Mérési útmutató a
Beágyazott és ambiens rendszerek laboratórium (vimia350)

Elosztott rendszerek és szenzorhálózatok 2.

című méréséhez

Készítette:
Orosz György
BME-MIT

2008. március

10. mérés

Elosztott rendszerek és szenzorhálózatok 2.

1. A mérés célja

A mérés során két olyan alkalmazást ismerhetünk meg, melyekben vezetéknélküli szenzorhálózat segítségével történik a működéshez szükséges jelek gyűjtése. Egyik alkalmazás egy akusztikus lokalizációs rendszer, melyben a szenzorok mérési eredményei alapján egy akusztikus forrás pozíciójának meghatározását végezzük el. Ilyen problémával találkozhatunk például olyan katonai alkalmazásokban, ahol egy lövés által keltett hanghullámok alapján több szenzor által érzékelt akusztikus jel felhasználásával próbálják meghatározni a lövést leadó személy pozícióját:
http://www.isis.vanderbilt.edu/projects/nest/applications.html. Az alkalmazás másrészről sok hasonlóságot mutat a célkitűzés (helymeghatározás) és a felhasznált eljárások tekintetében a radarrendszerekkel is.

Másik alkalmazás egy olyan aktív zajcsökkentő (ANC: Active Noise Control) rendszer, melyben a szenzorokat akusztikus zaj érzékelésére használjuk fel, és az érzékelt zaj alapján egy jelfeldolgozó processzor (DSP) olyan úgynevezett ellenzajt állít elő, mely az érzékelt zajjal ellentétes fázisú, így csökkenti annak teljesítményét az érzékelt pontban.

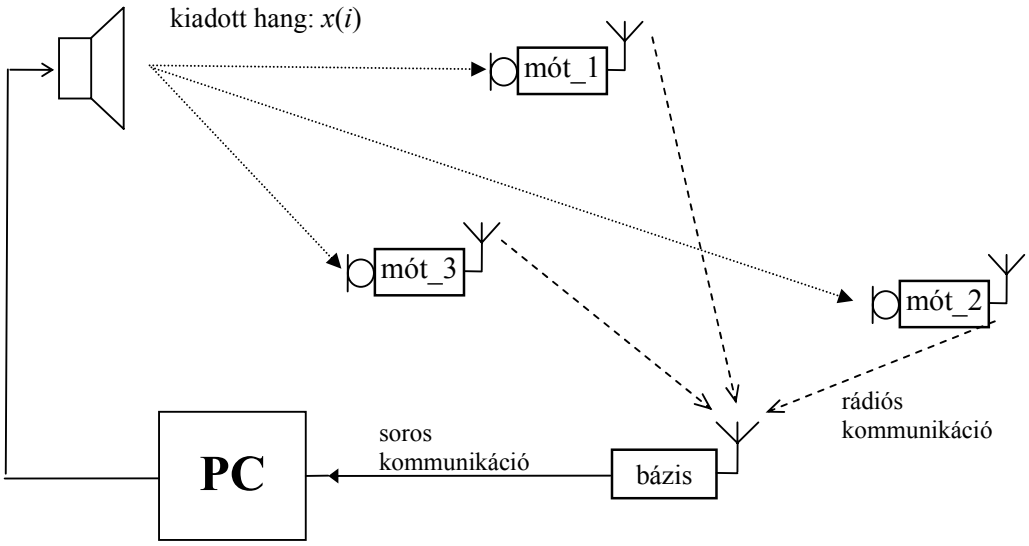
2. Akusztikus lokalizáció

2.1. A mérőrendszer felépítése

Az adott méréshez tartozó elrendezés az 1. ábrán látható. A mérés során egy hangszórón keresztül a PC hangkártyáját felhasználva egy hangmintát adunk ki. A hangmintát MATLAB segítségével állítjuk elő. A mótok a szenzorkártyájukon található mikrofon segítségével érzékelik a kibocsátott hangot, és a mótokon található mikrokontroller AD-átalakítóját használva 1800 Hz-es frekvenciával mintavételezik. A mintákat rádión keresztül folyamatosan továbbítják egy bázisállomás felé. Az adattovábbítás időosztásos (TDM) rendszerben működik: a mótok adott sorrendben egymás után továbbítják az adatokat.

A lokalizációs mérésben három mótot használunk. A rádión érkező adatokat egy bázisállomásként használt mót gyűjti össze, és soros porton továbbítja az adatot a PC felé, ahol a jelfeldolgozás történik. A PC-n egy adatgyűjtő program segítségével a bázisállomás által továbbított adatokat egy file-ba mentjük, és az itt található adatokat MATLAB segítségével dolgozzuk fel.

Mivel a mérésben az egyszerűség kedvéért csupán síkbeli lokalizációval foglalkozunk—tehát feltételezzük, hogy a hangforrás a mótok által meghatározott síkban található—így elegendő három darab szenzor használata. Térbeli lokalizációhoz ennél nagyobb számú szenzor szükséges.



1. ábra. A mérőrendszer felépítése

A lokalizáció alapgondolata az, hogy egy közös forrásból érkező hang különböző hosszúságú úton jut el az egyes szenzorokhoz. Amennyiben tudjuk mérni azt, hogy az egyes mótok mikor érzékelik a jelet, akkor meghatározható, hogy milyen időkülönbséggel érkezik meg a jel az egyes mótokhoz. Az érkezési idő különbségei (TDOA: Time Difference of Arrival) alapján megfelelő számú szenzor esetén meghatározható, hogy honnan jött az érzékelt jel, hiszen a hang terjedési sebességének ismeretében az időkülönbségekből a hangforráshoz viszonyított távolságkülönbség számítható. Az érkezési időpont önmagában azért nem jelent általában semmilyen információt, mert sokszor nem ismert a hang keletkezésének időpontja, tehát nincs mihez viszonyítani az időt.

Látható, hogy a rendszer működése szempontjából alapvető fontosságú a megfelelő időreferencia, hiszen az időkülönbségek (TDOA) mérésekor elengedhetetlen, hogy a szenzorok ugyanazon időponthoz tartozó mért adatokat továbbítsanak. Ha az egyes mintavételi időpontok között a különböző szenzorokon ΔT_d bizonytalanság van, akkor az időbeli adatok értelmezése is ilyen bizonytalansággal lehetséges. Ha a hang sebességét 340 m/s-nak tekintjük, akkor kiszámítható, hogy például 500 μs-os bizonytalanság a mintavételi időpontok között 17 cm-es hibát okozhat. Ennek kiküszöbölésére egy szinkronizációs algoritmus fut a mótokon. A mérés során a szinkronizációs algoritmus mélyebb ismerete nem szükséges, azzal a vezetéknélküli szenzorhálózatokhoz kapcsolódó másik mérésben ismerkedhetünk meg. A rendszer jobb áttekintése érdekében érdemes a 9. méréshez tartozó leírást is tanulmányozni.

A továbbiakban a lokalizáció két alapkérdésével foglalkozunk: hogyan határozható meg geometriailag egy hangforrás pozíciója a TDOA adatok alapján, és hogyan detektálható egy szenzoron egy adott akusztikus jel érkezési ideje.

2.2. A lokalizáció geometriai feladatai

Ebben az alfejezetben egy egyszerű lokalizációs feladat megoldásával ismerkedünk meg, és a megoldás alapján megfogalmazzuk a szenzorhálózat által teljesítendő követelményeket, illetve meghatározzuk a lokalizációhoz szükséges paramétereket, melyek mérését el kell végezni a szenzorhálózat és megfelelő jelfeldolgozási eszközök segítségével.

Tekintsük a 2. ábrát, melyen egy egyszerű lokalizációs feladat látható. A kör a hangforrást, a négyzetek a szenzorokat jelölik. Két szenzor esetében a hangforrás konkrét pozíciója nem határozható meg, a forrás iránya azonban igen. Az irányt az α -val jelölt beesési szög határozza meg. Az ábrán d_1 és d_2 az első és a második szenzormót hangforrástól mért távolságát jelöli. Tételezzük fel, hogy valamilyen módszerrel meghatároztuk, hogy egy adott hang milyen dt_{ij} időkülönbséggel érkezik meg az egyes szenzorokhoz. Mivel a hang sebessége ismert, ezért ez alapján meg lehet határozni az *i*-edik és a *j*-edik mót hangforrástól mért távolságának különbségét, melyet jelöljünk Δd_{ij}-vel:

$$\Delta d_{ij} = d_\mathrm{i} - d_\mathrm{j} = c \cdot dt_{ij}, \tag{1}$$

ahol c a hangsebességet jelöli. Szobahőmérsékleten számolhatunk $c = 340$ m/s-os sebességgel. d_i és d_j az *i*-edik és a *j*-edik mót hangforrástól mért távolságát jelöli.

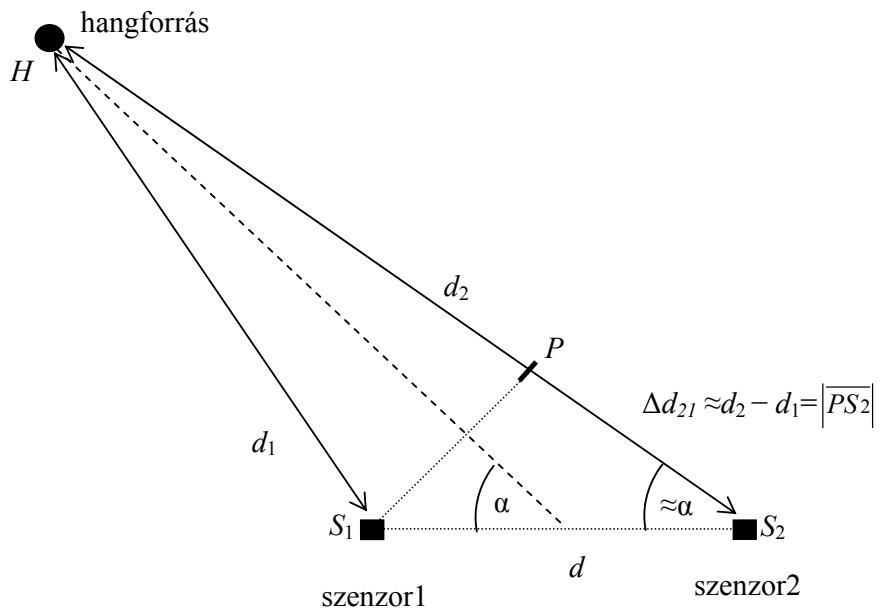
Az α beesési szög számításhoz azt a feltételezést használjuk, hogy a hangforrás elég messze van a szenzoroktól, így a forrást és a szenzorokat összekötő egyenesek egymással párhuzamosnak tekinthetők, tehát az ábrán S_2-vel jelölt csúcsnál található szög körülbelül megegyezik α-val. Ebből következik, hogy a P pontnál található szög körülbelül derékszög. Ezek után egyszerű trigonometriai összefüggést felhasználva számítható az α szög:

$$\cos(\alpha) = \frac{\Delta d_{ij}}{d}, \tag{2}$$

$$\alpha = \mathrm{acos}\left(\frac{\Delta d_{ij}}{d}\right). \tag{3}$$

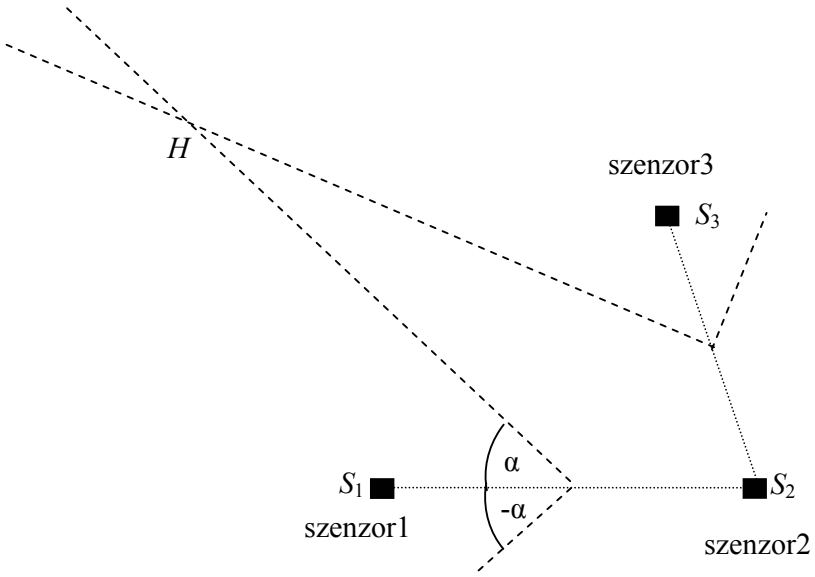
ahol d a szenzorok közötti távolságot jelöli, ezt előzetesen kell meghatározni. Ezzel tehát meghatároztuk a hangforrás irányát azzal a bizonytalansággal, hogy nem tudjuk, hogy az S_1S_2 egyenes bal vagy jobb oldalán van, hiszen ha tükrözzük a H pontot az S_1S_2 egyenesre, akkor is ugyanolyan Δd_{ij} mérhető. Matematikailag ez abból adódik, hogy a cos függvény páros, tehát $\cos(\alpha) = \cos(-\alpha)$, így a (3) egyenletnek két megoldása van: $\pm\alpha$.

A mérés hibaanalízisét elvégezve kiderül, hogy a mérés bizonytalansága annál nagyobb, minél kisebb az α szög, tehát amikor a hangforrás a szenzorok által meghatározott egyeneshez közel helyezkedik el.



2. ábra. Beesési szög meghatározása

A hangforrás pozíciójának meghatározásához felhasználhatunk több szenzor által becsült irányt a 3. ábrán látható módszerrel: páronként meghatározzuk a szenzorokra a hangforrás lehetséges irányát, és az így kapott félegyenesek metszete adja a becslést a forrás pozíciójára.

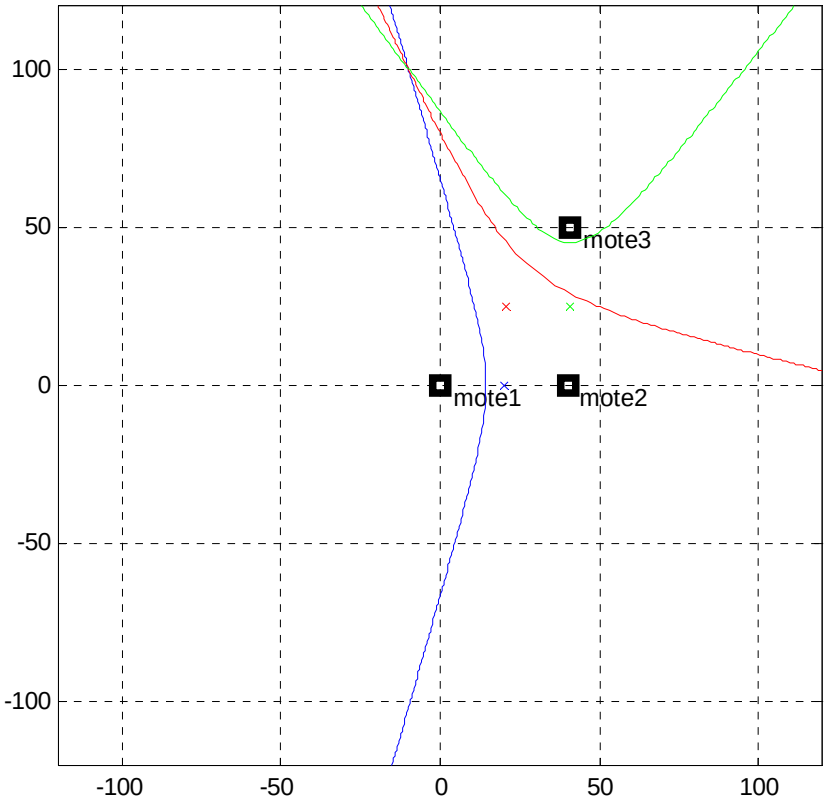


3. ábra. Pozícióbecslés iránymérés segítségével

Meg kell jegyeznünk, hogy a hangszóró lehetséges pozícióit valójában nem az ábrán szaggatott vonallal jelölt félegyenesek adják, hanem hiperbolák. Ez definícióból következik, hiszen a hiperbola azon síkbeli pontok halmaza, melyek két adott ponttól mért távolságának különbsége állandó. Esetünkben ez a két pont S_1 és S_2, a távolságkülönbség pedig Δd_{21}. A

félegyenesekkel való közelítés a forrás és a szenzorok megfelelő távolsága esetén használható.

A mérés során rendelkezésre áll két MATLAB függvény, melyek segítségével a mért távolságkülönbségek, illetve a mótok pozíciójának felhasználásával megjeleníthetjük a hangforrás lehetséges pozícióit. Ez mind a beesési szög módszerével, mind a bonyolultabb esetet tekintve lehetséges. A két függvény neve: `dispLocations` és `dispLocationsHiperb`. A függvényeknek paraméterként át kell adni a mótok pozícióját illetve a mért Δd_{ij} érkezési idő különbséget minden mótpárra. Az alábbi ábra a hiperbolákkal történő pozícióbecslést szemlélteti. Az egyes hiperbolák a hangforrás lehetséges pozícióját mutatják egy-egy mótpár esetén. A zöld görbe például a 2-es és a 3-as mót mérési eredménye alapján lehetséges pozíciókat szemlélteti. A görbék metszéspontja adja a forrás feltételezett pozícióját. A gyakorlatban az ábra vizuális kiértékelésével becsüljük a metszéspontot, de valós körülmények között a három görbe metszéspontja nem esik egybe a mérési hibák miatt, ekkor számíthatunk például a metszéspontok középpontjával, súlyozott átlagával.



4. ábra. Pozícióbecslés hiperbolákkal

Láthatjuk tehát, hogy a lokalizációt visszavezethetjük (3) és (1) felhasználásával időkülönbség mérésére: egy adott hangot (jelsorozatot) milyen időkülönbséggel érzékel két szenzor. Ezen probléma megoldására található válasz a következő részben.

2.3. Az érkezési idő detektálása

Az érkezési idő meghatározáskor abból indulunk ki, hogy a mérés során egy ismert $x(i)$ időfüggvényű hangot adunk ki hangszóró segítségével, majd a szenzorok által mért jelben szeretnénk megállapítani, hogy hol található a kiadott hangminta, vagy legalábbis a rá legjobban hasonlító rész. A feladat tehát egy ismert minta keresése egy adatfolyamban. Ebben az esetben abból a feltételezésből indulunk ki, hogy a mért $y(i)$ jel felbontható egy a kiadott $x(i)$ jelre hasonlító, és egy tőle teljesen függetlennek tekinthető $z(i)$ jelre, mellyel a mérési zajt, illetve egyéb, a környezetből származó akusztikus jelet veszünk figyelembe:

$$y(i) = x(i - T_d) + z(i). \tag{4}$$

T_d azt fejezi ki, hogy a keresett $x(i)$ jel nem a mérési regisztrátum elején kezdődik, hanem ahhoz képest valamilyen T_d idővel eltolva. A mérés célja ezen T_d meghatározása. A probléma megoldásához ismerkedjünk meg az úgynevezett keresztkorrelációs függvénnyel, melyet a következő képlet definiál mintavételezett jelekre:

$$R_{xy}(n) = \sum_{i=-\infty}^{\infty} x(i)y(i+n) \tag{5}$$

$R_{xy}(n)$ az x és y jel közötti keresztkorrelációs függvény értékét jelöli az n időpontban. A korreláció megértéséhez nézzük, hogyan is történik a kiszámítása, mit jelent az (5) képlet. Láthatjuk, a keresztkorreláció n-edik értékének kiszámításához az x jelet n mintával eltoljuk, és vesszük az y jellel vett szorzat összegét minden i időpontban a $(-\infty,\infty)$ intervallumban. Amennyiben az x és y jel nem hasonlít egymásra, úgy feltételezhető, hogy a szorzatuk körülbelül egyenlő valószínűséggel lesz pozitív vagy negatív, tehát ezen értékek összege várhatóan alacsony lesz. Amennyiben viszont a két jel hasonlít egymásra, és mindig egyszerre mozognak, akkor előjelük mindig megegyezik (tehát szorzatuk pozitív), így a szumma értéke a sok pozitív szám összege miatt nagy lesz. Mivel ezt az összegzést minden n mintával való eltolásra elvégezzük, így a korreláció egy olyan időfüggvény, mely megmondja, hogy két jelet egymáshoz képest fokozatosan eltologatva mennyire hasonlítanak egymásra. Egy jel korrelációját önmagával is kiszámíthatjuk, ezt autokorrelációs függvénynek nevezzük:

$$R_{xx}(n) = \sum_{i=-\infty}^{\infty} x(i)x(i+n) \tag{6}$$

Ez tranziens jelek esetében általában egy olyan $R_{xx}(n)$ függvény, melynek $n = 0$-ban van a maximuma, hiszen általában egy jel önmagára, és nem bármilyen eltoltjára hasonlít legjobban.

Lássuk, hogy az eddigieket hogyan használhatjuk fel a konkrét lokalizációs rendszerben, ahol egy ismert jelsorozatot kell keresni. Számítsuk ki a (4)-ben megadott $y(i)$ mért jel keresztkorrelációját a kiadott $x(i)$ jellel (5) alapján, és helyettesítsük be $y(i)$ értékét:

$$R_{xy}(n) = \sum_{i=-\infty}^{\infty} x(i)\left[x(i - T_d + n) \; + \; z(i+n)\right] \tag{7}$$

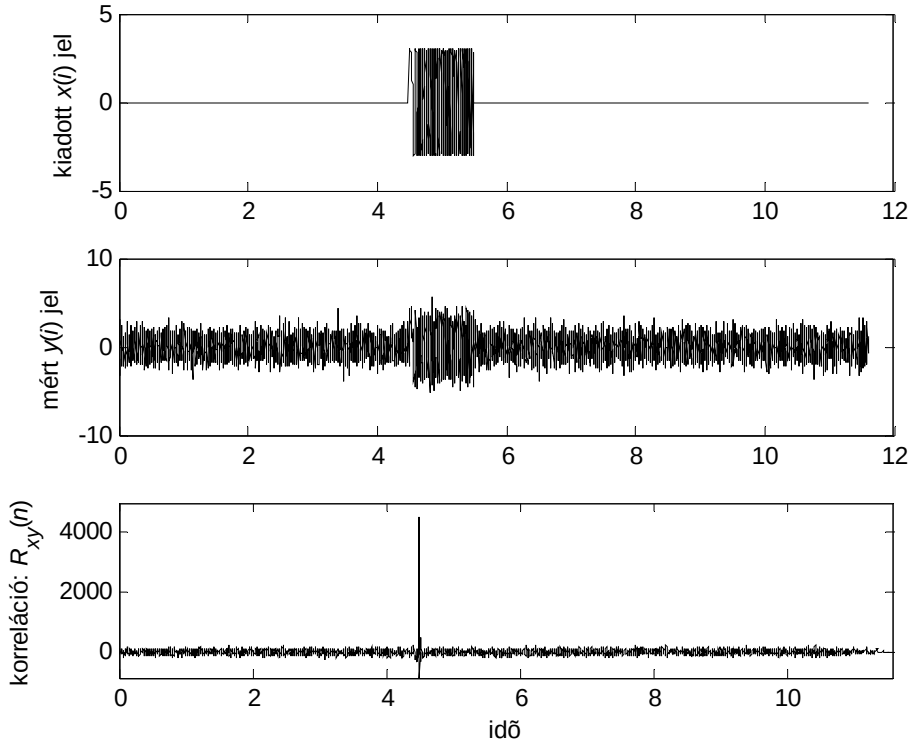
Felbontva a szummában található zárójelet, a következőt kapjuk:

$$R_{xy}(n) = \sum_{i=-\infty}^{\infty} x(i)x[i + (n - T_d)] + \sum_{i=-\infty}^{\infty} x(i)z(i+n) \tag{8.a}$$

$$R_{xy}(n) = \sum_{i=-\infty}^{\infty} x(i)x[i + (n - T_d)] = R_{xx}(n - T_d) \qquad (8.b)$$

Az átalakítás során kihasználtuk azt a feltételezést, hogy $z(i)$ nem függ $x(i)$-től, tehát keresztkorrelációjuk elhanyagolható. Láthatjuk, hogy ha a mért jel és a kiadott jel korrelációs függvényét kiszámítjuk, akkor a kiadott jel autokorrelációs függvényét kapjuk T_d-vel eltolva. Mivel az autokorrelációs függvénynek általában nullában van a maximuma, így T_d-vel eltolva T_d-be kerül a maximuma. A maximumhely megkeresésével tehát meghatározható T_d.

Az elmondottak szemléltetésére az 5. ábrán egy szimuláció eredménye látható. A felső ábra a kiadott $x(i-T_d)$ jel időfüggvényét mutatja. A jel 4.5 sec-ban kezdődik, tehát $T_d = 4.5$, a jel hossza pedig 1 sec. A középső ábrán a mért jel időfüggvénye található. A mért jelet zaj terheli. Látható, hogy habár valamelyest detektálható az időfüggvényben a jel kezdete, éles átmenet nem tapasztalható, a jel-zaj viszony igen rossz.



5. ábra. Korrelációs függvény demonstrációja

Az 5. ábra harmadik időfüggvényén az (5) egyenlet alapján számított korrelációs függvény látható. A korrelációs függvényben található csúcs az $x(i)$ mintasorozat helyét jelöli a mért $y(i)$ jelben. Mivel ez a csúcs igen domináns, így a detektálása is egyszerű. A csúcstól távolabbi részeken a korrelációs függvény értéke igen kicsi, hiszen a mérési zaj jó közelítéssel függetlennek tekinthető a kiadott $x(i)$ jeltől.

A csúcs detektálásának pontosságát ideális esetben is korlátozza a mintavételi frekvencia. Jelölje a mintavételi frekvenciát f_s, így a mintavételi időköz: $T_s=1/f_s$. Mivel a korreláció számításakor egész számú mintavételekkel toljuk el és hasonlítjuk össze a jeleket, így a hasonlóság detektálása is ilyen felbontással lehetséges. A mintavételi frekvencia tehát a következőképpen korlátozza a detektálás pontosságát:

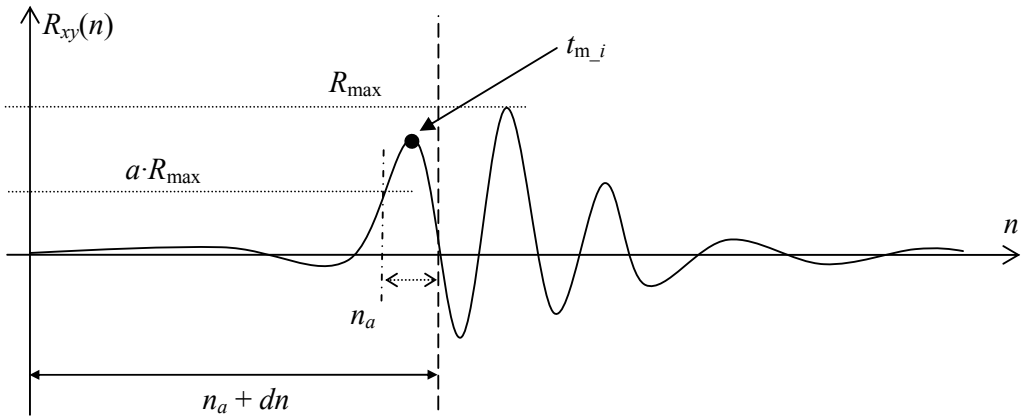
$$h_T = T_s = \frac{1}{f_s}, \tag{9}$$

ahol h_T a mérés abszolút hibája. A pontosság javításához ezért minél nagyobb mintavételi frekvencia szükséges. Előfordulhat azonban, hogy nem használható tetszőlegesen nagy mintavételi frekvencia. Például a jelenlegi rendszerben az adatátvitelre használt rádiós csatorna sávszélessége korlátozza a mintavételi frekvenciát, hiszen a mintavételezett jeleket valós időben mindhárom szenzor felől, folyamatosan továbbítani kell a bázisállomás felé. Erre a problémára jelentenek megoldást különféle interpolációs módszerek, melyek segítségével a mintavételezett jel függvénye előállítható a mintavételi pontok között is, tehát javul az időbeni felbontás. Erre ad egy módszert a 9. mérésben bemutatott interpolációs algoritmus, mely segítségével a mintavételi frekvencia tetszőleges egészszám-szorosára növelhető. A mérés során is ezt az algoritmust alkalmazzuk, ezért szükséges annak megismerése. Az algoritmus leírása megtalálható a 9. mérés útmutatójában, illetve függelékként ezen mérési útmutató végén. A mérés során nem kell az interpolációs módszert implementálni, ahhoz kész függvény áll rendelkezésre, melynek leírása szintén a mérési utasítás függelékében található. A továbbiakban feltételezzük, hogy a mintavételi frekvencia megfelelő mérési pontosságot tesz lehetővé.

Láthattuk, hogy ideális esetben a jel érkezési idejének meghatározásához elegendő a korrelációs függvényben látható csúcs pozíciójának meghatározása, mely sok esetben egy egyszerű maximumkereséssel elvégezhető. A valóságban azonban problémát okoz, hogy a hangszórón kiadott jel nem csak egyenes vonalú terjedéssel jut el a mikrofonokig, hanem többutas terjedés révén egyazon jelsorozat többször is, visszhangszerűen megjelenik a mért jelben. További probléma, hogy a kiadott hang a PC hangkártyájától egészen a mótokon végzett mintavételezésig több lineáris és nemlineáris torzítást is szenvedhet. Mindezen jelenségek a korrelációs függvényben további csúcsok megjelenését okozzák. A lokalizáció szempontjából csupán az első csúcs hasznos, ugyanis az jelenik meg az egyenesvonalú terjedés hatására, mely a hangszórótól mért távolságról hordoz információt. Sok esetben ez az első csúcs a legnagyobb, így egyszerű maximumkereséssel meghatározható az érkezési időpont. Előfordulhat olyan eset is, amikor egy következő lokális maximumhely magasabb, mint az első maximum. Ezt az esetet mutatja be a 6. ábra, amelyen a korrelációs függvénynek a függvény maximuma körüli kinagyított képe látható. Megállapíthatjuk, hogy a példában a második csúcs magasabb, mint az első csúcs, így egyszerű maximumkeresés nem alkalmazható az érkezési idő detektálására. Erre az esetre egy igen egyszerű algoritmus a következő:

- Határozzuk meg a korrelációs függvény maximumát. A maximum értéke R_{max}.
- Keressük meg azt a helyet, ahol a korrelációs függvény először túllépi az R_{max} érték a-szorosát. Legyen ezen pont pozíciója n_a. a megválasztása tapasztalati úton történhet, laboratóriumi körülmények között körülbelül 0.5-0.6 körüli érték lehet kiindulási alap.
- A következő lépésben a maximumot már csak a $[0 \ldots n_a+dn]$ intervallumban keressük. Az így megtalált csúcs tekinthető az egyenes vonalú terjedés hatására kialakult csúcsnak. dn megválasztása a kiadott jeltől függő paraméter, néhány tipikus mérési eredmény megtekintése után tapasztalati úton is meghatározható, de a későbbi részekben megadunk egy eljárást a becslésére. Fontos, hogy ne legyen nagyobb, mint az $x(i)$ jel autokorrelációs függvény főhullámának szélessége.

Ha ez az algoritmus sem eredményes, szemrevételezéssel kell megoldani az első maximum megkeresését.



6. ábra. Érkezési idő detektálása

A lokalizációs feladat első lépéseként tehát az eddigiekben leírt módszer segítségével minden egyes mót által mért jelre el kell végezni a korrelációszámítást az (5) képlet segítségével, ahogyan az az 5. ábrán is látható. Korrelációszámításra MATLAB-ban az `xcorr` függvény használható. A korrelációt a mótok által mért regisztrátumok között, illetve a MATLAB-ban generált gerjesztőjel sorozattal kell elvégezni, amelyet aztán a hangszóróra kiadunk.

Legyen például a MATLAB-ban generált és a hangszóróra kiadott mintasorozat (vektor) az `xg` változóban, a mótok által érzékelt, és a megfelelő mintavételi frekvenciára felinterpolált adatsorozat pedig egy három sorból és *K* darab oszlopból álló `moteDat` nevű mátrixban elhelyezve. A `moteDat` mátrix *i*-edik sorában tároljuk a *i*-edik mót által érzékelt jeleket, minden móthoz *K* darab érték tartozik. Ebben az esetben az *i*-edik mót, és a kiadott jel közötti korrelációja az `xg` vektor és a `moteDat` mátrix *i*-edik sorának korrelációjaként számítható, mely MATLAB-ban a következő művelettel végezhető el:

```
koor=xcorr(moteDat(i,:),xg);.
```

A korrelációs függvényben található első jelentős lokális maximum (első csúcs) helye megadja az *i*-edik móton a várt hangminta érkezési idejét, mely időt jelöljön t_{m_i}, ahol *i* az *i*-edik mótot jelöli. A mérés során nem közvetlenül használjuk fel ezt az eredményt, hiszen t_{m_i} az érkezési időnek csak az adott regisztrátumon belüli pozícióját adja meg, a mérés kezdete viszont bizonytalan lehet. Emiatt a $dt_{\_ij}$-vel jelölt érkezési időkülönbségeket határozzuk meg minden *i-j* mótpárra:

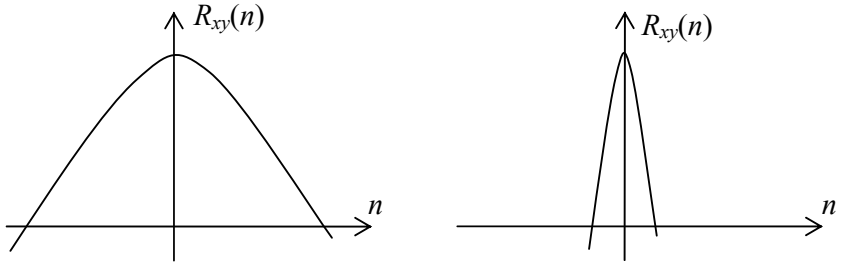
$$dt_{\_ij} = t_{m\_i} - t_{m\_j.} \tag{10}$$

Ez az eredmény már értelmezhető, hiszen az biztosítható, hogy a mótokon egyszerre kezdődjön a mérés, tehát ugyanahhoz a ponthoz viszonyítva mérjük t_{m_i}-t. Persze ehhez is szükséges a megfelelő szinkronizáció biztosítása. Az előző alfejezetben tárgyalt eljárásokban ezen $dt_{\_ij}$ érték már közvetlenül felhasználható: (1) és (3).

Mivel a mérés során végrehajtott keresztkorreláció számításában alapvetően a kiadott gerjesztőjel autokorrelációs függvénye jelenik meg, ezért áttekintjük a gerjesztőjel választásának különböző vonatkozásait.

2.4. Gerjesztőjel megválasztása

Az előző alfejezetben megismert módszer azon alapszik, hogy korrelációs technikával megkeresünk egy $y(i)$ mért jelsorozatban egy ismert $x(i)$ mintát. Ahol a minta található a jelsorozatban, ott a keresztkorrelációs függvénynek maximuma van, és (8.b) alapján beláttuk, hogy itt az autokorrelációs függvény jelenik meg ideális esetben. A pontos méréshez az szükséges, hogy ez a maximum minél keskenyebb legyen, hiszen ekkor a mérési zaj kevésbé befolyásolja a maximum pozícióját. Ezt szemlélteti a 7. ábra, ahol két korrelációs függvény főhullámát ábrázoltuk. Látható, hogy a jobb oldali, csúcsosabb korrelációs függvényen a maximum egyértelműbben detektálható, és kisebb a valószínűsége, hogy a mérési zaj miatt a korrelációs függvényben megjelenő ingadozások nem változtatják meg a maximum pozícióját.



7. ábra. Különböző autokorrelációs függvények

A minél pontosabb maximumhely-detektáláshoz tehát az szükséges, hogy a kiadott $x(i)$ jel autokorrelációs függvénye minél csúcsosabb, más szóval minél keskenyebb legyen. A korrelációs függvény szélességének meghatározásához vizsgáljuk meg újra az autokorreláció (6)-tal megadott definícióját. (6) alapján megállapítható, hogy a korreláció számítása gyakorlatilag az $x(i)$ és $x(-i)$ jel konvolúciójaként áll elő. $x(-i)$ az $x(i)$ időfüggvény tükörképe. Mivel az időtartománybeli konvolúció frekvenciatartományban szorzásnak felel meg, ez azt jelenti, hogy a korrelációs függvény Fourier-transzformáltjának számításához elegendő az $x(i)$ jel Fourier-transzformáltjának abszolút négyzetét venni:

$$\mathcal{F}\{R_{xx}(n)\} = \mathcal{F}\{x(i)\}\mathcal{F}\{x(-i)\} = |X(f)|^2, \tag{11}$$

ahol $\mathcal{F}$ a Fourier-transzformációt jelöli, $X(f)$ az az $x(i)$ jel Fourier-transzformáltja. (11) alapján megállapítható, hogy az autokorrelációs függvény Fourier-transzformáltja a jel teljesítménysűrűség-spektrumát adja, hiszen ha vesszük egy adott frekvencián a jel amplitúdójának négyzetét, az a teljesítményt adja meg az adott frekvencián. Mindez azért érdekes, mert tudjuk, hogy minél szélesebb egy jel frekvenciatartományban, annál keskenyebb időtartományban. Ahhoz tehát, hogy az autokorrelációs függvény minél keskenyebb legyen, szélessávú gerjesztőjelet kell alkalmaznunk. Erre a kritériumra természetesen többféle jel is megfelelő. A mérések során az úgynevezett chirp jelet használjuk gerjesztőjelként. Ennek időfüggvényét a következő egyenlet adja meg:

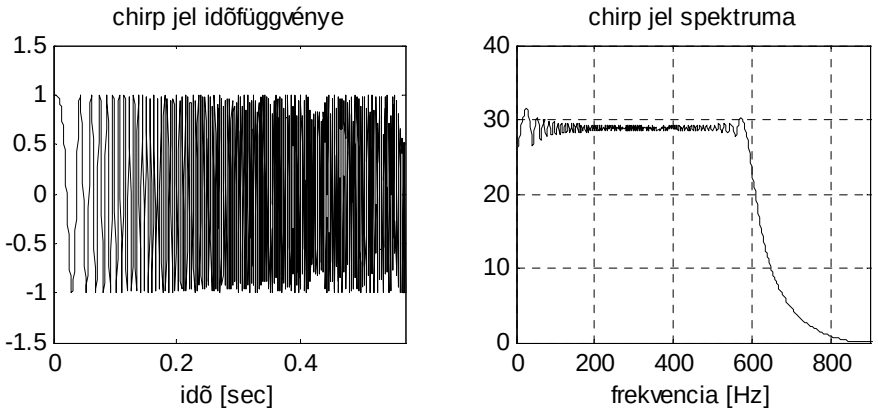
$$x(t) = A\sin\left[2\pi(f_0 + \frac{\Delta f}{2T}t)t + \varphi\right]; \quad t = [0...T] \tag{12}$$

A chirp jel tehát egy T hosszúságú, lineárisan növekvő frekvenciájú szinuszjel. A szinusz argumentumát idő szerint deriválva megkapjuk, hogy a jel frekvenciájának időfüggvénye:

$f(t) = f_0 + \frac{\Delta f}{T} t$, tehát a t=[0…T] értelmezési tartományában f_0-tól (f_0+Δf)-ig folytonosan változik a frekvenciája. Egy viszonylag jó közelítést jelent a chirp jel BW_{ch} sávszélességére, ha a jel $t = T$-ben felvett végső frekvenciáját vesszük:

$$BW_{\text{ch}} \approx f_0 + \Delta f \tag{13}$$

Egy $T = 0.6$ sec hosszúságú $f_0 = 0$ Hz és (f_0+Δf) = 600 Hz paraméterekkel rendelkező chirp jel időfüggvényét és spektrumát ábrázolja a 8.a. és 8.b. ábra. Látható, hogy a jel spektruma viszonylag egyenletes a [0 … BW_{ch}] intervallumban, bár látszik, hogy a sávszélességre adott BW_{ch} közelítés nem teljesen pontos.



8.a. ábra. *8.b. ábra*

8. ábra. Chirp jel időfüggvénye és spektruma. a: időfüggvény; b: spektrum

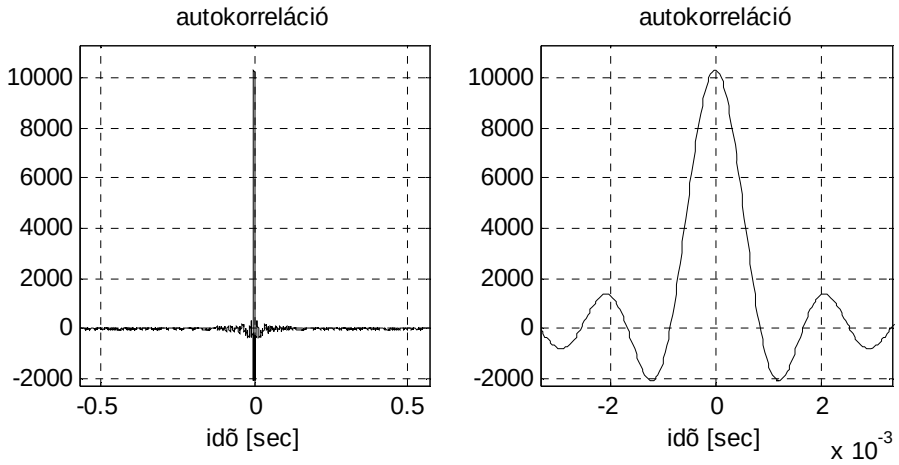
A 9. ábrán a chirp jel autokorrelációs függvénye látható (teljes és a főhullámra közelítve). A főhullám szélességére szintén adható egy nagyságrendileg megfelelő közelítés, ha a chirp jel teljesítménysűrűség-spektrumát egyenletesnek vesszük a [$-BW_{\text{ch}}$ … BW_{ch}] tartományban. (ne feledjük el, hogy a spektrum szimmetrikus 0-ra, ezért kell a negatív BW_{ch}-t is figyelembe venni). Ekkor a spektrum közelíthető egy négyszögablakkal, melynek szélessége $2BW_{\text{ch}}$. Mivel a négyszögjel inverz Fourier-transzformáltja sinc függvény, így az autokorrelációs függvény a következő módon közelíthető:

$$R_{xx}(t) \approx 2A \cdot \frac{\sin(2\pi \cdot BW_{\text{ch}} t)}{2\pi \cdot BW_{\text{ch}} t} = 2A \cdot BW_{\text{ch}} \operatorname{sinc}(2\pi \cdot BW_{\text{ch}} t) \tag{14}$$

Az autokorrelációs függvény főhullámának $2\Delta T$ szélessége meghatározható az $R_{xx}(\Delta T) = 0$ feltételből adódóan:

$$\Delta T = \frac{1}{2BW_{\text{ch}}} \tag{15}$$

Ezzel választ kaptunk a 6. ábrán használt *dn* meghatározására, hiszen azt a főhullám szélességének megfelelően kell megválasztani.



9.a. ábra. *9.b. ábra*

9. ábra. Chirp jel autokorrelációs függvénye

A chirp jel megfelelően nagy és jól kézben tartható sávszélessége miatt alkalmas keskeny autokorrelációs függvénnyel rendelkező jelek megvalósítására, mely az érkezési idő detektálása miatt fontos. Emiatt a mérések alkalmával is ezzel a jellel végzünk kísérleteket.

A gerjesztőjel előállításához kapcsolódó gyakorlati tudnivaló, hogy MATLAB-ban a chirp jel generálására a `chirp` nevű függvény használható. A MATLAB `help chirp` parancsával részletes leírást kaphatunk a használatáról. A jel paramétereit úgy érdemes megválasztani, hogy a feldolgozáskor ne keletkezzen túl sok minta. Ez néhány tized másodpercnyi, illetve egy másodperc nagyságrendnyi hosszúságot jelent. A chirp jel maximális frekvenciáját úgy kell megválasztani, hogy teljesítse a mótok mintavételi frekvenciája mellett a mintavételi tételt, tehát az 1800 Hz-es mintavételi frekvencia miatt ne legyen nagyobb 900 Hz-nél, viszont azért, hogy megfelelően keskeny legyen az autokorrelációs függvény, minél nagyobb sávszélességet kell beállítani.

A chirp függvénnyel generált jel a PC hangkártyája segítségével adható ki a hangszóróra. Ehhez a MATLAB `sound` illetve `soundsc` függvényei használhatóak. A függvénynek meg kell adni a kiadandó jelet illetve a hozzátartozó mintavételi frekvenciát. Vigyázat! A `sound` függvény esetén lejátszott jel amplitúdója [−1…+1] intervallumban kell, hogy tartózkodjon, ezen tartományon kívül „levágja” a jelet. A `soundsc` függvény automatikusan skálázza a selet. Részletes információért olvassuk el a MATLAB `help sound` parancsa által megjelenített információkat.

3. A lokalizációhoz felhasznált eszközök

A mérés során a mitmótokat használjuk az akusztikus jel érzékelésére. Mind a szenzor, mind a bázisállomás programja be van töltve a mótok programmemóriájába, a mérés során a mótokat nem kell átprogramozni, a különféle funkciók között a mótokon található gombok (SW) és kapcsolók (K) segítségével lehet választani. Kissé félrevezető lehet, de a mót I/O paneljén a gombok SW-vel vannak jelölve, így a félreértések elkerülése végett itt is ezt a jelölést alkalmazzuk. A megvalósított funkciók a következők:

Szenzorok:

K1:

ON állásban be van kapcsolva a szinkronizáció, ekkor a 9. méréshez tartozó útmutatóban leírt szinkronizációs algoritmus fut a mótokon. Minden egyes mót és a bázisállomás is az 1-es számmal jelölt móthoz szinkronizálódik, így az 1-es számú móton a szinkronizáció nem állítható, hiszen az a szinkronizációs referencia. Mint láttuk, a mintavételezés szinkronizálása az érkezési idők egymáshoz viszonyított pozícióinak meghatározása miatt fontos, ezért a K1 ***ON*** állásban legyen, kivéve, amikor a szinkronizálatlanság hatását vizsgáljuk.

OFF állásban nincs szinkronizáció.

K4: az AD-átalakítás forrását választhatjuk ki.

OFF állásban a szenzorkártya Line-In bemenetét mintavételezi.

ON állásban a mikrofon jelét mintavételezi a szenzor. Mivel mérés során akusztikus jeleket mérünk, így ***ON*** állásba kell kapcsolni. Mérés során az MK2-vel jelölt mikrofon jelét használjuk.

A hálózat működését alapvetően az 1-es számú mót vezérli. A mintavételezett hangjeleket tartalmazó rádiós csomagokat ugyanis időosztásos hálózati működéssel továbbítják a mótok a bázisállomás felé. Ez az időosztásos működés periodikus, és olyan hálózati periódusokból áll, melyek során minden mót adott sorrendben továbbítja az addig összegyűjtött új adatokat. Egy-egy hálózati periódus elejét az 1-es számú mót rádiós üzenete jelzi, és ahhoz szinkronizálódik a többi mót, illetve az határozza meg mikor küldhetik a következő adatcsomagot. A hálózat működéséhez az 1-es mót bekapcsolása mindenképpen szükséges.

Bázisállomás:

K1:

ON állásban be van kapcsolva a szinkronizáció, ekkor a 9. méréshez tartozó útmutatóban leírt szinkronizációs algoritmus fut a bázisállomáson, és a PC felé történő adattovábbítása szinkronizálódik az 1-es számú szenzor mintavételezéséhez. Ez azért szükséges, mert ha gyorsabban vagy lassabban történne az adatok továbbítása, mint ahogyan azok a szenzorhálózat felől rendelkezésre állnak, akkor vagy elfogyna (alulcsordulna) egy idő után a továbbítandó adat, vagy lassabb továbbítás esetén felhalmozódna (túlcsordulna).

OFF állásban nincs szinkronizáció.

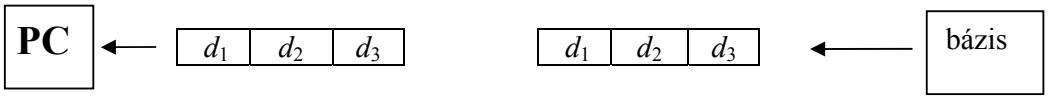
Ebben a mérésben a K1 kapcsolót ***ON*** állásban kell tartani.

SW2 / SW3:

A gombokat megnyomva leállíthatjuk / elindíthatjuk a bázisállomás soros porton történő adattovábbítását. Normál működés során nem kell használni.

Amennyiben rendellenes, nem várt működést tapasztalunk, a mótokon található reset gomb megnyomásával újraindítható az adott mót.

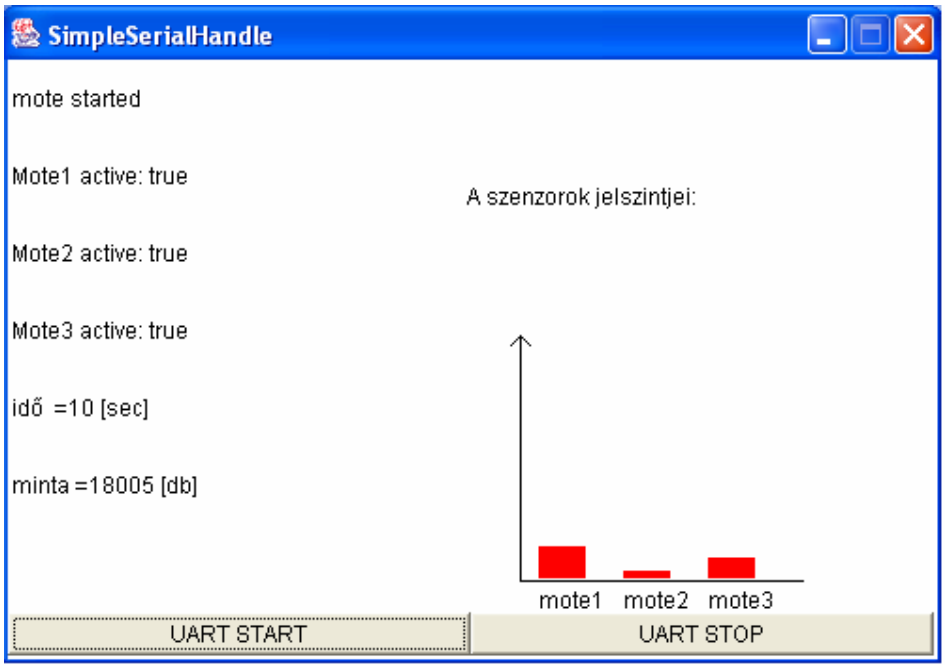
A bázisállomás a következő formátumban továbbítja az adatokat soros porton keresztül:



10. ábra. A PC és bázisállomás közötti kommunikáció

A bázisállomás a szenzor mótok 1.8 kHz-es mintavételi frekvenciájával megegyező ütemben 3 byteos adatokból álló csomagokat továbbít soros porton a PC felé. Az első d_1 byte az 1-es, a d_2 byte a kettes a d_3 byte a hármas mót aktuális hangmintáját tartalmazza. Amennyiben nem érkezik meg időben egy adott szenzortól az adat, a bázisállomás úgy nullákat továbbít az adat helyett. Ez történhet például rádiós csomagok sérülésekor.

A soros porton érkező adatok tárolásához a SimpleCollect nevű program használható. Ennek kezelői felülete a 11. ábrán látható. Mielőtt elindítjuk a programot, kapcsoljuk be a bázisállomást, és csatlakoztassuk PC-hez soros porton. A program indítás után leállítja a bázisállomáson az adatküldést, ezt az UART START gomb megnyomásával indíthatjuk el ismét, amint minden készen áll a mérésre. Az UART STOP gombbal leállítjuk a kommunikációt és kilépünk a programból. A program megjeleníti az egyes szenzorokhoz tartozó jelszinteket, illetve hogy „aktív”-e a mót, tehát érkezett-e tőle adat. Ezen funkció segítségével a mérés kezdetén lehetőség nyílik annak ellenőrzésére, hogy működőképes-e a rendszer.



11. ábra. Adatok mentését végző program

A program a beérkezett adatokat két file-ba menti. Az egyik file a programot tartalmazó könyvtár [motedata] könyvtárában található mic.dat file, melyben mindig a legutóbbi adatsorozat érhető el. A [motedata/Backups] könyvtárban minden mérés eredményét megtalálhatjuk mic_*dátum*.dat formátumban. Ez fontos lehet, ha esetleg véletlenül újra elindítjuk a programot, mielőtt befejeztük volna a file feldolgozását (ekkor ui. felülíródik a régi file), illetve érdemes az adott méréshez tartozó file-okat elmenteni, így jegyzőkönyv írásakor ezek alapján reprodukálhatóak az eredmények. Mindkét file-ban az egymás után érkező adatok a következő formátumban találhatóak meg a file-ban:

d_1, d_2, d_3
d_1, d_2, d_3
d_1, d_2, d_3
d_1, d_2, d_3
…

ahol az adott d_i értékek az adott pillanatban az i-edik mót által küldött értéket jelöli. Ezek időben összetartozó adatok.

Összefoglalva a lokalizáció során elvégzendő feladatok:

1. Indítsuk el az adatgyűjtést a PC-n.
2. Állítsuk elő a megfelelő gerjesztőjelet MATLAB-ban és sugározzuk ki a PC-hez csatolt hangszóró segítségével.
3. Amint megtörtént a hangminta lejátszása, állítsuk le az adatgyűjtést. Ezen műveleteket viszonylag gyorsan egymás után kell végezni, nehogy túl sok minta keletkezzen, ez ugyanis az adatfeldolgozás idejét nagyban befolyásolja.
4. Olvassuk be MATLAB-ban az adatfile-ba mentett információkat. Érdemes minden mérés során megjeleníteni az időfüggvényeket, hogy ki tudjunk szűrni bizonyos hibákat (pl. túl sok csomag veszett el…).
5. Amennyiben túl hosszú a regisztrátum, válasszuk ki a hasznos részt.
6. Végezzük el a mért jel interpolációját a megadott módszer segítségével. A gerjesztőjel előállításakor mindenképpen olyan mintavételi frekvenciát kell használni, hogy az egész számú többszöröse legyen a mótok mintavételi frekvenciájának, hiszen csak így lehet egész arányú interpolációval a mintavételezett jelet azonos mintavételi frekvenciára interpolálni. A mért és a kiadott jel összehasonlítása csak megegyező mintavételi frekvencián lehetséges. Például jó választás lehet a 36 kHz-es mintavételi frekvencia a hangkártya esetén, mert egy 20-szoros interpolációval a mótok 1800 Hz-es mintavételi frekvenciával gyűjtött adatsorozata egyszerűen felinterpolálható 36 kHz-re.
7. Végezzük el a korrelációszámítást.
8. Keressük meg a korrelációs függvényben található első csúcsok pozícióját, és mótpáronként határozzuk meg a csúcsok távolságának különbségét, tehát az érkezési idők különbségeit (TDOA).
9. A TDOA alapján számítsuk ki a mótok forrástól vett távolságának különbségét.
10. A távolságkülönbségek alapján végezzük el a forrás geometriai helyének meghatározását valamelyik ismertetett módszer segítségével.

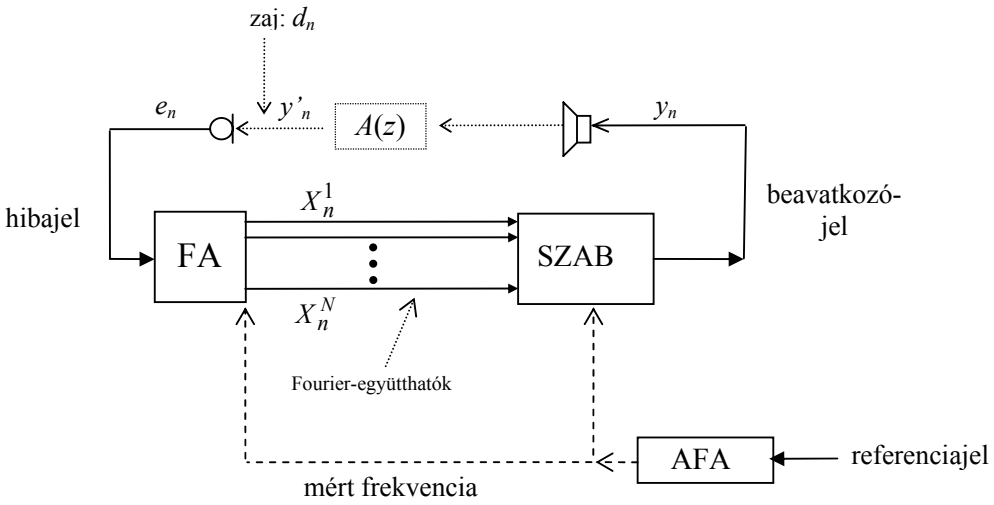
A sikeres mérés érdekében a felkészülés során érdemes a következő MATLAB-ban végzendő feladatokkal gyakorlatban is megismerkedni: megadott formátumú file beolvasása (pl. `fopen`/`fscanf`/`fclose` műveletek). Chirp jel generálása (`chirp` parancs). Mótok adatainak tömbös kezelése, változóstruktúrák megtervezése. Interpoláció áttekintése (csatolt file). Eredmények megjelenítése (`figure`/`subplot`/`plot…`). Korrelációszámítás (`xcorr`).

4. Aktív zajcsökkentő rendszer vizsgálata

4.1. Az aktív zajcsökkentés és a zajcsökkentő algoritmus bemutatása

Az aktív zajcsökkentő (ANC) rendszerek célja, külső akusztikus zajok elnyomása a zajjal ellentétes fázisú ellenzaj hangszórón keresztül történő kisugárzásával. Az aktív zajcsökkentő rendszerek az elnyomandó zajt mikrofonok segítségével érzékelik, majd az érzékelt zaj alapján a zajcsökkentő algoritmus által előállított ellenzajt hangszórók segítségével sugározzák ki, mely kioltja a zajt. A bonyolult algoritmusok implementálása a nagy számítási teljesítményt biztosító jelfeldolgozó processzorok (DSP) alkalmazásával lehetséges. A DSP-hez az érzékelő és beavatkozó elemeket analóg-digitális (AD) és digitális-analóg (DA) átalakítókkal csatlakoztatjuk. A számítás során gyakran felhasználunk egy olyan külső referenciajelet, melyből információt nyerünk az elnyomandó zajról (pl. frekvencia).

A mérés során egy úgynevezett rezonátoros zajcsökkentő rendszert vizsgálunk, melyet kifejezetten periodikus jelek elnyomására fejlesztettek ki. A zajcsökkentő algoritmus felépítése a 12. ábrán látható.



12. ábra. Aktív zajcsökkentő algoritmus

Az algoritmus alapja, hogy a mért jelet az ábrán FA-val jelölt úgynevezett Fourier-analizátor algoritmus Fourier-együtthatóira bontja, és ezek után harmonikusonként végezzük el a zajcsökkentést. A FA struktúra pontos ismerete nem szükséges a méréshez, leírása a függelékben található. Az algoritmusnak szüksége van egy referenciajelre, mely alapján a DSP méri a zaj frekvenciáját az ábrán AFA-val (Adaptív Fourier analizátor) jelölt blokk segítségével. A frekvencia ismerete a zaj Fourier-felbontása miatt szükséges. A zaj Fourier-együtthatóit felhasználva egy szabályozási algoritmus olyan beavatkozójelet állít elő, mely csökkenti a zaj amplitúdóját.

Az ábrán látható, hogy a DSP-n megvalósított szabályzó y_n kimenő jele, ami maga a kiadott ellenzaj, nem közvetlenül jut el a zajt érzékelő mikrofonhoz, hanem egy $A(z)$ átviteli függvénnyel jelölt úgynevezett másodlagos úton keresztül. $A(z)$ magában foglalja például a DA-átalakító, hangszóró, akusztikus út stb. átviteli függvényét is. Nyilván ezek hatással

vannak a kiadott jel paramétereire, tehát mire a kiadott ellenzaj a mikrofonhoz ér, akkorra megváltozik mind az amplitúdója, mind a fázisa, és az $A(z)$ átviteli függvénynek megfelelően módosított y'_n jel alakul ki a mikrofonnál:

$$y'_n = A(z)\, y_n\,, \tag{16}$$

Az ábrán látható módon az e_n hibajel a következő alakban írható fel, mivel e_n a zaj (d_n) és az ellenzaj szuperpozíciójaként alakul ki:

$$e_n = d_n + y'_n\,, \tag{17}$$

A teljes zajcsökkentéshez nulla zaj tartozik, tehát $e_n = 0$, ezért (17) miatt $y'_n = -d_n$. Behelyettesítve a (16) képletet, a következőt kapjuk:

$$y_n = A(z)^{-1}\, d_n, \tag{18}$$

$A(z)^{-1}$ az átviteli függvény inverzét jelöli, d_n pedig a zajt. Látható, hogy az átviteli függvény ismerete alapvető fontosságú. Amennyiben az átviteli függvényt nem ismerjük elég pontosan, akkor akár az is előfordulhat, hogy az ellenzaj erősíti a zajt, így a rendszer instabil lesz.

A szabályozó algoritmus nem közvetlenül a (18)-nak megfelelő jelet számítja és adja ki beavatkozójelként, hanem iteratívan módosítja a kimenetet mindig az aktuálisan mért zaj alapján. Jelöljük az e_n hibajel i-edik felharmonikushoz tartozó Fourier-együtthatóját az n-edik időpontban X_n^i-el (ld. 12. ábra). Ezek természetesen komplex számok, melyek abszolút értéke a mért zaj i-edik felharmonikusának amplitúdóját, szögük az i-edik harmonikus fázisát adja meg. Amennyiben a zajnak a mintavételi frekvenciához viszonyított relatív frekvenciája θ, akkor az i-edik harmonikus frekvenciája $\theta_i = i \cdot \theta$. A Fourier-együtthatókat felhasználva e_n jel a következő módon állítható elő:

$$e_n = \sum_{i=0}^{N-1} X_n^i e^{-j2\pi\theta i n}\,, \tag{19}$$

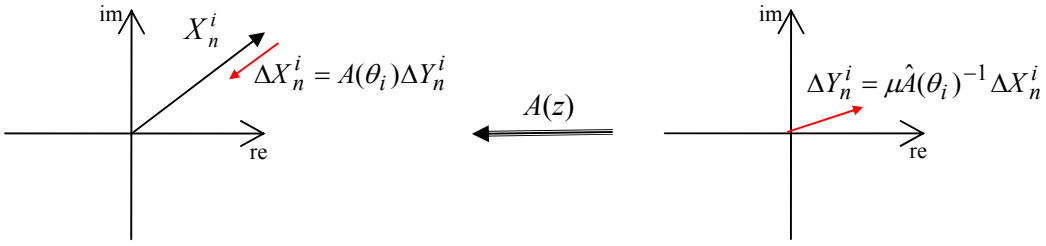
ahol N a Fourier-együtthatók számát adja meg. Hasonló jelölések mellett legyen az y_n beavatkozójel i-edik felharmonikushoz tartozó Fourier-együtthatója Y_n^i. Ezen jelölések használatával a mintavételi időpontokban végrehajtandó szabályozó algoritmus a következő módon írható fel:

$$Y_{n+1}^i = Y_n^i - \Delta Y_n^i = Y_n^i - \mu \hat{A}(\theta_i)^{-1} X_n^i\,, \tag{20}$$

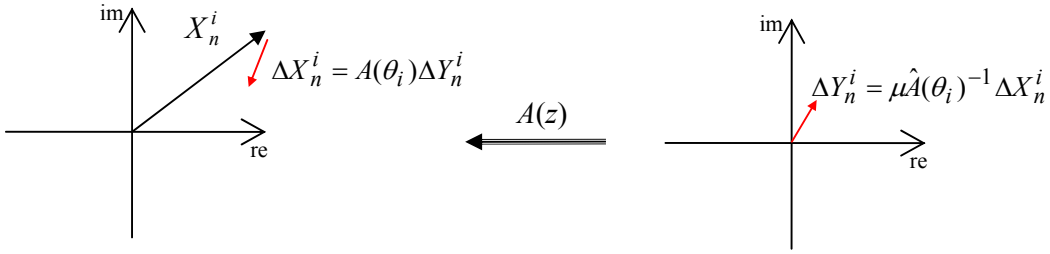
$\hat{A}(\theta_i)^{-1}$ az átviteli függvény inverzének becslőjét jelöli, és a másodlagos út átviteli függvénye által okozott hatás kompenzálása miatt jelenik meg. Azért használjuk a becslő kifejezést, ugyanis az átviteli függvény helyett annak valamilyen modelljét használjuk, mely a mérésből és modellezésből adódó hibákkal terhelt. Ideális esetben természetesen $\hat{A}(z)=A(z)$, így teljes mértékben, hiba nélkül figyelembe tudjuk venni a másodlagos út hatásait.

A (20) egyenlet által leírt algoritmus gyakorlatilag egy integrálásnak felel meg (mindig a változó korábbi értéket növeljük tovább), ez biztosítja a nulla hibát, feltéve, hogy stabil a rendszer. A μ paraméter egy bátorsági tényező, azt adja meg milyen léptékben módosítsuk az együtthatókat. Minél nagyobb annál gyorsabb a rendszer, de nagy μ esetén a módosítások túl nagyok lehetnek, így instabilitáshoz vezet. μ-vel gyakorlatilag a hurokerősítést lehet állítani.
(20) helyességének illusztrálásához tekintsük először a 13. ábrát. Láthatjuk, hogy helyes rendszermodell esetén a zaj amplitúdója a lehető legnagyobb mértékben csökken, hiszen a módosítás teljes mértékben a zajjal ellentétes irányú. Mivel a beavatkozójel változtatása arányos a hiba nagyságával, így az origóhoz (nulla zajhoz) közeledve egyre kisebb mértékben módosul a zaj, tehát beáll a nullába.

Az algoritmus bizonyos szintű hibát is elvisel az átviteli függvény becslésében. A 14. ábrán látható esetben $A(z)$-t valamilyen hibával ismerjük, emiatt az ábrán is látható módon a beállás során az ellenzaj nem teljesen önmagával ellentétes irányba módosítja a zajt, ennek ellenére folyamatosan csökkenti annak amplitúdóját. Ez a modellezési hiba a beállás lassulását okozza, hiszen nem a legrövidebb úton jut be a hibajel a nullába. A hibajel állandósult állapotban pontatlan $\hat{A}(z)$ esetén is nullába tart, mivel a hibajel mindig csökken. Amennyiben a valódi $A(z)$ és a becsült $\hat{A}(z)$ átviteli függvény fázisa között a fáziseltérés eléri a 90°-ot, akkor a módosítás mindig merőleges a zaj éppen aktuális Fourier-együtthatójára, így X_n^i elegendően kis módosítások esetén körpályán halad. A rendszer ennél nagyobb hibát nem képes elviselni. Amennyiben $A(z)$ és $\hat{A}(z)$ fázisa közötti hiba meghaladja a 90°-ot, a hiba folyamatosan nő, a rendszer instabillá válik.

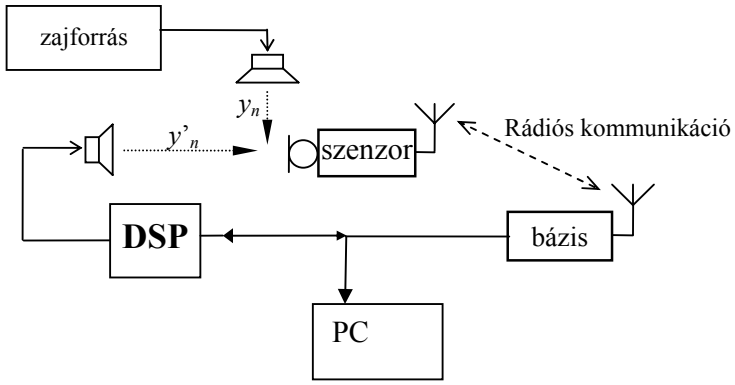


13. ábra. Zajcsökkentő algoritmus szabályzó algoritmusának működése helyes rendszermodell esetén: $\hat{A}(z)=A(z)$



14. ábra. Zajcsökkentő algoritmus szabályzó algoritmusának működése pontatlan rendszermodell esetén: $\hat{A}(z)\neq A(z)$

A mérés során használt zajcsökkentő rendszer különlegessége, hogy a zajérzékelést vezetéknélküli szenzor segítségével oldjuk meg. A rendszer blokkvázlata a 15. ábrán látható.

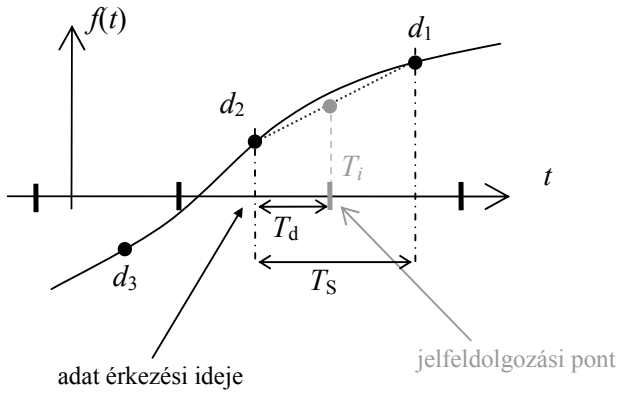


15. ábra. Aktív zajcsökkentő rendszer blokkvázlata

A zajérzékelő szenzorhálózat működése teljes egészében megegyezik a lokalizációnál használt rendszerével, a különbség csupán annyi, hogy egyetlen szenzor jelét használjuk föl. A szenzor tehát folyamatosan küldi rádión keresztül az adatokat, melyeket a bázisállomás továbbít a DSP felé. A PC a rendszer monitorozására szolgál.

A rendszerben szükséges a DSP és a bázisállomás működésének szinkronizálása, ugyanis a DSP 2 kHz-es sebességgel dolgozza fel az adatokat, míg a bázisállomás az 1.8 kHz-es frekvenciával mintavételezett jeleket küldi, tehát a minták érkezése és feldolgozása nem szinkronizált. Ennek kiküszöbölésére a DSP-n egy lineáris interpolációs módszer segítségével becsüljük a feldolgozandó jelet a jelfeldolgozási időpontokban. A lineáris interpoláció bemutatása a 9. mérésben olvasható. Röviden összefoglalva: a 16. ábrán látható módon a DSP a T_i-vel jelölt jelfeldolgozási időpontban a bázistól érkező utolsó két minta alapján (amelyek d_1 és d_2) becsülni tudja a feldolgozandó jel T_i-ben felvett értékét.

A rendszer alkalmas a működéshez szükséges $A(z)$ átviteli függvény mérésére is. A mérés során egy léptetett frekvenciájú szinusszal végigpásztázza a működési frekvenciatartományt. Identifikáció során a DSP a zajelnyomó hangszóróra egy nulla fázisú szinuszjelet ad ki, így a hibamikrofonon visszamért jel alapharmonikusának Fourier-együtthatója (amely X_n^1) éppen az átviteli függvény adott frekvencián vett értékét adja, hiszen a mért jel a másodlagos út által egy nulla fázisú, egységnyi amplitúdójú szinuszos jelre adott válasza.



16. ábra. Szinkronizáció lineáris interpolációval

4.2. Az aktív zajcsökkentő rendszer üzemeltetése

4.2.1. Fizikai kiépítés

A 17. ábrán a DSP-vel kapcsolatban álló eszközök összekapcsolásának módja látható.

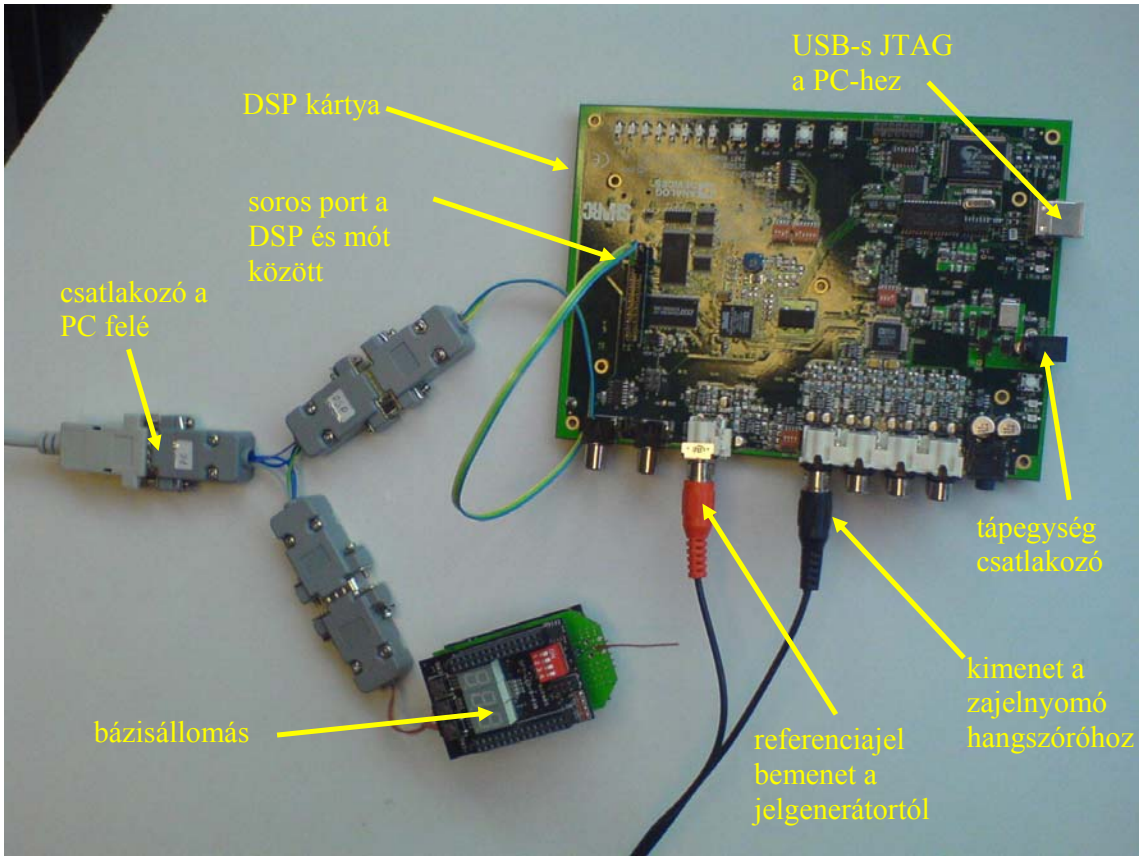
A jelfeldolgozó processzor egy ADSP-21364-es típusú 32 bites lebegőpontos processzor. A processzor végzi a jelfeldolgozási műveletek végrehajtását. A DSP a képen látható fejlesztői kártyán található. A fejlesztői kártya tartalmaz egy sztereo AD-és négy sztereo DA-átalakítót is. Az AD-átalakítót használjuk fel a referenciajel mintavételezésére. A DA-átalakító az ellenzaj kiadására szolgál. Ezt a DA átalakítót az ellenzajt kisugárzó hangszóróhoz kell csatlakoztatni.

A bázisállomás a DSP kártya felé soros porton küldi az adatokat. A speciális soros porti csatlakozót az ábrán látható módon kell csatlakoztatni a DAI jelzésű csatlakozósor 2-4-6 csatlakozójára: DAI2↔kék, DAI4↔sárga, DAI6↔zöld. Az összeköttetés megteremtése után reseteljük a bázisállomást. A kártya üzembe helyezéséhez adjunk tápfeszültséget a kártyának és az USB-s JTAG csatlakozóval kapcsoljuk össze a PC-t és a DSP kártyát.

Egy speciális csatlakozóval kössük össze a PC-t a DSP-t és a bázisállomást. Ennek segítségével a későbbiekben lehetőség lesz a PC-n a mért zaj Fourier-együtthatóinak megfigyelésére: 12. ábrán X_n^i értékek.

A Fourier-együtthatók közül az alapharmonikus X_n^1 együttható oszcilloszkópon is megjeleníthető. A DSP kártyán található 3.5-ös jack csatlakozót kössük egy oszcilloszkóp 1-es és 2-es csatornájához, és kapcsoljuk az oszcilloszkópot XY módba. Ekkor az X csatornán az érzékelt zaj alapharmonikusának valós része, az Y csatornán a képzetes része jelenik meg. A funkció főleg szemléltető és hibakereső eszközként alkalmazható, hiszen a valódi vektor megjelenítése nem lehetséges, ugyanis a DSP DA-átalakítója AC csatolású. Ehelyett felváltva $\pm X_n^1$ kerül kijelzésre, mivel ennek DC szintje eleve nulla.

A zajcsökkentő rendszer tesztelése során egy jelgenerátorral állítunk elő periodikus zajt. A generátor kimenetét egy hangszóróra kötve keletkezik az akusztikus zaj. A mérés során a jelgenerátor kimenetét a DSP referenciajelet érzékelő AD-bemenetére kell csatlakoztatni.

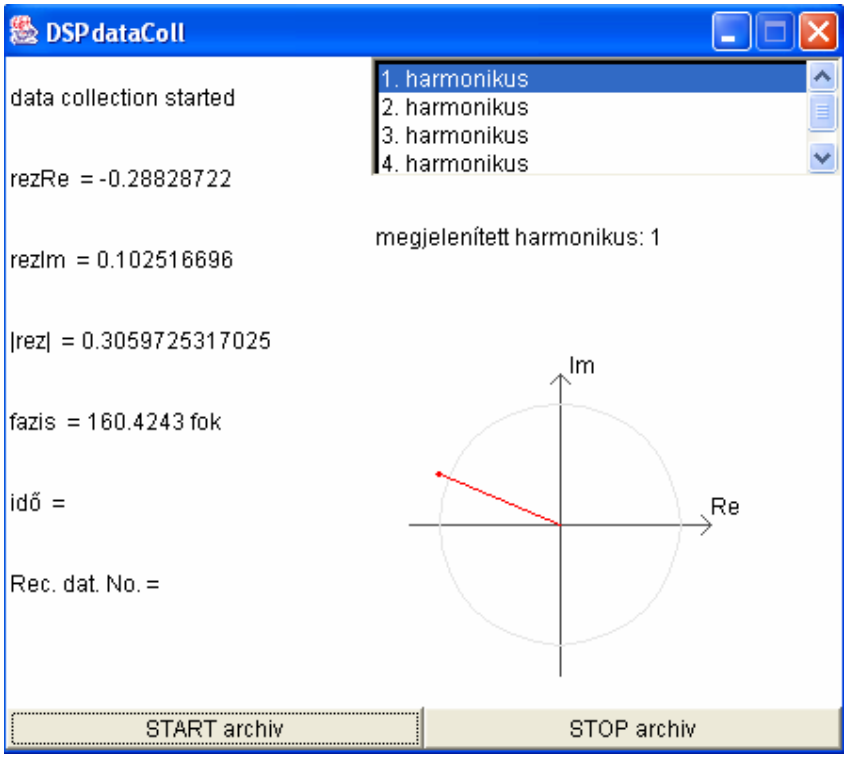


17. ábra. Zajcsökkentő rendszer összeállítása

4.2.2. Szoftverelemek

PC-s monitorozó program

Rendelkezésre áll egy PC-re írt alkalmazás, mely segítségével megjeleníthetőek működés közben a 12. ábrán X_n^i-vel jelölt értékek, melyek a hibajel Fourier-együtthatói. Ezen program segítségével tehát működés közben tanulmányozható, hogyan változik zajelnyomás közben a zaj fázisa és amplitúdója. Ezeket az információkat a DSP küldi a PC felé soros porton keresztül. A program neve DSPdata.



18. ábra. PC-s monitorozó program kezelői felülete

DSP-hez tartozó fejlesztőrendszer

A DSP kártya működtetése a VisualDSP++ 4.5 nevű fejlesztői környezet segítségével lehetséges. A VisualDSP++ 4.5 program elindítása előtt csatlakoztassuk a PC egyik USB portjához a DSP kártyát. A kapcsolat létrejöttét az USB kábel és a tápegység csatlakoztatása után az USB MONITOR LED jelzi. Ennek felgyulladása után indítható a Visual DSP++ 4.5 fejlesztői környezet.

A VisualDSP-hez tartozó rövid leírás megtalálható külön dokumentumként a Visual_DSP.pdf nevű file-ban.

A mérés során egy már kész programot használunk, mely alkalmas a másodlagos út átviteli függvényének mérésére és a zajcsökkentési algoritmus végrehajtására. A DSP-hez tartozó fejlesztői környezetben az egyes alkalmazások projektek keretében kerülnek kifejlesztésre. A mérés során felhasznált zajcsökkentő projekt neve ANCsimpData, és megtalálható a méréshez tartozó anyagokat tartalmazó könyvtárban. A projekt megnyitása a **File→Open→Project…** menü segítségével lehetséges. Itt keressük meg a megfelelő projektet.

A projekthez rendelt file-ok a baloldali file ablakban láthatóak. Ezek közül a mérés szempontjából fontos a parameters.c file, ugyanis ennek segítségével lehetséges az identifikáció, illetve a zajcsökkentés közötti váltás megvalósítása megfelelő konstansok beállításával. Amennyiben identifikációt szeretnénk végezni, abban az esetben a parameters.h file-ban a következő sorban kell a kommentezést megszüntetni:

```
#define IDENTIFICATION_MODE
```

Ezzel egyidőben a következő sort elejére tegyünk kommentező jeleket (két darab ’/’ jel):

```
//#define ANC_MODE
```

Amennyiben a zajcsökkentési funkciót szeretnénk használni, abban az esetben az ellenkező beállításokat kell alkalmazni:

```
//#define IDENTIFICATION_MODE
#define ANC_MODE
```

A definiált konstansok segítségével a fordító meg tudja határozni, melyik funkcióhoz tartozó kód kerüljön lefordításra.

A megfelelő üzemmód kiválasztása után az F7 billentyű megnyomásával, vagy a **Project→Rebuild** menüpont segítségével le kell fordítanunk a programot. Ekkor a program betöltődik a DSP programmemóriájába. Ezután az F5 billentyű, vagy a menüsorban található  gomb megnyomásával indítható el a program. Szükség esetén Shift+F5 billentyűkombinációval, vagy  gombbal leállítható (Mindezen funkciók többféle módon elérhetőek. Bővebb információk megtalálhatóak a Visual_DSP.pdf leírásban). A program leállítása után, azt újra elindítva az F5 billentyűvel, a program végrehajtása a leállítási ponttól folytatódik.

Mind az identifikációs, mind a zajcsökkentő program esetén a mót és a DSP között a megfelelő a kapcsolatot a LED7 és LED8 jelzésű LED-ek villogása jelzi. Ezt érdemes ellenőrizni első futtatáskor.

A következőkben az identifikáció és zajcsökkentés végrehajtásához szükséges lépések kerülnek bemutatásra.

Identifikáció

Feltételezzük, hogy a ANCsimpData projekt meg van nyitva.
A parameters.h file-ban a következő beállításoknak kell lennie:

```
#define IDENTIFICATION_MODE
//#define ANC_MODE
```

A következő részben NAGYBETŰVEL szedett értékek a parameters.h file-ban találhatóak.
A megfelelő beállítások végrehajtása után F7-tel lefordítva a projektet, a program betöltődik a DSP programmemóriájába. A betöltést a „Load complete.” felirat jelzi. Amint betöltődött a program, az F5 gombbal indítható az identifikáció. Az identifikáció során folyamatosan növekvő frekvenciájú szinuszos hangot kell hallanunk. A DSP adott *df* felbontással megméri az átviteli függvényt a működési tartományban.
A *df* felbontás számítható a következő módon: *df* = 1000/ID_LENGTH;
Az identifikáció IDENT_FROM_FREQ Hz-től f_{max} = ID_TOP_VAL**df* Hz-ig terjed. Minden frekvencián a parameters.h file-ban megadott IDENT_DUR konstans által meghatározott ideig történik a mérés. Amennyiben a későbbiekben kiderül, hogy sikertelen az identifikáció, érdemes ezt az időt megnövelni.

Az identifikációhoz megfelelő hangerő beállítása is szükséges. Kis hangerő esetén a zaj miatt pontatlan a mérés, nagy hangerő esetén a torzítás okoz hibát.

Az identifikáció befejezésekor a DSP által kiadott gerjesztőjel megszűnik, tehát elhallgat a szinuszos sípoló hang. Ekkor leállítható az identifikációs program (Shift+F5). Az identifikáció eredményét ekkor el kell mentenünk a PC-re, ugyanis a zajcsökkentésben az átviteli függvény inverzét használjuk fel. Az inverz számítása a PC-n MATLAB segítségével

történik. Az adatok elmentése a VisualDSP **Memory→Dump** menüpontjának kiválasztásával lehetséges. Az adatok elmentésekor beállítandó paramétereket a projektet tartalmazó könyvtárban található calcDumpData.m MATLAB file futtatásával kaphatjuk meg. A Dump menüpont egy adott memóriarész vagy változó file-ba mentésére használható. Az identifikáció során az átviteli függvény értékei az atvitel_re és atvitel_im változókba kerülnek. Ezen változók nevét kell megadni címként egymás után. Tehát a dump műveletet mindkét változóra külön-külön el kell végezni. A file-ok nevét a MATLAB program szintén megadja, azt csak be kell írni a megfelelő helyre: az átviteli függvényt tartalmazó file-okat a projektet tartalmazó könyvtárban található atvitel nevű könyvtárba kell elhelyezni atvitel_re.dat és atvitel_im.dat néven.

Az átviteli függvény inverzének számítása a projekt könyvtárában található atvKarInv.m MATLAB file lefuttatásával lehetséges. A file egyben meg is jeleníti az átviteli függvényt, és az i_atvitel_re.dat és i_atvitel_im.dat file-okba menti az inverz átviteli függvényt. Ezt a két file-t a zajcsökkentő program használja.

Zajcsökkentés

A zajcsökkentő program futtatásához a parameters.h file-ban a következő konstansdefiníciónak kell szerepelniük:

```
//#define IDENTIFICATION_MODE
#define ANC_MODE
```

A program az F7 billentyű megnyomásával fordítható le, és az F5 billentyű segítségével indítható el. Mielőtt elindítjuk a programot, győződjünk meg róla, hogy csatlakoztatva van-e a referenciajel a DSP kártya bementére és a zajelnyomó hangszóró a kártya kimentére.

A zajelnyomó program esetén a DSP kártyán található nyomógombok a következő funkciókat nyújtják:

FLG1: kimenet letiltása: a gomb nyomva tartása alatt nem adja ki a beavatkozójelet a DSP, és a szabályzó működése leáll. Ez a funkció alkalmas a ki- és bekapcsolt zajcsökkentés közti különbség vizsgálatára.

FLG2: kimenet letiltása és a szabályzó resetelése: a gomb nyomva tartása alatt nem adja ki a beavatkozójelet a DSP, és a szabályzóban található Y_n^i állapotváltozók (ld. (20)-as képlet) értéke nullázódik. Ezen funkció alkalmas a rendszer beállási tulajdonságainak vizsgálatára.

A zajcsökkentő rendszerben beállítható, hogy mekkora legyen az üzemi frekvenciatartomány. Ezt a parameters.h file-ban található `REZ_KILEP_FREKI` konstans határozza meg. Ajánlott értéke 600 Hz.

A mérés során vizsgálni fogjuk, hogy hogyan befolyásolja a (20) által meghatározott szabályzási algoritmus tulajdonságait a benne szereplő μ paraméter értéke, mely gyakorlatilag a szabályozási kör hurokerősítését adja meg. Ennek tesztelése érdekében különböző μ értékek mellett mérünk tranziens viselkedést. A paraméter változtatásához a következő lépéseket kell végrehajtanunk:

- Állítsuk le a futó programot a Shift+F5 gombokkal. A VisualDSP lehetőséget kínál a processzor memóriatartalmának manipulására (írás/olvasás), amennyiben a program futtatását megállítottuk. Ehhez meg kell nyitni egy memóriatartalmat megjelenítő ablakot.
- Amennyiben még nincs megnyitva az ablak, úgy a **Memory→Two Column** menüpont segítségével nyithatunk egy új ablakot.

- A megjelenő kis ablakhoz tartozó felső szövegmezőbe írjuk be a megjeleníteni kívánt változó nevét. Esetünkben a `nuO` nevű változó tartalmazza a μ paraméter értékét, tehát `nuO` nevet kell a mezőbe írni.
- Amennyiben még nem tettük meg, állítsuk be a megjelenített mező formátumát jobb gombbal az értékre kattintva. A formátum legyen 32 bites float szám.
- Bal egérgombbal kétszer a változóra kattintva a változó értéke átírható. Az ENTER billentyű megnyomásával érvényesíthetjük a beírt értéket.
- Az F5 gombot megnyomva a program újra elindul az új μ értékkel.

Mérési feladatok

1. Lokalizáció

1.1. Helyezzük üzembe a mérőrendszert: kapcsoljuk be a mótokat (szenzorokat és a bázisállomást is), csatlakoztassuk a PC-hez a bázisállomást. A PC-s monitorozó program elindításával ellenőrizzük, hogy működik-e a rendszer (pl. a mikrofonok megfújásakor nő-e a jelszint a kijelzőn).

A mérés során elvégzett feladatokat érdemes minél inkább automatizáltra elkészíteni: ne kelljen például minden mót esetén elvégezni ugyanazokat a műveleteket. Emiatt célszerű az egyes mótokhoz tartozó eredményeket úgy tárolni, hogy akár számmal is hivatkozhassunk rá, például mátrixban, vagy ún. cell típusú tömbökben. A végrehajtott műveleteket is érdemes mindenképpen egy script file-ba írni, hogy ne kelljen minden kísérlet alkalmával újra végrehajtani egyesével a parancsokat.

1.2. Generáljuk le a megfelelő chirp jelet. Adjuk ki a hangszóróra. Ügyeljünk a megfelelő mintavételi frekvencia megválasztására. A kiadott jelről a mótok segítségével készítsünk felvételt.

1.3. Végezzük el MATLAB-ban egy mérési eredményt tartalmazó file beolvasását. Jelenítsük meg a mérési eredményeket. Távolítsuk el a DC komponenst (vonjuk ki a jel átlagát), mert a DC jel csak az unipoláris AD-átalakítás miatt keletkezik. Amennyiben túl hosszú a regisztrátum, távolítsuk el a szükségtelen részeket. Állítsunk be a hangszórón megfelelő hangerőt, hogy megfelelően detektálható legyen a jel.

1.4. Végezzük el a hangminta interpolálását úgy, hogy mintavételi frekvenciája megegyezzen a kiadott jel mintavételi frekvenciájával. Vizsgáljuk meg az interpolált függvényt.

1.5. Számítsuk ki a kiadott chirp jel és az érzékelt jel keresztkorrelációs függvényét minden mótra (használjuk az `xcorr` függvényt). Jelenítsük meg a korrelációs függvényeket. Vizsgáljuk meg a csúcsok alakját. Milyen széles egy maximumhoz tartozó hullám? Hasonlítsuk össze a várt szélességgel (lásd: gerjesztőjel megválasztására vonatkozó fejezet). Végezzünk több mérést is különböző pozíciókban, és vizsgáljuk meg a jellemző korrelációs függvényeket.

1.6. MATLAB-ban keressük meg a korrelációs függvényben a minta érkezési idejét jelző csúcsot. Amennyiben az egyszerű maximumkeresés nem jár eredménnyel, valósítsuk meg azt a csúcskereső algoritmust, mely a korrelációs függvényben megkeresi az első csúcsot (6. ábra). Végezzük el a számítást az összes szenzorra. Ellenőrizzük az algoritmust több mérés esetén, és szemrevételezéssel vizsgáljuk meg, valóban megtalálta-e a megfelelő csúcsot.

1.7. Számítsuk ki a megtalált csúcsok közötti távolságokat minden mótpárra. Az időkülönbségekből számítsuk ki a távolságkülönbségeket.

1.8. Végezzünk egyszerű mérést az eredmények ellenőrzésére. Tegyünk a hangszóróval egyvonalba két mótot, és ellenőrizzük, hogy működik-e a távolságmérés! Mik lehetnek a hiba okai (pl. teljesül-e a síklokalizációs feltételezés). Helyezzük fokozatosan közelebb/távolabb a mótokat (pl. 5 cm-enként) és mérjük meg a távolságokat. Adott pozícióban végezzünk több mérést is. Mennyire ingadoznak az egy elrendezéshez tartozó mérési eredmények?

1.9. Egy, a mérésvezető által megadott egyszerű elrendezés esetén mérjük meg a beesési szöget két mót esetén.

1.10. Végezzük el a mérést három mót esetén, és a rendelkezésre álló MATLAB függvények segítségével határozzuk meg a forrás pozícióját.

1.11. *Kiegészítő feladat: végezzük el a méréseket interpoláció nélkül. Mekkora az interpoláció nélkül végzett mérés felbontása?*

1.12. *Kiegészítő feladat: végezzünk mérést a szinkronizáció hatásának vizsgálatára. Ehhez használjunk két mótot, és kapcsoljuk ki a szinkronizációt. Adott mérési elrendezés esetén ismételjük meg többször a mérést, és vizsgáljuk, hogy hogyan változik a mért távolság.*

2. Aktív zajcsökkentő rendszer vizsgálata

2.1. Állítsuk össze a zajcsökkentő rendszert. Gondoskodjunk a DSP kártya tápellátásáról és csatlakoztassuk a PC-hez az USB-s JTAG-et. Indítsuk el a VisualDSP fejlesztői környezetet. Reseteljük a mótot és kössük össze a mótot a PC-t és a DSP-t az erre szolgáló csatlakozók segítségével: 17. ábra. Az 1-es számú szenzor mótot kapcsoljuk be, a többit kapcsoljuk ki. Csatlakoztassuk a jelgenerátorból jövő jelet a DSP bemenetéhez, de egyelőre a zajforrásként szolgáló hangszóróhoz még ne. Kössük a zajelnyomó hangszórót a DSP kimenetére. Kössük az oszcilloszkóp két csatornájára a DSP kártya jack csatlakozóval ellátott kimenetét. Kapcsoljuk az oszcilloszkópot XY módba. Nyissuk meg az ANCsimpData nevű projektet.

2.2. A parameters.h file-ban állítsuk be az identifikációs módot. Fordítsuk le a projektet és indítsuk el az identifikációt. Gondoskodjunk megfelelő hangerőről. Az identifikáció végeztével mentsük le az eredményeket és végezzük el az átviteli függvény invertálását. Az identifikáció közben lehetőleg ne csapjunk zajt, az zavarja a mérést. Az identifikáció megkezdése után ne változtassuk meg sem a hangszóró, sem a mót pozícióját, mert az befolyásolja az átviteli függvényt.

2.3. A parameters.h file-ban állítsuk be a zajcsökkentő módot. A beavatkozó hangszórón állítsunk be maximális erősítést. Indítsuk el a zajcsökkentést. Próbáljuk ki, hogy stabil-e a rendszer. Amennyiben nem stabil a rendszer, próbáljuk meg más frekvencián használni, illetve a μ hurokerősítést csökkenteni. Előfordulhat, hogy nem volt elég alapos az identifikáció. Több sikertelen próbálkozás esetén végezzünk új identifikációt. Instabil rendszer esetén ne hagyjuk, hogy a beavatkozójel túlságosan megnőjön, az FLG2 gombbal mihamarabb reseteljük a rendszert.

2.4. Stabil rendszer esetén csatlakoztassuk a jelgenerátort a zajforrásként szolgáló hangszóróhoz. A mótokra felszerelhető egy külső zajmérő mikrofon, mellyel megfigyelhető a zajcsökkentés mértéke. A mikrofont a PC-k mikrofon bemenetéhez csatlakoztatva megfigyelhető a jel. A külső zajmérő mikrofon segítségével mérjünk le egy beállási tranzienst. Határozzuk meg a beállás idejét.

2.5. A már ismertetett módszer segítségével változtassuk a μ hurokerősítést. Figyeljük meg, hogyan változik a tranziens viselkedés (FLG2-vel resetelhetjük az algoritmust, és újabb beállásokat vizsgálhatunk). Milyen értéknél válik instabillá a rendszer? Hol a leggyorsabb a beállás?

2.6. Kapcsoljuk ki a zajt, és mozdítsuk el a szenzort az eredeti pozíciójából. Mi történik? Miért?

2.7. Kapcsoljuk be a zajforrást, és FLG2 folyamatos megnyomásával kapcsoljuk ki a zajcsökkentést. A PC-s program segítségével figyeljük meg, hogy különböző frekvenciájú zaj esetén a szenzort kb. 20 cm-rel elmozdítva mennyit változik a mért jel alapharmonikusának fázisa. Milyen következtetést lehet levonni az átviteli függvény megváltozására vonatkozóan?

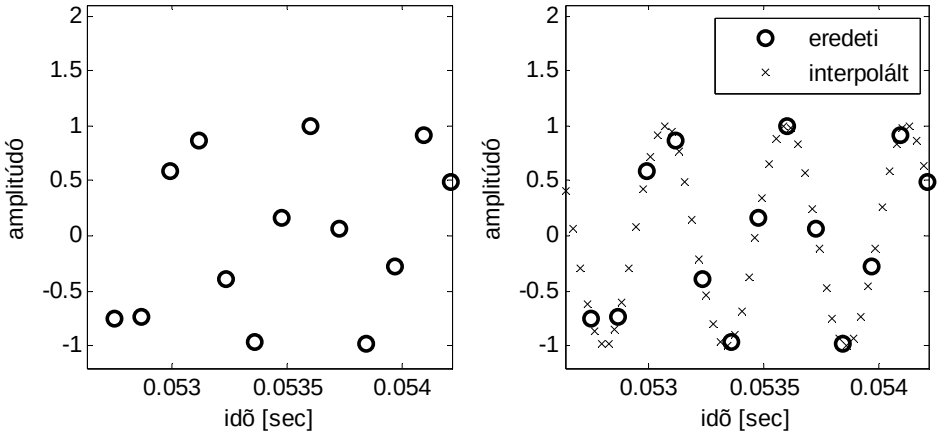
2.8. A móthoz csatlakoztatott mikrofon segítségével végzett mérés eredményét dolgozzunk fel MATLAB-ban. Végezzünk spektrumanalízist, mekkora a zajelnyomás mértéke?

Függelék

Interpoláció bemutatása

A gyakorlatban több olyan probléma is felmerül, amikor egy adott mintavételi frekvenciával mért regisztrátumot más mintavételi frekvenciával szeretnénk feldolgozni, más mintavételi frekvenciának megfelelően szeretnénk megjeleníteni. Ez történhet például különböző sebességgel működő hangrögzítő és -lejátszó eszközöknél, szükség lehet a mintavételi frekvencia növelésére az időtartománybeli jel pontosabb vizsgálatához is. Ez utóbbi fontos lehet, ha vizuálisan szeretnénk megvizsgálni egy jelet, ekkor ugyanis a mintavételi frekvenciához közel eső jelek már nehezen vizsgálhatóak.

Tekintsük például a 19. ábrát. Látható hogy a 19.a. ábrán igen nehezen határozható meg, hogy egy szinuszos jelről van szó annak ellenére, hogy a mintavételi tételt betartottuk, mivel egy periódusból több, mint két mintát vettünk. Ezzel szemben a 19.b. ábrán négyszeres interpoláció után a jel alakja már jobban kivehető, nagyobb arányú interpolációnál természetesen még nagyobb javulást várhatunk. Ez hasznos lehet akkor is, ha a jel valamilyen időtartománybeli paraméterét szeretnénk minél pontosabban meghatározni. Példa lehet erre egy egyszerű nullátmenet vagy maximumhely vizsgálata, ahol a megnövelt mintavételi frekvenciának köszönhetően pontosabban be lehet határolni a nullátmenet vagy a maximum helyét; erre az ábra szintén jó példa lehet.



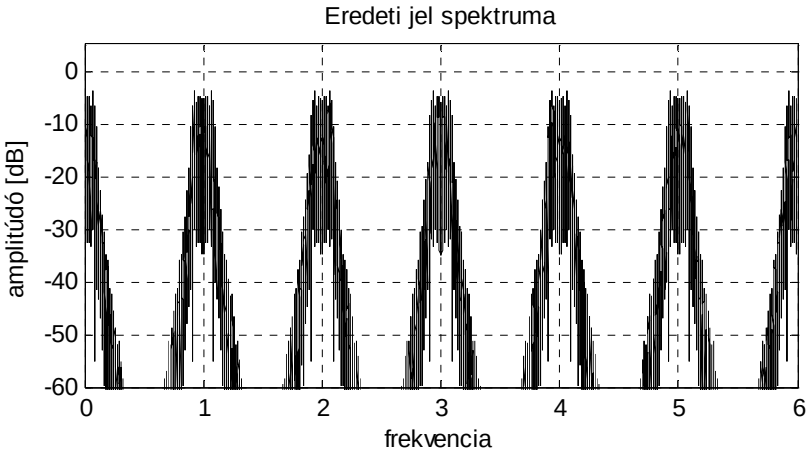
19.a. ábra *19.b. ábra*

19. ábra. Eredeti mintavételi frekvencián és a mintavételi frekvencia négyszeresére növelésével (interpolálásával) megjelenített jel

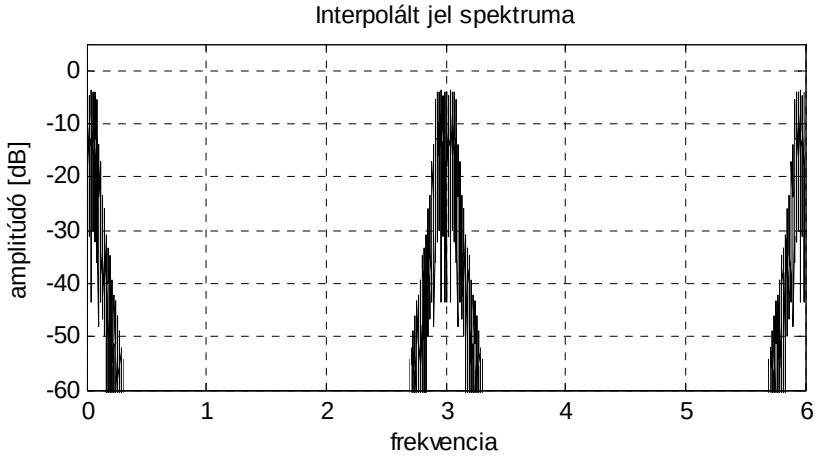
A következőkben arra az esetre térünk ki, amikor az eredeti mintavételi frekvenciának egész számú többszörösére kell növelni a mintavételi frekvenciát. Amennyiben betartottuk a mintavételi tételt elméletileg ennek nincsen akadálya, hiszen ekkor a mintavételezett jelből akár a folytonos jel is előállítható.

Az algoritmus megismerése előtt tekintsük a 20. és 21. ábrát, melyeken egy egységnyi mintavételi frekvenciával mintavételezett jel és ugyanazon, de háromszoros mintavételi frekvenciával mintavételezett jel spektruma látható. Mivel a mintavételi frekvencia rendre

$f_s = 1$ és $f'_s = 3$, így az alapsávi spektrum is ennek megfelelően ismétlődik a mintavételezés miatt (természetesen az ismétlődés (-∞,∞) tartományban fennáll, az ábrákon csupán egy szeletet ragadtunk ki). A háromszoros mintavételi frekvenciával mintavételezett jel úgy is felfogható, mintha az egyszeres mintavételi frekvenciával mintavételezett jel mintavételi frekvenciáját háromszorosára interpoláltuk volna. Az interpoláció célja tehát látható: valahogyan el kell távolítani a spektrumból azokat a részeket, melyek nem szükségesek a magasabb mintavételi frekvencia esetén. Példánkban (20. és 21. ábra) ez az 1, 2, 4, 5… frekvencia körüli megismételt alapsávi spektrum eltávolítását jelenti.



20. ábra. Eredeti (interpolálandó) jel spektruma. Mintavételi frekvencia=1, tehát f_s=1



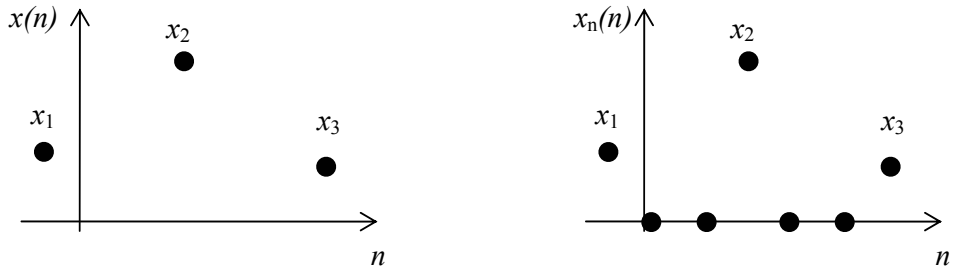
21. ábra. Interpolált jel spektruma. Új mintavételi frekvencia=3, tehát f'_s=3

A fent vázolt feladat megoldására például a következő, két lépésből álló algoritmus használható, mely viszonylag egyszerű eszközökkel megvalósítható. (természetesen többféle módszer is létezik)

1. lépés

Tekintsünk egy N_i-szeres interpolációt, tehát legyen az új mintavételi frekvencia N_i-szerese az eredetinek: $f'_s = N_i \cdot f_s$. Állítsunk elő egy olyan jelet, melynek mintavételi frekvenciája legyen f'_s és úgy kapjuk meg, hogy az eredeti f_s mintavételi frekvenciával mintavételezett jel mintái közzé $(N_i - 1)$ darab nullát szúrunk be. Ezt szemlélteti a 22. ábra. Ekkor maga a jel

időfüggvénye megegyezik az eredeti f_s mintavételi frekvenciával mintavételezett jel időfüggvényével, hiszen a mintavételi pontok között a függvény egyébként is nulla volt. Mindez azért van, mert a mintavételezés megfelel Dirac-delta sorozattal való szorzásnak (modulációnak), és a Dirac-delta függvény csupán egy pontban, a mintavételi pontokban nem nulla. A mintavételi pontok között tehát nullákat kapunk. Mindez azt jelenti, hogy a nullákkal kiegészített jel spektruma is megegyezik az eredeti jel spektrumával, mely a 20. ábrán látható.



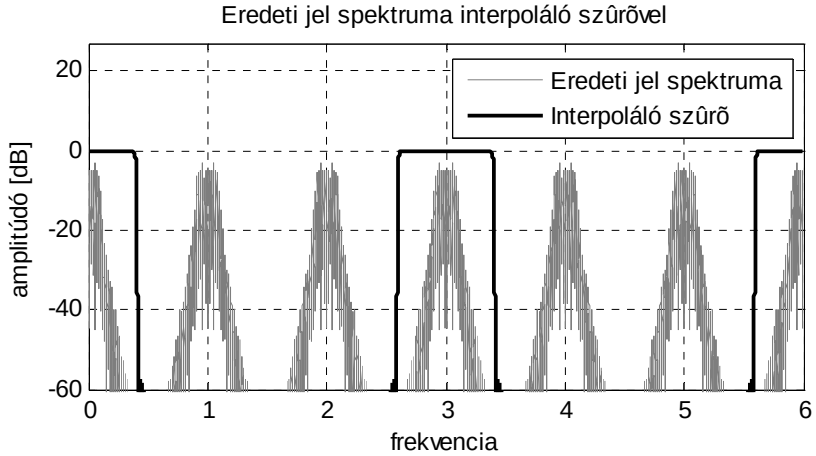
eredeti mintasorozat: [x_1 x_2 x_3

nullákkal kiegészített mintasorozat: [x_1 0 0 x_2 0 0 x_3 0 0…]

22. ábra. Nullákkal kiegészített jel

2. lépés

Második lépésként eltávolítjuk a nem szükséges részeket a spektrumból. Ennek szemléltetése látható a 23. ábrán.



23. ábra. Interpoláló szűrő karakterisztikája az interpolálandó jellel együtt f_s=1, f'_s=3

Szürkével láthatjuk a nullákkal kiegészített jel spktrumát, mely, mint már beláttuk, megegyezik az eredeti mintavételi frekvenciájú jel 20. ábrán látható spektrumképével, viszont a mintavételi frekvencia immár f'_s, mely a példában $f'_s = 3$. A nemkívánatos komponensek eltávolítása a spektrumból egy aluláteresztő szűrő segítségével lehetséges. A szűrőt úgy kell megtervezni, hogy az eredeti f_s mintavételi frekvencia fele alatti komponenseket meghagyja, a magasabb frekvencián található komponenseket kiszűrje. Ez a szűrő a 23. ábrán vastag fekete vonallal látható. Megállapíthatjuk, hogy a szűrő eltávolítja az 1, 2, 4, 5… frekvenciák körül ismétlődő alapsávi spektrumokat, de a 0, 3, 6… frekvenciák körül ismétlődő alapsávi

spektrumokat megtartja. Ezzel már el is jutottunk a megkívánt spektrumképhez, mely a 21. ábrán látható.

Felmerülhet a kérdés, hogy miért ismétlődik az interpoláló szűrő karakterisztikája f_s's frekvenciánként. Ennek oka, hogy a mintavételezett rendszerekben a mintavételezett jel spektruma mintavételi frekvenciánként ismétlődik. A digitális szűrő pedig valójában csupán egy mintasorozat, mellyel konvolváljuk a szűrendő jelet. A mintasorozat pedig a szűrő impulzusválasza. Kissé más megközelítésből vizsgálva a problémát: egy f_s mintavételi frekvenciával mintavételezett jelet szűrve, a szűrt jel szintén f_s mintavételi frekvenciával áll rendelkezésre, tehát ennek spektruma is f_s-enként kell, hogy ismétlődjön. Mivel a szűrt jel az eredeti jel spektrumának és a szűrő átviteli karakterisztikájának szorzata, így a szűrt jel spektruma csak akkor ismétlődhet f_s-enként, ha a szűrő átviteli karakterisztikája is f_s-enként ismétlődik.

A szűrés megvalósításával kapcsolatban érdemes megemlíteni, hogy ideális esetben az f_s alatti részen az interpoláló szűrő átvitele egységnyi lenne, a felette lévő részben pedig nulla. A valóságban természetesen ilyen szűrő nem létezik, figyelembe kell venni, hogy a szűrő meredeksége nem lehet végtelen. Emiatt a szűrő törésponti frekvenciáját (ahol az átviteli karakterisztika elkezd csökkenni) kissé f_s alá kell tervezni, tehát a szűrő befolyásolja az alapsávi eredeti spektrumot. Másrészt el kell fogadni, hogy a szűrő elnyomása f_s felett sem végtelen, tehát f_s felett is tartalmazni fog az interpolált jel spektruma az interpolálandó jelből származó komponenseket. Ez mind csökkenti az interpoláció minőségét, de megfelelő szűrő használatával ezen hibák elhanyagolhatóvá tehetőek.

A MATLAB-ban a szűrők tervezésére többféle módszer áll rendelkezésre. IIR szűrők tervezéséhez használható például a `butter, cheby1, cheby2, ellip...` függvény, FIR szűrők tervezéséhez használhatjuk a `firpm` (7.0-nál korábbi verziókban `remez`), `fir1, fir2, firls...` függvények. A használatukkal kapcsolatos ismeretek megtekinthetők a MATLAB help segítségével. A megtervezett szűrővel való szűrés egyszerűen elvégezhető a MATLAB `filter` parancsa segítségével. Itt fontos lehet, hogy FIR szűrők esetén a szűrő átviteli függvényének nevezője konstans egy, a `filter` parancs számára szükséges átviteli függvény nevező helyére 1-es írandó. A megtervezett szűrő átviteli karakterisztikájának ellenőrzésére használhatjuk a `freqz` parancsot.

A mérés során rendelkezésre áll egy kész függvény, mely segít az interpoláció végrehajtásában:

`interpolate` függvény használata:

Tekintsük a következő példát: egy `x` vektorban található adatok interpolációját szeretnénk elvégezni, a kívánt új mintavételi frekvencia az eredeti mintavételi frekvencia 20-szorosa. Ekkor a következő függvényhívással végezhető el az interpoláció:

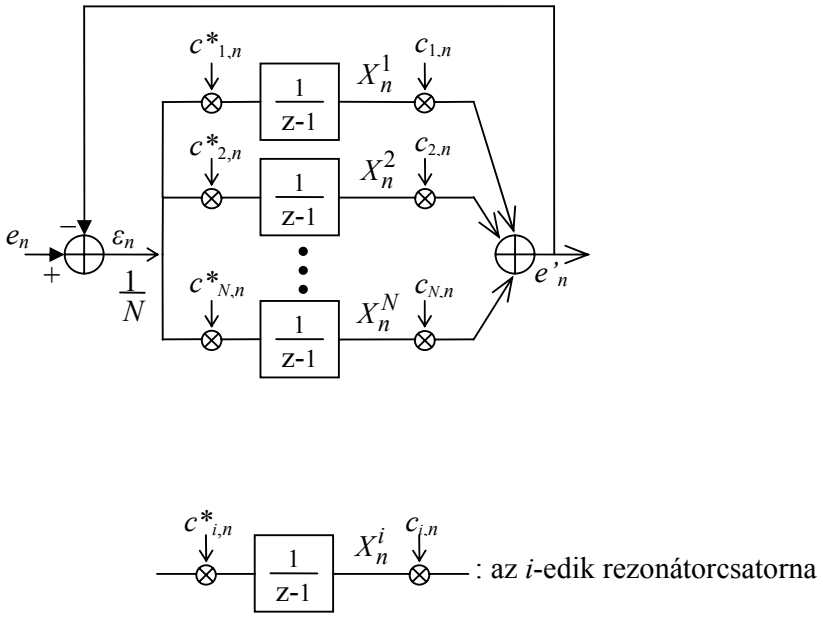
```
y = interpolate(x,20); %y az interpolált értékeket tartalmazza.
```

Az interpolálást végző függvény kódja:

```
function interpDat = interpolate(datToInterp,intRat)
%interpDat = interpolate(datToInterp,intRat)
%A fügvény a datToIntep paraméterként megadott jel interpolációját végzi el.
%Az interpoláció aránya az intRat paraméterrel adható meg.
  datToInterp=datToInterp(:)';  %sorvektorrá alakítás
  interpDat = [datToInterp;zeros(intRat-1,length(datToInterp))];
  interpDat = interpDat(:)'; %kiegészítés nullákkal (első sorral együtt)
  [intfb,intfa] = butter(6,0.8/intRat);  %interpoláló szűrő tervezése
  interpDat = filter(intfb,intfa,interpDat)*intRat; %szűrés
```

A Fourier-analizátor struktúra működése

A Fourier-együtthatók előállítása a *24*. ábrán látható rezonátor alapú Fourier-analizátor segítségével történik.



24. ábra. Fourier-analizátor (FA)

Az ábrán X_n^i jelöli az e_n jel komplex Fourier-együtthatóit, valamint $c_{i,n} = \exp(j \cdot 2 \cdot \pi \cdot i \cdot f_1 \cdot n)$, amelyek a felbontás alapjául szolgáló bázisfüggvények. f_1 az alapharmonikus mintavételi frekvenciához képest vett relatív frekvenciátját, i a harmonikus számát, n az időt, * pedig a komplex konjugáltat jelöli. Látható, hogy az alapharmonikus frekvenciájának ismerete szükséges a felbontás elvégzéséhez, az határozza meg a bázisfüggvények frekvenciáját.

Az analizátor N db úgynevezett rezonátor csatornát tartalmaz. Egy csatorna működése a következő: az i. rezonátor a bemenetén lévő jelben található i. harmonikust a $c^*_{i,n}$ forgó egységvektor segítségével zérus frekvenciára keveri. A DC jelre az $\frac{1}{z-1}$ átviteli függvénnyel jellemezhető integrátor átvitele végtelen. Az integrátor kimenetén egy X_n^i komplex szám jelenik meg, melyet a $c_{i,n}$ forgó vektor visszakever az eredeti i. harmonikus frekvenciájára. Mindez úgy is értelmezhető, hogy az integrátor végtelen erősítését eltoltuk az i. harmonikus frekvenciájára, tehát az i. rezonátor csatorna i. harmonikusra vett átvitele végtelen. Mivel a rezonátorok egy visszacsatolt szabályozási körben találhatóak, így a felbontandó e_n jelben található i. harmonikust hiba nélkül képes előállítani. A 24. ábra alapján a Fourier-analizátor kimenetét (21) egyenlet írja le:

$$e'_n = \sum_{i=-N}^{N} X_n^i c_{i,n}, \quad \text{ahol} \quad c_{i,n} = \exp(j \cdot 2\pi \cdot i f_1 n). \tag{21}$$

Az egyenlet a jelek Fourier-felbontásának alakját adja, melyben X_n^i az i. harmonikus Fourier-együtthatóját jelöli.

Mint láttuk, a felbontás során fontos, hogy a $c_{i,n}$ függvényekben f_1 megegyezzen a zaj valódi alapharmonikus frekvenciájával (f_{1_zaj}). Erről egy adaptív Fourier-analizátoros struktúra (AFA) gondoskodik. Az AFA egy PLL funkcionalitású struktúra, mellyel frekvenciamérés valósítható meg. Működésében kihasználjuk, hogy amennyiben a 24. ábrán látható FA-ban nem teljesül $f_{1\_zaj} = f_1$, akkor az X_n^1 együttható a frekvenciaeltéréssel arányos sebességgel forog. Ennek oka, hogy az f_1 frekvenciától eltérő f_{1_zaj} frekvenciájú jel felfogható úgy is, mint egy f_1 frekvenciájú, de folytonosan változó fázisú jel. Ez a folytonos fázisváltozás jelenti a Fourier-együttható vektorának forgását. A forgás iránya függ az eltérés irányától (kisebb/nagyobb frekvencia), a forgás sebessége pedig függ az eltérés nagyságától. Ezek alapján meghatározható f_{1_zaj} frekvencia, és f_1 folyamatos hangolásával $f_{1\_zaj} = f_1$ teljesíthető.