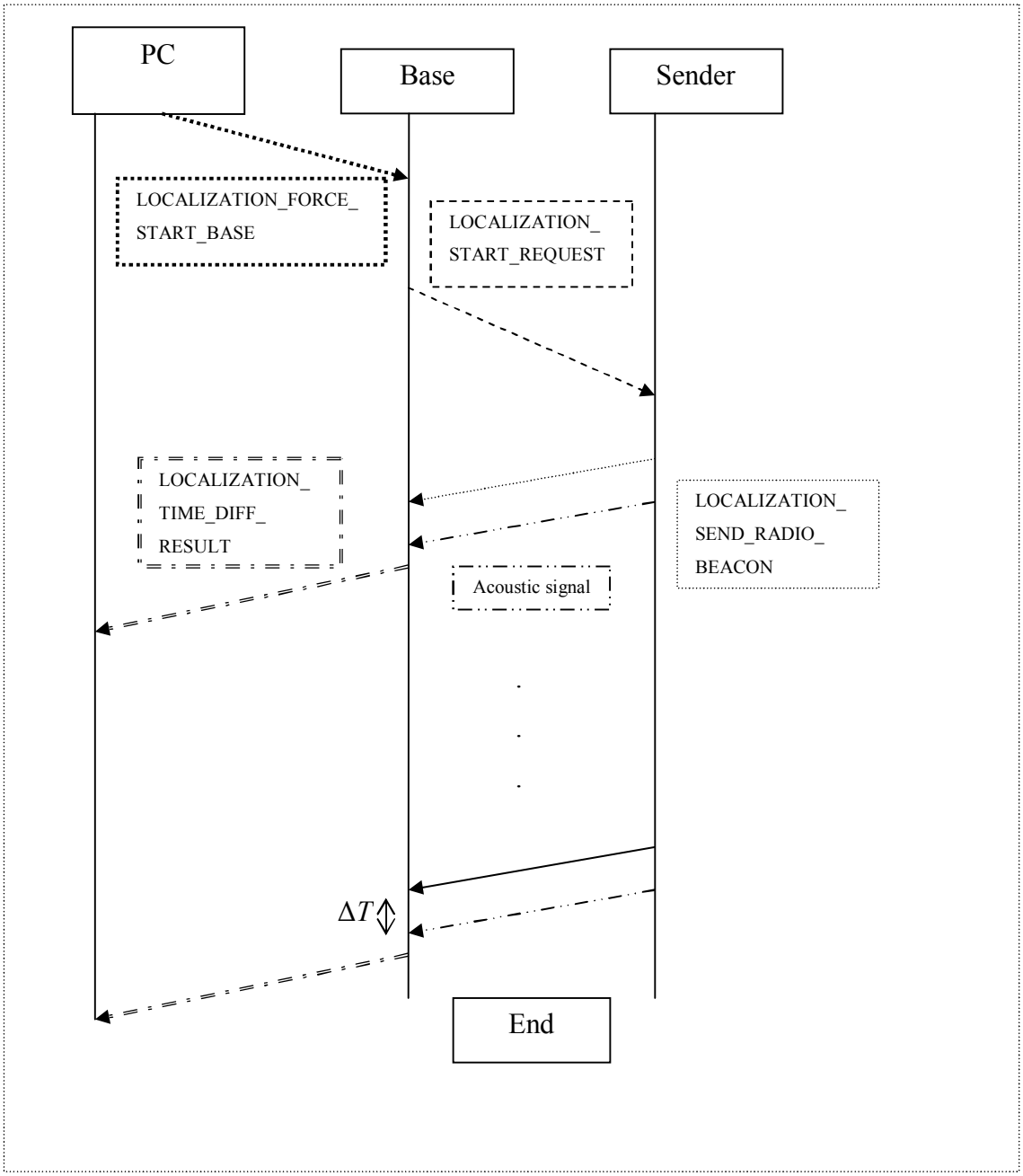
1. 5.7. ábrán látható az az eset amikor a bázis sender üzemmódban működik. A PC-ről érkező Localization_force_start_base üzenetből érzékeli, hogy ő a sender. Elkezdi sugározni a Localization_send_radio_beacon jelet és az akusztikus jelet, majd az observer-től visszakapott rádiós Localization_time_diff_result üzenetet UART-on továbbítja a PC-nek feldolgozásra.
2. 5.8. ábra a bázis observer módban való működését illusztrálja. A PC az előző esethez hasonlóan kiadja a Localization_force_start_base üzenetet, de itta bázis ennek a hatására kiküldi a Localization_start_request üzenetet. Az adott ciklusban kijelölt sender veszi ezt, küldi a Localization_send_radio_beacon jelet

és az akusztikus jelet. A bázis méri a beérkezési időkülönbséget és minden mérés után UART-on küldi a PC-nek a Localization_time_diff_result üzenetet.

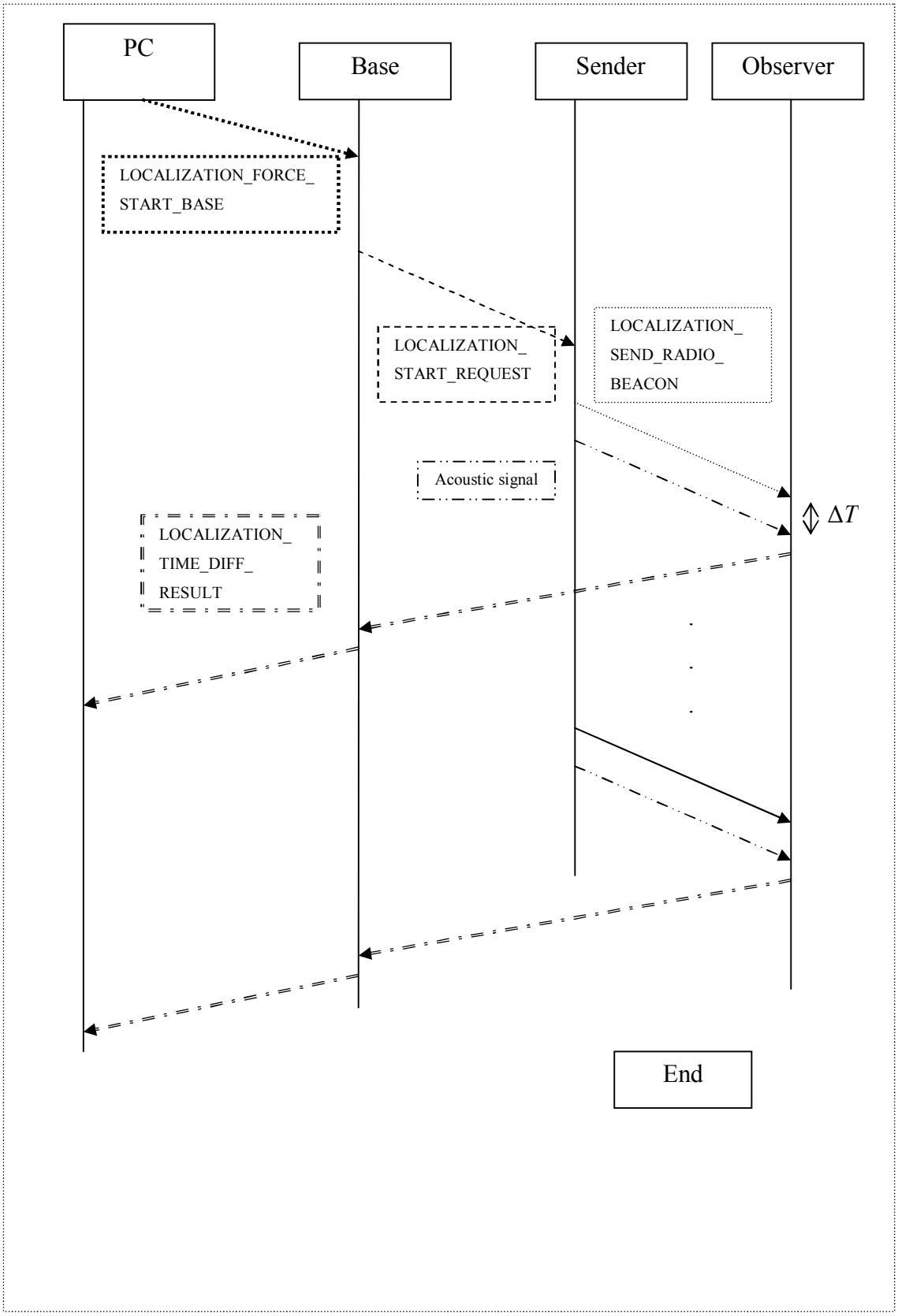
3. Az 5.9. az általános működést mutatja be, amikor a bázis csak adattovábbító funkciót lát el. A működés hasonló a 2. pontban leírtakhoz, a különbség csupán annyi, hogy a bázis helyett egy általános mote az observer. Az aktuális observer rádión elküldi minden mérés után az eredményeket, amit a bázis továbbít a PC felé.



5.7.ábra A bázis működése sender módban



5.8. ábra. A bázis observer módban van



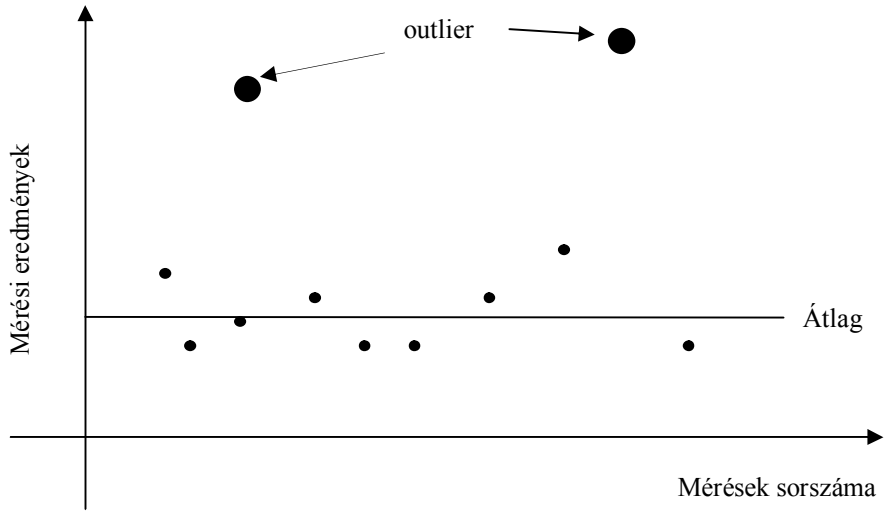
5.9. ábra. Általános működés, bázisállomás adattovábbító üzemmódban

6. Mérési eredmények feldolgozása

Az 5.2. fejezetben már foglalkoztam a működés során előforduló hibalehetőségekkel és azok forrásaival. Ebben a fejezetben bemutatom azokat az algoritmusokat, amikkel a hibákat korrigáltam, hogy a szenzorhálózat minél pontosabb távolságadatokat adjon vissza.

6.1. Zajos mérési eredmények és outlierek

Az egyik leggyakoribb előforduló hiba a véletlen zajból eredő eltérések. Ezek a véletlen jellegű hibák átlagolással kiszűrhetőek. Minél több mérést végzünk annál jobban megbecsülhető a pontos távolságérték. A 6.1. ábrán ezeket a hibákat tartalmazó méréseket a kisebb fekete ponttal jelöltem.



6.1. ábra. Mérési hibák típusai (illusztráció)

A másik, rendellenes működésből származó hibákat outliereknek nevezzük. Azon méréseket tekintjük outliereknek, melyek szignifikánsan eltérnek a véletlenszerű hibákkal terhelt mérések átlagától. Ezeket a hibákat illusztrálják a 6.1. ábrán a nagy

fekete pontok. Az átlagtól való eltérés mértékét a szóráshoz képesti viszonnyal jellemezzük. A véletlenszerű hibákkal terhelt mérések átlagának becslése első megközelítésben a mérési regisztrátum mediánjával történik, erre ugyanis a néhány kiugró eredménynek jóval kisebb hatása van, mint a számtani átlagra. Ezen megfontolás miatt a medián várhatóan jobb becslést ad a zajos mérések átlagára, mint az egyszerű számtani átlagolás. Az általunk használt algoritmusban azon értékeket tekintjük outliereknek, melyeknek a mediántól való távolsága nagyobb, mint a számított szórás kétszerese. A kétszeres szórást azért választottuk, mert tapasztaltok alapján ez elég kis érték ahhoz, hogy kellő megbízhatósággal szeparáljuk az outliereket, de a hasznos mérési eredményeket csak kis mértékben dobjuk el. (Például normális eloszlású zajt feltételezve a méréseknek csupán körülbelül 5%-a esik ezen tartományon kívül.) Tapasztalatok alapján a fent leírt módszert érdemes többször megismételni a mérési regisztrátumon. Ennek oka, hogy amennyiben az outlierek értékei nagyban különböznek egymástól, akkor első körben elképzelhető, hogy csak a nagyobb értékű outliereket távolítja el az algoritmus. Az eljárás iteratív megismétlésével a különböző nagyságrendbe eső outlier-ek is nagy megbízhatósággal eltávolíthatóak. Az eljárást addig ismételjük, amíg az adott iterációban található eltávolított mérési eredmény. Az iterációk számát érdemes korlátozni (esetünkben 4-es korlátot alkalmaztunk), hogy előre nem látott okok miatt a hasznos mérési eredményekből ne távolítson el túl sokat az algoritmus.

Outlierek eltávolításának algoritmikus leírása a következő:
(a következőkben az m_i, az i-edik mérési eredményt jelöli)

1. regisztrátum szórásának számítása (σ)
2. regisztrátum mediánjának számítása (M)
3. ha $(m_i - M) \geq 2\,\sigma$, akkor az i-edik eredményt eltávolítom
4. ha van eltávolított elem, és az iterációk száma <4, akkor 1. pont ismétlése

6.2. Kalibráció

Vannak olyan hibatagok, melyek rendszeres hibát okoznak (például: beállási és feléledési idők miatt bekövetkező késleltetés). A rendszeres hibák hatása átlagolással nem csökkenthető, hanem egy kalibrációs lépést iktatunk be a mérésbe. Azt

feltételezzük, hogy a valódi és a mért értékek között lineáris összefüggés van, így a korrekció elvégezhető egy lineáris transzformációval. Az ilyen jellegű korrekció teljes mértékben képes az ofszet jellegű hibák korrigálására, illetve egy arányossági tényező figyelembevételére. A korrekció alakja a következő:

$$y_m = a \cdot x_m + b. \tag{6.1}$$

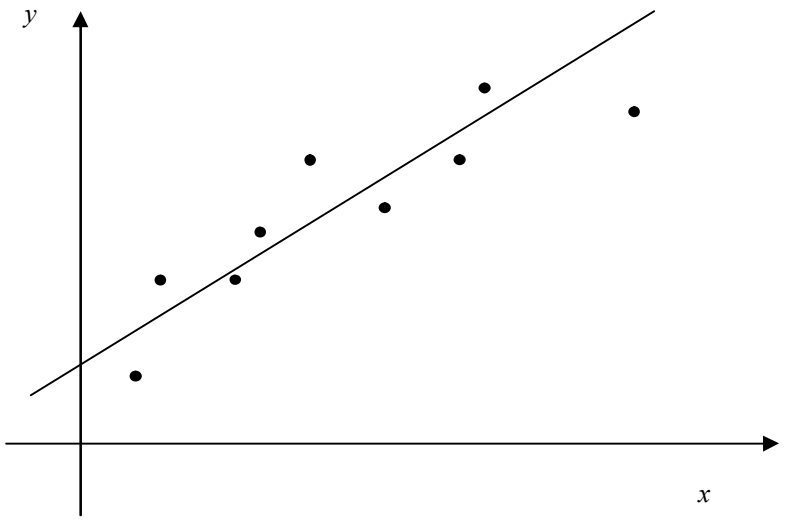
A képletben x_m a szenzorhálózat által mért eredmény, melyből származtatjuk az y_m értéket, mely már a valódi távolság becslőjének tekinthető. Az egyenletben a b jelű tag az ofszetkorrekcióért felelős, az a tényező pedig arányos hibák kompenzálásáért felelős, például nagyobb távolságok esetében (kisebb érzékelt jelszint) lassabban áll be az érzékelő. Az egyenletben az a és b konstansok előzetes mérések, tehát a kalibráció segítségével határozhatóak meg lineáris regresszió segítségével. A lineáris regresszió során legkisebb négyzetes értelemben a legoptimálisabb a és b paraméterek kerülnek kiszámításra. Erre a MATLAB hatékony segítséget kínál, például a `polyfit` parancs segítségével. A kalibráció menete a következő. Ismert távolságokban méréseket végzünk (például mérőszalag segítségével megmérjük a távolságot). A méréseket különböző távolságok esetén végezzük el, például 10-20 cm felbontásban. Minél több pontban mérünk, várhatóan annál pontosabb lesz a kalibráció eredménye. A mérési eredmények összegyűjtését követően a mérési pontpárokra egy egyenest illesztünk az illusztrált ábrának (6.2. ábra) megfelelően. Az ábrán x jelöli a szenzorhálózat által mért értékeket, y pedig a valódi, például mérőszalaggal lemért eredményeket.

A kalibráció során is történik bizonyos szelekció az adatok között hasonlóan a 6.1. alfejezetben leírtakhoz. Amennyiben egy adott mérés az outlier-ek eltávolítását követően is túlságosan bizonytalan, akkor ezt nem vesszük figyelembe a kalibrációs során. Ezen mérések eltávolítása automatizált módon történik. Ennek módszere a következő. Kiszámítjuk minden egyes távolságmérés esetén a mérés szórását (az outlier-ek eltávolítása utáni szórást). Megnézzük, hogy a különböző távolság esetén végrehajtott mérések szórásainak mi az átlagos szintje. Amely mérések ezen átlagtól jelentősen eltérnek, azokat nem vesszük figyelembe. Jelentős eltérésnek tekintjük az átlagos szórástól való legalább háromszoros eltérést. Ezek után a minden eddigi szűrésen átment mérési eredményekre illesztünk egy lineáris egyenest.

Az algoritmus rövid leírása:

1. mérések végzése többféle ismert távolság esetén a szenzorhálózattal
2. mérések előfeldolgozása a 6.1. alfejezetben leírtak alapján
3. nagy szórású mérési eredmények eltávolítása
4. kalibráció elvégzése lineáris egyenes illesztésével: *a* és *b* paraméterek meghatározása (6.1) egyenletben

A kalibráció elvégezését követően az *a* és *b* konstansok segítségével már a nyers mérési eredmények konvertálhatóak valódi távolságinformációvá.



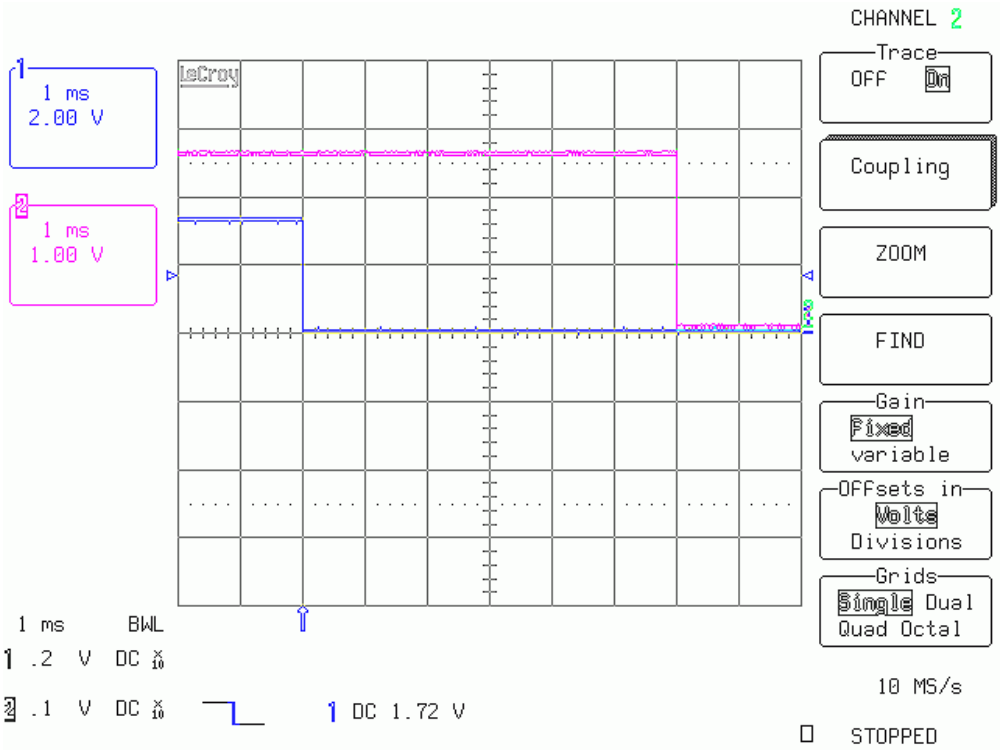
6.2. ábra. Egyenes illesztése (illusztráció)

7. Implementációs eredmények

A 7. fejezet bemutatja a mote-okra kifejlesztett és az adatfeldolgozást végző PC oldali, MATLAB-ban implementált programokat.

7.1. Lokalizációs szoftver a mote-okon

A lokalizációs szoftver fejlesztésének folyamatát három nagyobb részegységre osztottam. Az első részben kifejlesztettem a hang kiadását megvalósító rendszert, ezzel méréseket végeztem. Teszteltem az akusztikus jel generálását és detektálást, oszcilloszkóp segítségével. Az ezzel kapcsolatos mérési eredmény a 7.1. ábrán látható, ahol az oszcilloszkóp két csatornáján a jelek lefutó élei jelzik rendre a rádiós és az akusztikus jel fogadását.

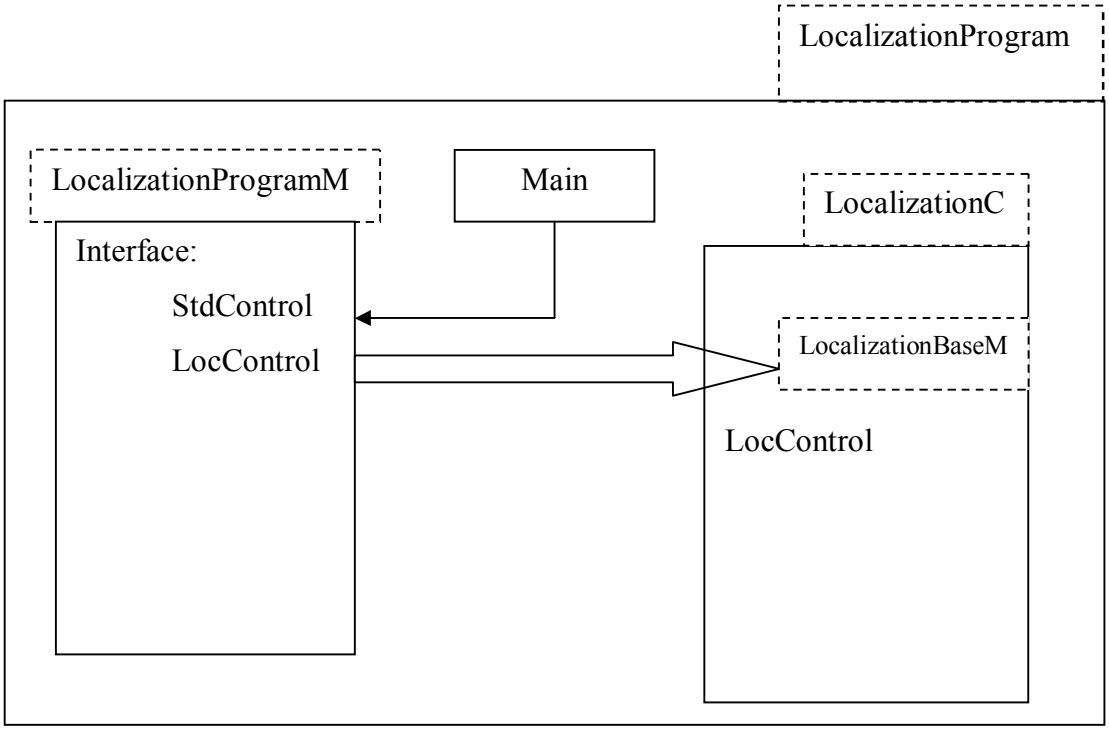


7.1. ábra. Akusztikus szignál kiadása(kék) és vétele(rózsaszín) oszcilloszkópon

Második lépesben külön fejlesztettem a kezdeményező base mote, valamint a hang és rádiós üzenetet kiadó sender mote programját. Ez a program is működőképes

volt, azonban külön kellet a bázis és a sender programját a megfelelő mote-okra telepíteni, és a kommunikáció csak a bázis és sender mote között valósulhatott meg, azaz két általános, nem bázis tulajdonságokkal rendelkező szenzor között nem volt lehetséges a távolság meghatározása.

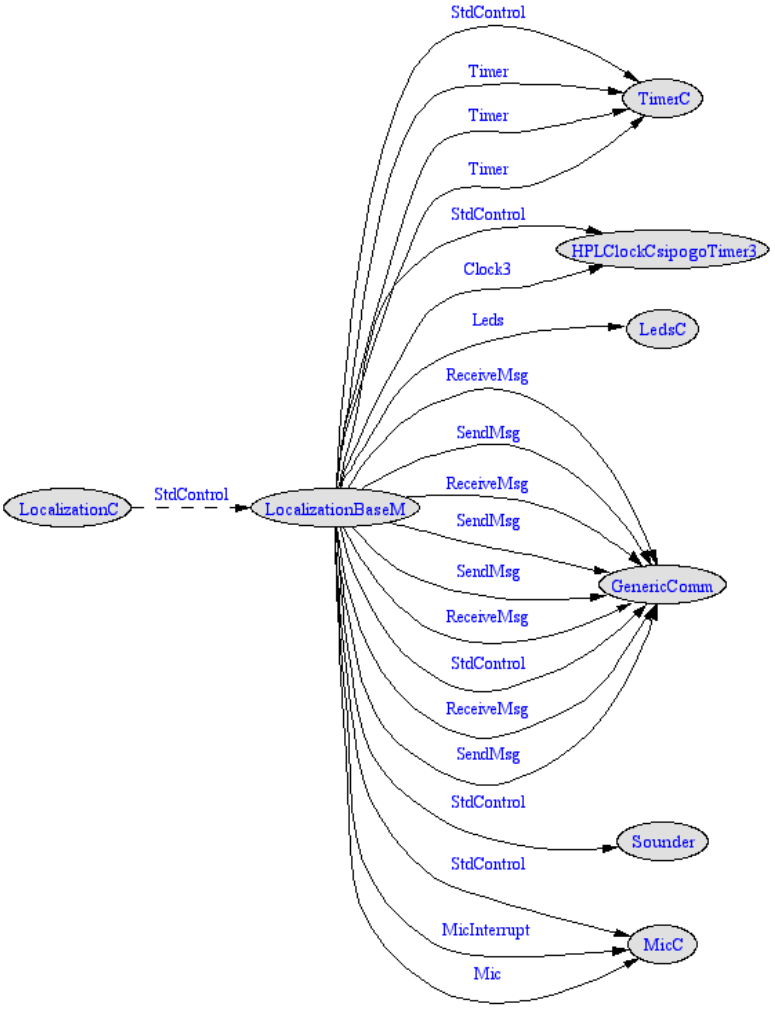
A lokalizációs szoftver a végső formájában egy külön komponensként került megvalósításra. Minden mote, ami a szenzorhálózatba tartozik, ugyanazt a programot használja a működésének megfelelőn. Ez azért is előnyös, mert más programokban is újrahasznosítható különösebb változtatások nélkül, és a feladtok jól szeparálhatóak. A PC-ről megadott azonosítók és a mérési ciklust meghatározó paraméterek segítségével a hálózat összes eleme meg tudja határozni az adott ciklusban betöltött feladatát.



7.2. ábra. Lokalizációs program komponensei

A 7.2 ábrán látható a megavalósított lokalizációs rendszer szoftverének komponensdiagramja. A legfelső hierarchiai szintű komponens a *LocalizationProgram*, ez a kerete az egész rendszernek. A rendszer fő eleme a lokalizációért felelős *LocalizationC* nevű konfiguráció, mely magába foglalja a lokalizációhoz felhasznált komponenseket. Mivel külön komponensként került megvalósításra, így a lokalizációs feladatot szoftverfejlesztési szinten el tudjuk szeparálni a szenzorok egyéb tevékenységeit megvalósító programrészektől. A *LocalizationC* komponensen belül a

LocalizationBaseM modulban történt a dolgozatban leírt lokalizációs lépések implementálása: különböző üzenetek lekezelése és küldése, mérőjelek generálása és időmérés. A lokalizációs komponens a LocControl interfészen keresztül vezérelhető: inicializálhatjuk a modult, engedélyezhetjük, illetve letilthatjuk a lokalizációt, illetve értesítést kapunk az interfészen keresztül, ha lokalizációs üzeneteket kap a szenzor. A *LocalizationProgramM* másodlagos funkciót tölt be a lokalizáció szempontjából. Jelen rendszerben feladata a lokalizációs modul inicializálása és a lokalizáció engedélyezése. Későbbiekben ezen modulban lehet implementálni a mote nem lokalizációhoz köthető feladatait (pl. környezeti paraméterek mérése). A 7.2. ábrán látható Main modul egy alapértelmezett modul a TinyOS operációs rendszerben. Hasonló szerepet tölt be, mint a C programozási nyelv main függvénye, tehát belépési pontot biztosít a programba, itt indítható el a felhasználói program az StdControl interfészt felhasználva.



7.3. ábra. LocalizationBaseM által használt modulok és interfészek

A *7.3*. ábrán a *LocalizationBaseM* által használt modulok és a köztük lévő kapcsolat látható. A modul a *TimerC, MicC, Sounder, LedsC, GenerComm* és *HPLClockCsipogoTimer3* komponenseket használja.

A *TimerC* modul egy lokalizációs cikluson belüli időzítésben játszik szerepet. Segítségével állítható be, hogy milyen hosszú legyen egy akusztikus szignál „csipogás”, és a mikrofon interrupt engedélyezését is ennek a modulnak a segítségével végzi a program. Ezt a három funkciót reprezentálja a három Timer nevű nyíl, melyek a *LocalizationBaseM*-től aTimerC-be mutatnak.

A *MicC* segítségével történik a mikrofon megszakításának engedélyezése és a mikrofon érzékenységének beállítása. Szintén ez a komponens jelzi egy event segítségével az akusztikus jel érzékelését.

A *Sounder* a hang kiadására szolgáló modul. Elindításakor megszólal a buzzer, leállításakor pedig elhallgat. A hang hosszát a hang kiadása és letiltása közt eltelt idő szabályozásával lehet beállítani.

A *GenericComm* segítségével történik a rádiós üzenetek küldése és fogadása. Mind a négy üzenettípus küldésére az üzenettípushoz tartozó interfészben implementálva van egy SendMsg függvény és egy visszaigazoló sendDone függvény, ami tulajdonképpen egy visszaigazoló jel arra, hogy az adott üzenetet sikerült kiküldeni. Ennek a függvénynek fontos szerepe van az akusztikus jel kiadásának engedélyezésében, mivel a mérést kezdeményező Localization_send_radio_beacon üzenet kiküldését visszaigazoló SendBeaconMsg.sendDone függvényének a hatására szólal meg a buzzer, és indul el a hang hosszúságát szabályozó számláló is. A rádiós üzenetek fogadását is ez a komponens végzi.

A *LedsC* a LEDek vezérléséért felelős, segítségével a LED bekapcsolható, kikapcsolható.

A *HPLClockCsipogoTimer3* modulnak kiemelt szerepe van, mert ebben a modulban történik a távolságmérés szempontjából egyik legfontosabb lépés, az egyszerre kiadott rádiós és akusztikus jel beérkezésének időkülönbség mérése. Itt történik a timer leosztásának beállítása és a számláló újraindítása minden egyes mérés esetén. A tizenhat bites számláló értékének kiolvasását is itt hajtjuk végre. Ezen számláló értékét továbbítjuk egy következő lépésben a PC felé.

7.2. PC oldali program

A számítógéppel a szenzorhálózat egy base funkciójú mote segítségével tartja a kapcsolatot. A PC-n a MATLAB nevű program segítségével dolgozzuk fel a beérkező adatokat, továbbá ugyanezen a felületen keresztül tudunk beavatkozni a lokalizációs rendszer működésébe. A kapcsolat soros (RS232) porton keresztül valósul meg, ez az UART kommunikáció.

A MATLAB-ban megírt függvényeket és szkripteket három csoportba lehet sorolni. Az egyik csoport feladata az UART kezelése, adatok küldése és fogadása, valamint megfelelő UART formátum előállítása. Ezek a kommunikációs funkciót ellátó programok. A függvények másik része a mért adatok feldolgozásáért felelősek, és kalibrációs feladatokat látnak el. Ezen függvények a jelfeldolgozási funkciókat valósítják meg. A megjelenítés és vezérlés pedig egy általam kifejlesztett GUI, vagyis grafikus kezelő felületen keresztül történik.

A soros kommunikációnál figyelembe kellett venni azt, hogy a mote milyen formátumban tud adatokat küldeni és feldolgozni. Ezt a formátumot nevezzük az UART kommunikáció keretének. A keret első és utolsó karaktere jelzi az adott üzenet kezdetét és végét, ebben az esetben ez a 0x7E hexadecimális érték. A következő bájt egy protokol érték. A hatodik bájtól található az adatmező, ebbe a részbe tudjuk beírni az általunk használni kívánt információkat. Az adatmezőben a több byteos adatok esetén először az adat LSB és utána az MSB byte-ja következik (little-endian formátum). Az utolsó előtti byte egy checksum, azaz ellenőrző érték. A kommunikációs protokolban a 0x7D és 0x7E karakterek speciális karakterek. A 0x7E karakter jelöli a kommunikációs keretek elejét és végét, a 0x7D escape karakter segítségével pedig a két speciális karakter szúrható be az adatsorba, amennyiben azok szerepelnek a továbbítandó adatok között. Az eredeti adatsorban a 0x7E speciális karaktert a soros vonalon átküldött adatsorban található 0x7D 0x5E karaktersor, 0x7D speciális karaktert pedig a 0x7D 0x5D karakterek jelzik. Ezzel a módszerrel elérhetjük azt, hogy az UART-on küldött adatok között a keretező 0x7E karakter ne szerepeljen, ugyanis azt más speciális karakterek helyettesítik. A PC-n kiküldött üzenet megfelelő formátumúvá alakítását a MATLAB-ban megírt LOCALIZATION_FORCE_START_BASE_CORR függvény végzi el. A bejövő adatok feldolgozása esetén az InputMOTE nevű függvény végzi a karakterkonverziót, az adatsor keretekre bontását pedig az UARTHandler nevű függvény. Tulajdonképpen ez a függvény végzi az UART olvasását, és előállítja a

feldolgozható üzeneteket, így az adatok kezelése sokban leegyszerűsödik. Egy tipikus UART üzenet decimális értékekké átváltva:

126 66 255 255 12 125 93 8 10 0 11 0 30 0 50 0 0 0 126

Ez már egy korrigált üzenet, amiben a speciális karakterek helyére már a megfelelő kettős helyettesítő értékek vannak írva. Az adott keretben a nyolcadik bytetól kezdődik az adatmező, a tízes jelenti a sender mote azonosítóját a tizenegy az observerét, a harminccal beállítjuk a deltaSoundot, tehát két csipogás között eltelt időt millisecundumban. Az ötvenes szám megadja a mérési ciklus hosszát, tehát a maxLocNumWithinCycle értékét.

Az előbb említett MATLAB függvények a legfontosabb kommunikációs funkciót ellátó programok.

A mérési eredmények feldolgozásáért az InputMoteDistance, estimDistance, a kalibrációért a saveDistData függvények a felelősek.

Az InputMoteDistance-ban a mote-tól érkező, az alapvető formátumkonverzión átesett üzenetkereteket dolgoztam fel: az üzenetben szereplő időkülönbségeket a megfelelő bájtokból kiolvasom, és átszámítom centiméterben megadott távolságértékekké. Optimális esetben minden ciklus végén egy maxLocNumWithinCycle darabnyi mérési értékből álló tömböt kell kapnunk, azonban gyakran előfordul, hogy nem érkezik be az összes mérés a PC-re például a rádiós kommunikáció miatti adatvesztés következtében, vagy ha az akusztikus jelet nem érzékeli a szenzor. Ebben az esetben az elveszített adatokat nem vesszük figyelembe a mérés során.

Az így kapott távolságadatok az estimDistance függvény segítségével dolgozhatjuk fel a 6.1. fejezetben említett algoritmusokat felhasználva. A visszatérési értékében lévő távolságok között már csak az adott kritériumoknak megfelelő mérési eredmények szerepelnek, továbbá ezen értékek szórása és átlaga.

Többek között ezeket az eredményeket is felhasználja a saveDistData függvény, amely elmenti a mért adatokat egy adattömbbe valamint adatfájlba, ahonnan az adatok később is visszanyerhetőek, így például végrehajtható egy kalibrációs fázis a mérések újbóli elvégzése nélkül. A függvény az eltárolt adatok felhasználásával, az előbb említett estimDistance függvény segítségével kiszűri az outliereket, a helyes eredményekből szórást és átlagot számít, és elvégzi az eredmények grafikus kijelzését.

Következő lépésben a 6.2. alfejezetben kifejtett kalibrációs algoritmust hajtja végre a program, és a megfelelő mérési eredményekre a MATLAB `polyfit`

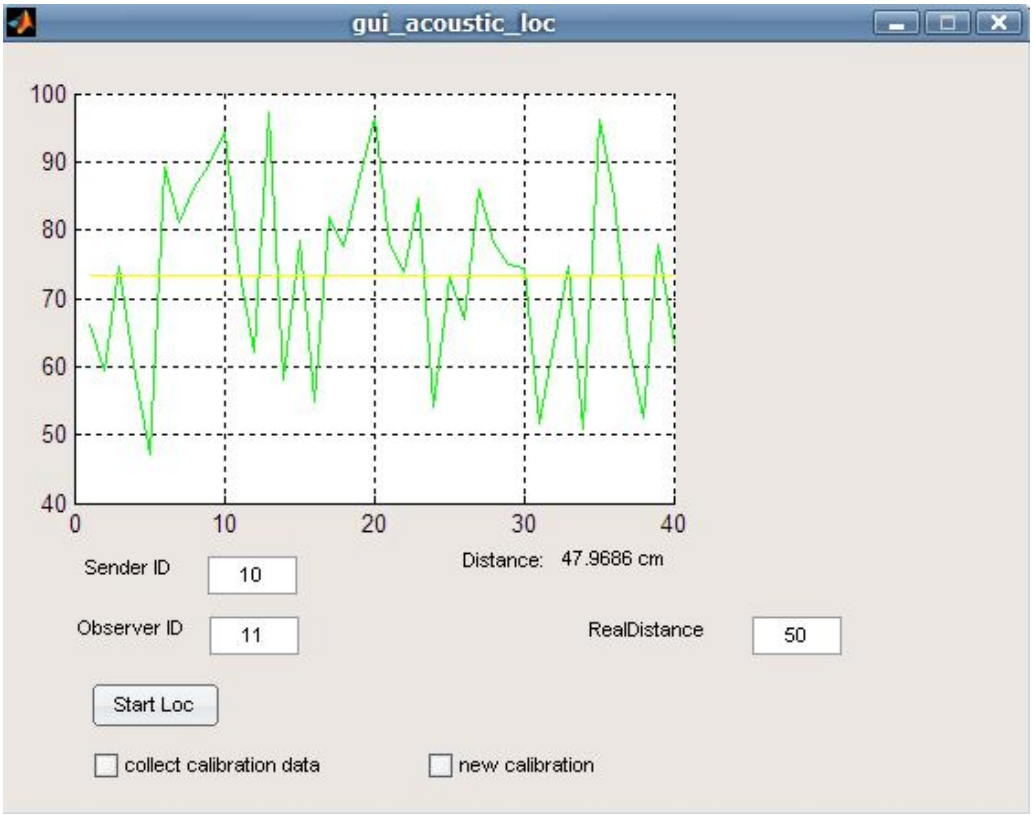
parancsával egy egyenest illesztünk. Ez történik a kalibrációs fázisban. A pontosabb eredmények érdekében akár minden mote-ra külön-külön el lehet végezni a kalibrációt.

Abban az esetben, amikor már befejeztük a kalibrációt, és el szeretnénk kezdeni a tényleges távolságok meghatározását, akkor már nem mentjük el a mért értékeket, hanem az illesztett egyenes segítségével, a (6.1) egyenletet kiértékelve egy kalibrált távolságértéket kapunk.

A praktikusabb kezelhetőség érdekében kifejlesztettem a MATLAB segítségével egy, a 7.4. ábrán látható grafikus kezelőfelületet a szenzorhálózatban történő mérések kezelésére. A kezelői felületen a következő beállítási lehetőségek találhatóak:

- a lokalizációban résztvevő mote-párok beállítása
- kalibrációs adatok készítése
- új kalibrációs fázis kezdése (régi adatok törlése)

Amennyiben a kalibrációt már elvégeztük, akkor a valós távolságot is megjelenítjük centiméterben. A kezelői felületen található grafikonon a mérés után kirajzolódnak az adott mérés során meghatározott távolságok az outlierek eltávolítása után, és a távolságok átlaga. A kezelői felületen a „Distance:” mező után kalibrált mérés esetén a (6.1) egyenlet felhasználásával korrigált átlagos távolság jelenik meg, kalibráció nélküli távolság meghatározás esetén pedig a szenzorhálózat által mért távolság. A mérés indítása a Start Loc gombbal lehetséges.

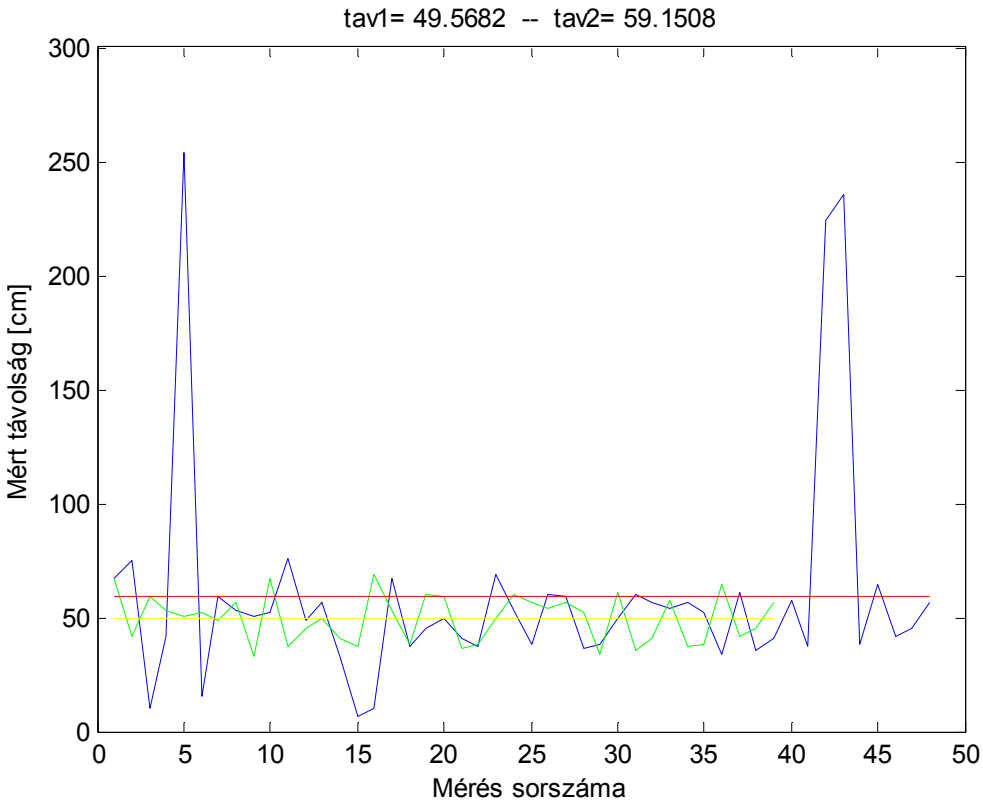


7.4. ábra. A szenzorhálózat kezelőfelülete

8. Mérési eredmények

Az eddigi fejezetekben bemutatott szenzorhálózattal különböző teszteket végeztem el a mérés pontosságának és a mérési tartomány meghatározására, továbbá a szenzorhálózat megbízható működésének tesztelésére.

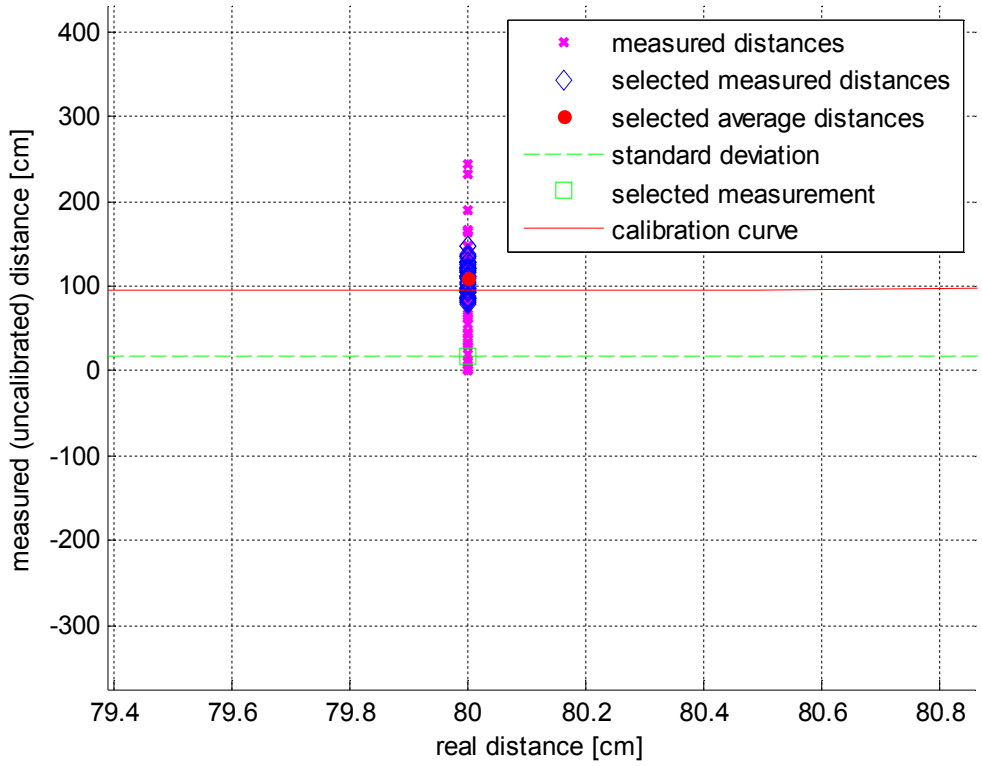
A tesztelés során egy adott mérési ciklushoz tartozó eredményeket ábrázoló grafikon látható a 8.1. ábrán. A kék grafikon az eredeti, míg a zöld a 6.1. alfejezetben leírtak alapján korrigált mérési eredményeket ábrázolja Látható, hogy a tav2 az eredeti, még korrigálatlan méréseknek az átlagát adja meg (piros vonallal jelölt szint), míg a tav1 a már outlierektől megtisztított értékek átlagát mutatja cm-ben (sárga vonallal jelölt szint).



8.1.ábra. Egy mérési ciklus eredménye

A kalibrációs méréseknek a kalibrációs konstansok meghatározásán kívül önmagukban is jelentős szerepük van, hiszen kalibráció során nagy számú, szisztematikusan elvégzett mérések eredményei állnak elő. Ezen eredmények

felhasználhatóak a rendszer viselkedésének jellemzésére, emiatt a következőkben bemutatásra kerül néhány kalibrációs mérés eredménye.

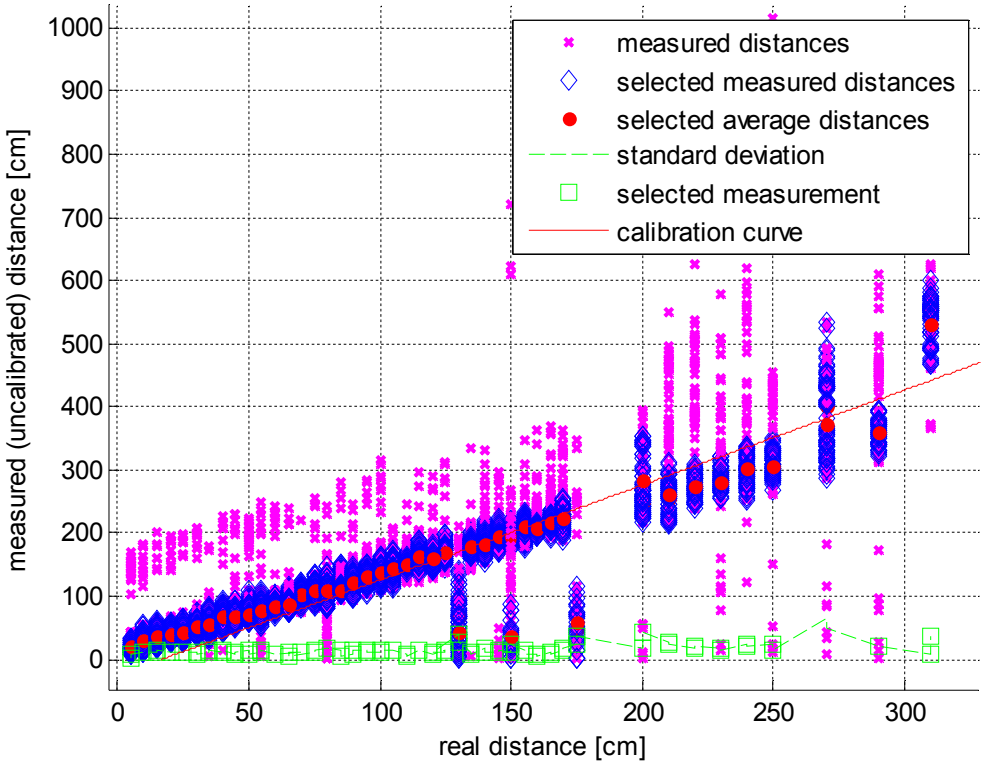


8.2. ábra. Egy mérési ciklus eredménye kalibrációnál

A *8.2.* ábrán egy adott távolságon, ebben az esetben 80 cm-en végzett kalibrációs mérés eredménye látható. A zöld négyzet szimbólum az adott mérési eredmények szórását jelöli. Az illesztett egyenest a piros vonal jelöli természetesen nem csak erre az egy mérésre vonatkoztatva. A rózsaszín X-el jelölt adatok a mérési eredmények, ezek közül ki vannak válogatva a 6.1. alfejezetben leírtak alapján a használható mérések, melyeket a kék, csúcsára állított négyszög jelöl. A piros pont ezeknek a korrigált mérési eredményeknek az átlagát adja meg, tulajdonképpen ezekre a korrigált átlagokra történik a kalibráció végén az egyenes illesztése.

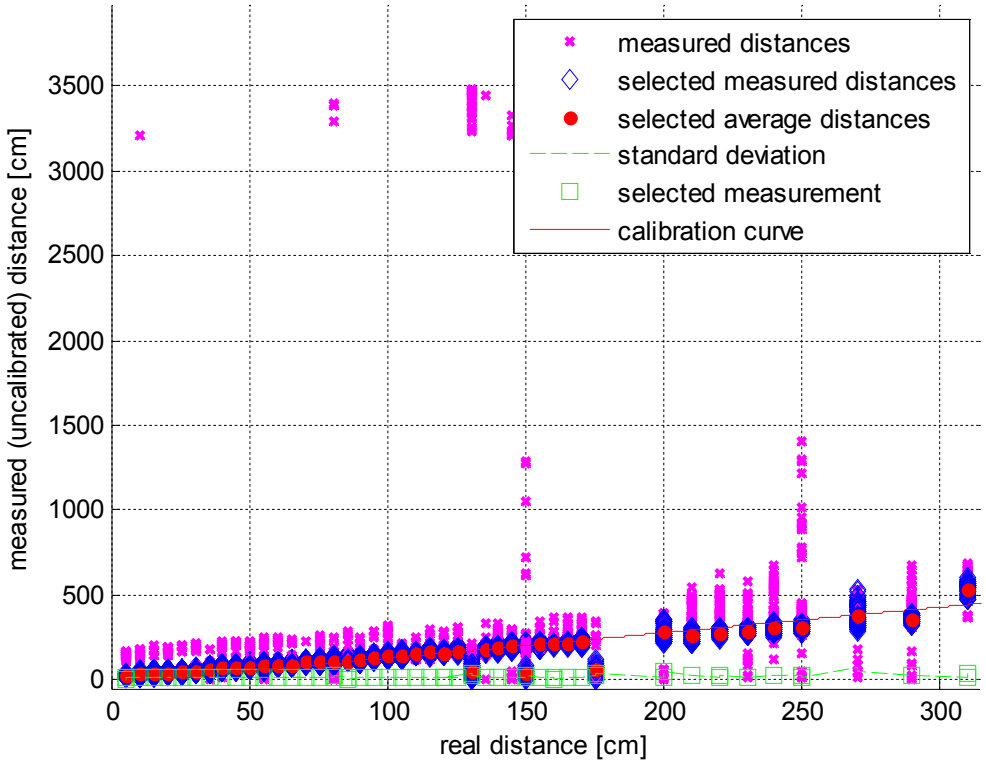
Első lépésben 5-310 cm-ig végeztem méréseket kalibráció céljából, ennek az eredménye a 8.3. ábrán látható. A mérések alapján állítható, hogy a távolság változását megfelelően követik az eredmények. A távolság növelésével a mért távolság is növekszik. A kalibrációs ábrán látható, hogy közel lineárisan helyezkednek el a mért értékek a körülbelül 120 cm-ig tartó tartományban. 120 cm fölött már adódnak bizonytalanságok a mérésekben, de eddig kifejezetten megbízhatóan mér a rendszer.

200-300 cm-es tartományban láthatóan a szórás is meglehetősen nagy. Körülbelül 20-35 cm a szórás a nagyobb távolságok esetén, az első egy méteren a legrosszabb esetben is 15 cm alatt marad.



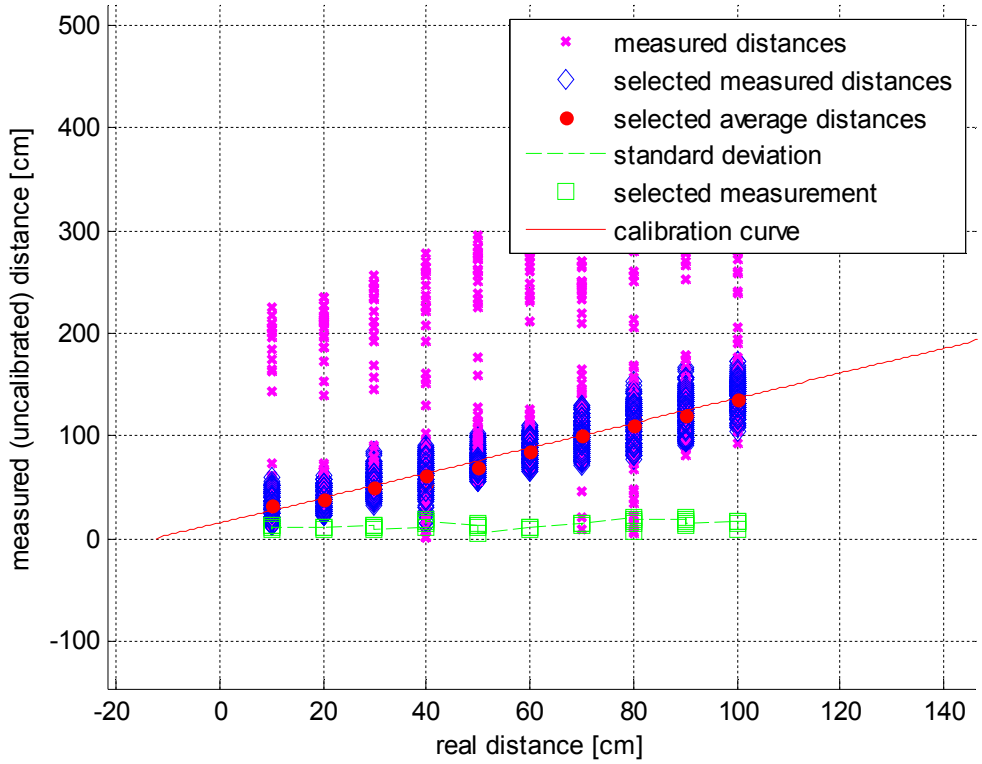
8.3.ábra. 5-310cm illesztett egyenes

A 8.4. ábrán egy nagyobb tartományban ábrázoltam a méréseket, így a jelentősen kilógó eredmények is láthatóak, míg a 8.3. ábrán a hasznos tartományra fókuszáltam.



8.4. ábra. 5-310cm illesztett egyenes (nagyobb tartományban)

Második esetben 10-100 cm-ig végeztem a kalibrációt. Ezeket a mérési eredményeket mutatja a 8.5. ábra. A legnagyobb szórás közel 20 cm, de átlagban körülbelül 10 cm körül mozog.



8.5. ábra. 10-100 cm kalibrációja

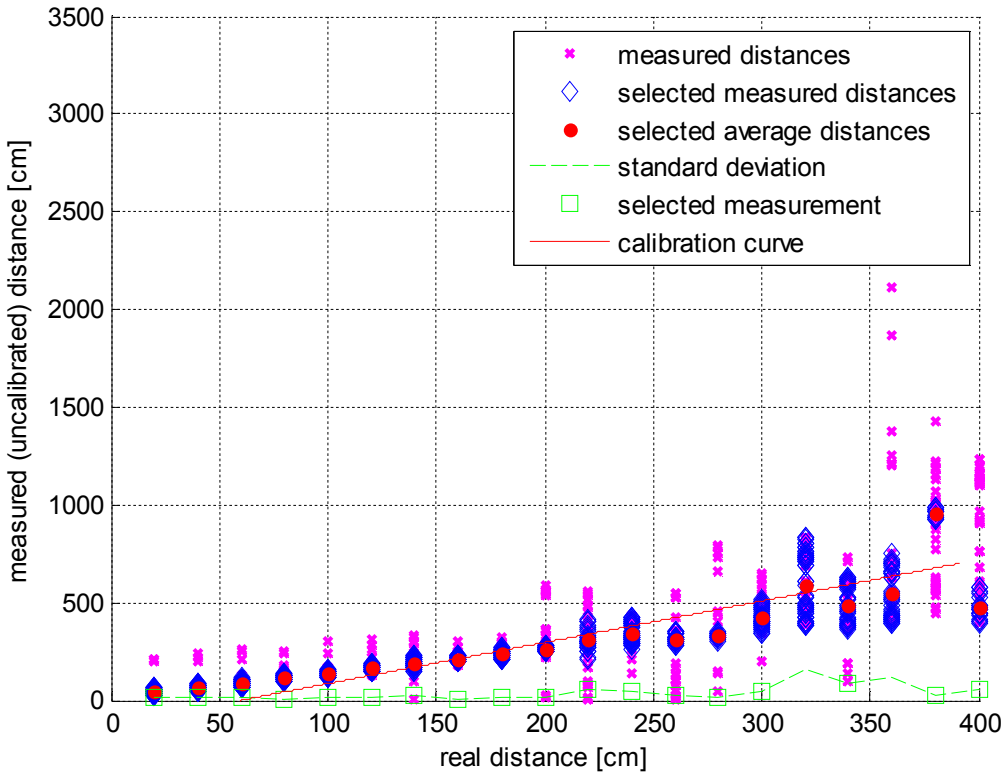
Az 5-310 cm és 10-100 cm-es tartományban végzett kalibrációs eredmények felhasználásával is végeztem ellenőrző méréseket, melyeknek eredményei a II. táblázatban találhatóak. Az ellenőrző mérések során fix, előre lemért távolságokra helyeztem el egymástól két szenzort, és három mérést végeztem minden pozícióban. Az így kapott méréseket a kalibráció felhasználásával korrigáltam. A táblázat minden sorában fel van tüntetve a valódi távolság és a hozzá tartozó három kalibrált mérés eredménye (mérés 1 … mérés 3).

II. Táblázat. *Ellenőrző mérések eredményei*

5-310cm kalibrációs tartomány	valódi távolság [cm]	mért távolság [cm] mérés 1	mérés 2	mérés 3
	15	37,36	35,4	37,12
	30	45,69	48,2	45,84
	50	67,2	67,07	64,36
	75	90,03	87,1	84,2
	100	105,79	111,07	101,86
	150	172,17	149,98	143,55
10-100cm kalibrációs tartomány	valódi távolság [cm]	mért távolság [cm] mérés 1	mérés 2	mérés 3
	15	17,321	18,01	15,78
	30	29,965	27,8	29,82
	50	46,023	47,97	44,6
	75	76,08	69,24	69,34
	100	101,98	102,45	101,4
	150	137,4	147,6	151,6

Az 5-310 cm-en végzett kalibrációhoz képest jelentkező javulás a mérési hibákban valószínűsíthetően annak köszönhető, hogy a kalibrációt a nagyobb távolságban (2-3 m-en) elvégzett kis pontosságú mérések lerontják. A 100 cm-es tartományban végzett kalibráció esetén viszont a kalibrációs mérések szórása kicsi, tehát ebben a tartományban pontosabb mérést tesznek lehetővé. Ezen megfigyelés felveti annak a lehetőségét, hogy a kalibrációt érdemes lehet mérési tartományonként elvégezni, így kisebb távolság esetén pontosabb mérést érhetünk el.

A harmadik mérés során a cél a mérési távolságtartomány meghatározása volt, ezért 20-400 cm-ig végeztem méréseket. Két méter fölött végzett mérések eredményei már közel sem lineárisak, mint ahogy a 0-100 cm-es tartományban tapasztalható. Az eredmények szórása nagy, 340 cm-es mérésnél 91 cm, ami már pontos mérésre használhatatlan. Kijelenthető, hogy a szenzorhálózat megbízható működésének a határa körülbelül 2 méteres távolságon belül van.



8.6. ábra. 20-400 cm-es kalibráció

Több mérés és a teszteredmények értékelése után kijelenthető, hogy az akusztikus távolságmérés módszerével működő szenzorhálózat a célnak megfelelően működik. A 0-150 cm-es tartományban akár 5-10 cm-es pontatlansággal képes megmérni a távolságot. Az utolsó mérésből látható, hogy akár 4 méteres távolságra is tud távolságot becsülni, de a 2 méter feletti mérések már nem megbízhatóak, akár 50-80 cm-es szórása is lehet ezeknek a méréseknek. Nagyobb távolságok mérése esetén inkább csak arra használható, hogy megbecsülje, hogy az adott szenzor a mérőjelet kiadó egység közelében található-e, változik-e a helyzete, esetleg egy adott tartományon kívül, vagy belül található-e az adott mote.

9. Összefoglalás, Kitekintés

A dolgozatomban bemutatásra került egy akusztikus távolságmérés alapuló szenzorhálózat kifejlesztése. A távolságmérésre felhasznált technika a jelek terjedési idejének különbségén alapszik. Bemutattam rendszer tervezési és implementációs kérdéseit. Mind a szenzorokon, mind a PC-n megalkottam a lokalizációhoz szükséges szoftver környezetet. A tapasztalatok alapján a szenzorok által mért közvetlen adatok nem használhatóak a távolságok pontos meghatározására, ezért a dolgozatom szerves részét képezi az eredmények korrigálása, megfelelő átlagolás és kalibráció elvégzése.

A kifejlesztett rendszerben történő mérések azt mutatják, hogy a megvalósított távolságmérési eljárás körülbelül egy-két méter távolságban nyújt megbízható eredményt, ebben a tartományban használható fel távolságadatok kinyerésére. Nagyobb távolságok esetén,(körülbelül három-négy méter) a működés nem megbízható, ebben az esetben szimbolikus lokalizációra képes, mely során nem konkrét távolsági adatokkal dolgozunk, hanem különböző tulajdonságokat rendelünk az adott mérésekhez (például közelében van a szenzor).

A rendszer továbbfejlesztésére több irány felvázolható. Például a rendszert alá lehetne vetni kültéri és további beltéri teszteknek, hogy megvizsgáljuk a reflexió hatását. Másik irány a multi-hop hálózatra való kiterjesztés. A most használt csillag elrendezést, egy nagyobb kiterjedésű, kisebb rendszerigényű multi-hop hálózattá lehetne alakítani. A jelenlegi centralizált feldolgozási rendszert ennek az alternatívájává, tehát decentralizálttá lehetne továbbfejleszteni, ahol már feldolgozott adatokat küldünk a mote-ok segítségével, ez nagyban csökkentené a rendszer leterheltségét.

A mérési eredményeknél megadott pontosság növelésére egy további lehetőséget kínál a több szenzor méréseiből történő információkinyerés, aggregáció. Ezen ötletet a dolgozatomban nem vizsgáltam, de a szenzorhálózatokban alkalmazott sokszor nagy számú node magában hordozza a lehetőséget.

Irodalomjegyzék

[1] Simon Gyula: Mik a szenzorhálózatok?
URL: http://www.dcs.vein.hu/~simon/SH/

[2] Shooter Localization
URL: http://w3.isis.vanderbilt.edu/projects/nest/applications.html

[3] A. H. Sayed, A. Tarighat, and N. Khajehnouri, "Network-based wireless location: challenges faced in developing techniques for accurate wireless location information," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 24–40, 2005.

[4] N. Patwari, J. Ash, S. Kyperountas, I. Hero, A.O., R. Moses, and N. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 54–69, 2005.

[5] Guoqiang Mao, Barış Fidan and Brian D.O. Anderson „Wireless sensor network localization techniques," *Computer Networks*, Volume 51, no. 10, 11 July 2007, Pages 2529-2553

[6] WIKIPEDIA: GPS
URL: http://hu.wikipedia.org/wiki/GPS

[7] K. Pahlavan, X. Li, and J. P. Makela. „Indoor geolocation science and technology," *IEEE Commun. Magazine*., February 2002.

[8] K. Romer, "The lighthouse location system for smart dust," in *Proceedings of MobiSys 2003 (ACM/USENIX Conference on Mobile Systems, Applications, and Services)*, 2003, pp. 15–30.

[9] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *Proc. of the Sixth Annual ACM International Conference on Mobile Computing and Networking*, 2000, pp. 32–43.

[10] Dr. Vidács Attila előadása: Lokalizáció és nyomkövetés, mobilitás
URL:
http://portal.mit.bme.hu/oktatas/targyak/vimm9084/jegyzet/Vidacs_09_I/091202.pdf

[11] Joshua A. Tauber. „Indoor Location Systems for Pervasive Computing." August 2002.

[12] Jeffrey Hightower, Gaetano Borriello. "SpotON: An Indoor 3D Location Sensing Technology Based on RF Signal Strength." *UW CSE Technical Report #2000-02-02*, University of Washington, February 2000.

[13] Crossbow, *MPR/MIB Users Manual*, Revision B, June 2006

[14] Crossbow, *MTS/MDA Sensor Board Users Manual*, Revision B, June 2006

[15] Németh Zsolt, „Helymeghatározás vezetéknélküli rádióhálózatokban”, BSc szakdolgozat, Budapesti Műszaki és Gazdaságtudományi Egyetem, Méréstechnika és Információs Rendszerek Tanszék., 2009

[16] TinyOS Tutorial:
URL: http://www.tinyos.net/nest/doc/tutorial/