Akustisk lokalisjonssystem m/instrumenteringssystem

RAPPORT

Jakob Vahlin & Andreas Horpedal Bugge

Fakultet for informasjonsteknologi og elektroteknikk
Norges teknisk-naturvitenskapelige universitet

Sammendrag

Vinkelen til et innkommende lydsignal er blitt beregnet med et akustisk lokalisjonssystem som sammenlikner lydsignalene fra tre mikrofoner plassert i en likesidet trekant. Et instrumenteringssytem som sampler lydsignalene fra mikrofonene er laget. De samplede dataene lagret på en Raspberry Pi hvor vinkelen er blitt beregnet digitalt med et Python-script.

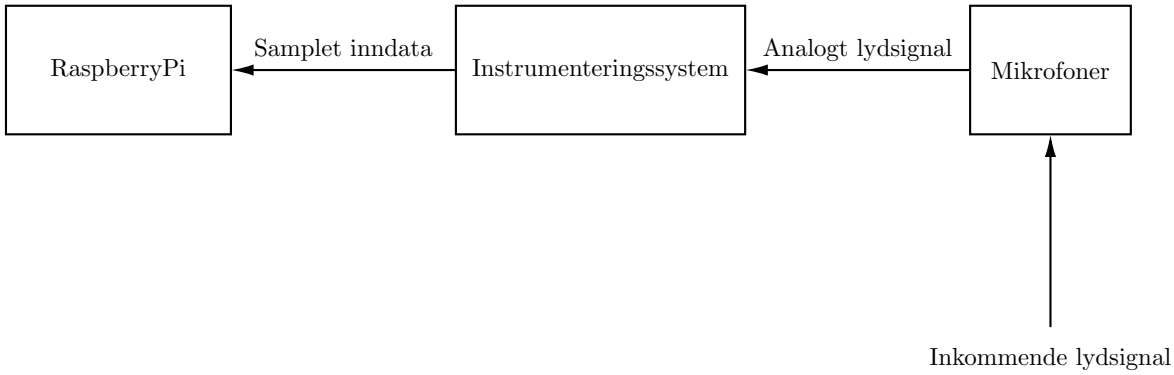
Det har blitt gjort målinger av 10 forskjellige vinkler, jevnt fordelt på en 360° gradskive. Vinkelestimatet for en spesfikk vinkel er basert på 5 målinger av samme vinkel. Det gjennomsnittlige avviket fra referansevinkelen for alle målinger ble 1.87° med et gjennomsnittlig standardavvik på 4.29 °. Dette betraktes som et svært godt resultat når usikkerheten i referansevinkelen som forsøkes målt er 6°. Instrumenteringssystemet er i stand til å sample og lagre informasjonen fra lydsignalet i stor nok grad til at feilkildene hovedsaklig kommer fra usikkerhet i målemetoden og liten avstand mellom mikrofonene.

Innhold

1 Innledning

En mikrofon er en type sensor som omformer lyd i form av trykkvariasjoner i luften til et elektrisk signal. Helt siden den første mikrofonen ble utviklet på 1870-tallet i England av David Edward Hughes [1], har denne typen sensorteknologi fått en rekke anvendelser. Dette kan være alt fra anvendelser de fleste er kjent med, som forsterkning eller opptak av lyd, til bruk av av ultralyd til å stille medisinske diagnoser eller lete etter olje hundrevis av meter under havoverflaten.

I denne rapporten vil et design og implementasjon av et akustisk lokalisjonssystem som beregner vinkelen på et inkommende lydsignal bli diskutert. Lokalisjonssytemet består av 3 mikrofoner som tar opp lyden til et innkommende lydssignal samt et instrumenteringssystem sampler og lagrer lydsignalene på en Raspberry Pi. På denne vil vinkelen bli beregnet digital. Figur 1 vister et blokkskjema av hele systemet.

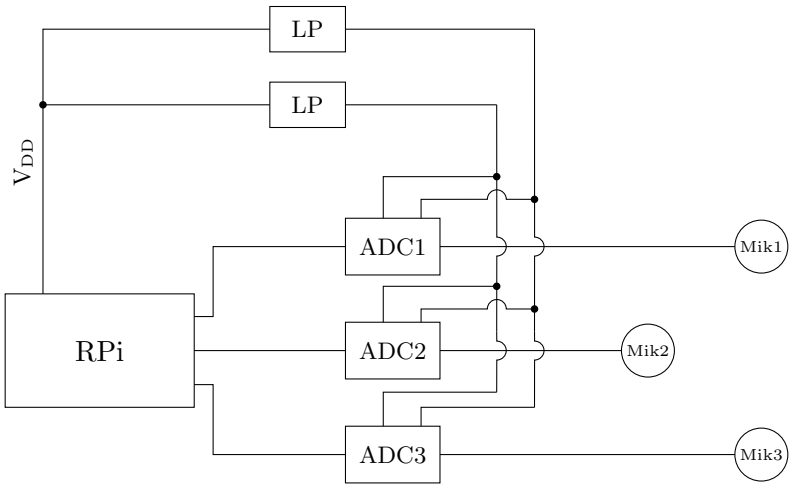


Figur 1: Blokkskjema for hele det akustiske lokalisjonssystemet.

2 Teori

2.1 Instrumenteringssystem

For å digitalt behandle lyd fra en mikrofon må lyden mikrofonen tar opp konverteres fra et analogt til et digitalt signal slik at lydsignalet kan lagres på datamaskinen som skal gjennomføre beregningene. Til dette trengs en "*Analog to Digital Converter*" (ADC) som kan konvertere et innkommende analogt signal til et digitalt signal. Ettersom lokalisjonssystemet som nevnt innledningsvis består av 3 stk mikrofoner som tar opp lyd samtidig trengs et instrumenteringssystem med 3 stk ADCer, en til hver mikrofon. En blokkskjemarepresentasjon av det akustiske lokalisjonssytemet med oppdeling av instrumenteringssystemet er vist i Figur 2.

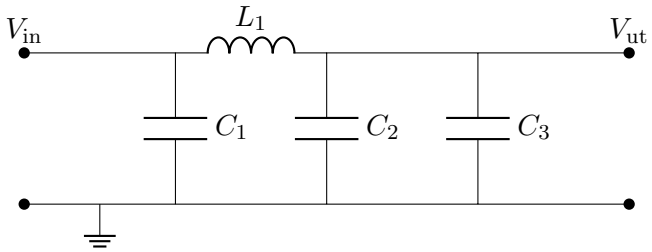


Figur 2: Blokkskjemarepresentasjon av det akustiske lokalisjonssytemet. Det plasseres et lavpassfilter mellom RPIens og ADCens referenasespenning og forsyningspenning.

Som vist i Figur 2 plasseres et lavpassfilter (indikert med LP) på forsyningsspenningen og referansespenningen til ADCen for å separere den analoge delen fra den digitale delen. Lavpassfilteret blir brukt som et dekoblingsnettverk mellom ADCene og datamaskinen som lagrer de samplede dataene. Dette blir gjort for å fjerne støy på DC signalene slik at unøyaktigheter i målingene reduseres, samt for å minimiere avviket fra det ideele teoretiske systemet. Lavpassfilteret dekoblingsnettverket består av er vist i Figur 3. RPien kobles til V_{in} og ADCen kobles til V_{ut}.

Spolen L_1 bør han en høy impedans slik at mesteparten av "støystrømmen" går gjennom C_1 eller C_2 og ikke "gjennom" filteret og dermed ikke inn i ADCen eller RPien. I tilegg danner spolen L_1 sammen med kondensatoren C_2 et lavpassfilter, som demper høyfrekvent støy. Kondensatoren C_3 bør være signifikant mindre enn C_2. En stor kondensator C_2 vil ha stor impedans ved høyre frekvens, noe som i utgangspunktet er ønskelig i et lavpassfilter. Men pga parasitt induktans har en stor kondensator også høy impedans ved høye frekvenser (pga spoleliknende egenskaper). C_2 alene vil gi god dempning for "mellomstore" frekvenser, men for veldig høye frekvenser vil dempningen være redusert. Derfor introduseres en kondensator C_3 med lav kapasitans ettersom denne vil ha lavere impedans ved veldig høyre frekvenser.

Størrelsen på kondensatoren C_1 bør også være relativt stor (samme størrelsesordning som C_2) for å stabilisere spenningen fra RPien]



Figur 3: Lavpassfilter til støyfjerning på forsyning- og referansespenningen i instrumenteringssystemet.

For at all informasjon om det analoge lydsignalet skal bli ivaretatt i det tilsvarende samplede digitale lydsignalet må samplingsfrekvensen, f_s, oppfylle Nyqvist teoremet, gitt ved

$$f_s > 2 \cdot f_{max} \tag{1}$$

hvor f_{max} er den maksimale frekvensen i båndbredden til signalet som blir samplet. Ved testing av systemet vil lydbølgene fra et håndklapp bli brukt som lydkilde. Hørbare lydbølger har en båndbredde lagt mellom 20 og 20000Hz [2]. Følgelig bør samplingsfrekvensen til ADCen være større en 40kHz for å unngå aliasing.

2.2 Beregning av retning til lydkilde

Krysskorrelasjon og oppsampling

Krysskorrelasjonen mellom to tids-diskre signaler $x(n)$ og $y(n)$ er gitt av

$$r_{xy}(l) = \sum_{n=-\infty}^{\infty} x(n+l)y(n), \text{for l} = 0, \pm 1, \pm 2, \pm 3, ... \tag{2}$$

hvor $r_{xy}(l)$ er et mål på likheten mellom signalene $x(n)$ og $y(n)$ ved en gitt lag, l [3]. Jo større verdi på $r_{xy}(l)$, desto likere er signalene $x(n)$ og $y(n)$. Krysskorrelasjoen mellom to like signaler, der det ene har en tidsforsinkelse relativt til det andre, vil altså gi en høy verdi. Basert på dette kan krysskorrelasjonen brukes til å beregne tidsdifferansen mellom to like signaler. Dette gjøres ved å bruke (2) til å beregne hvilken lag, l, som gir maksimal verdi av $|r_{xy}(l)|$. Tidsforsinkelsen mellom de to signalene er så gitt av (3) der f_s er samplingsfrekvensen [4].

$$\Delta t = \frac{l}{f_s} \tag{3}$$

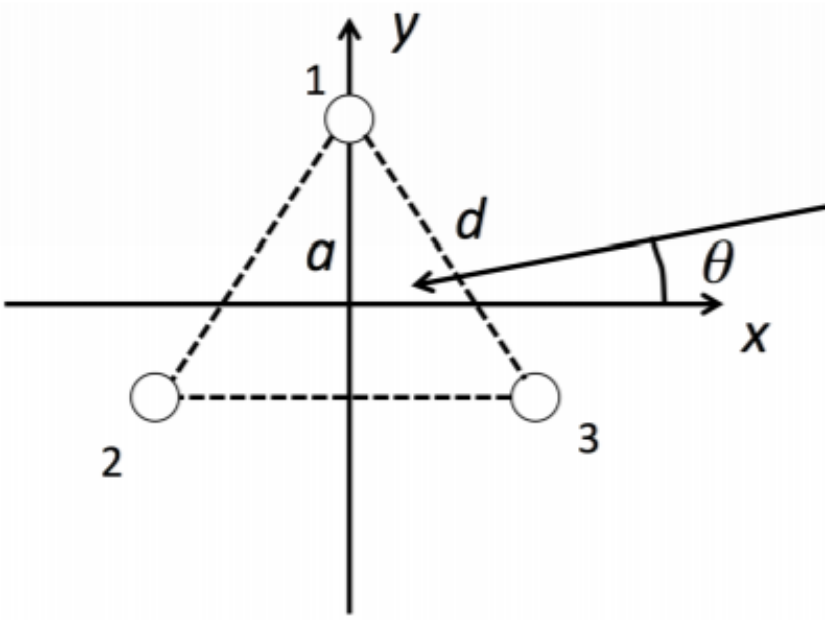
Når vi bestemmer hvor $\mid r_{xy}(l) \mid$ har sitt maksimum, kan vi i utgangspunktet ikke gjøre dette med større nøyaktighet enn hele samplingsverdier. Med andre ord så får vi en maksimal usikkerhet på ± 0.5 samplingsverdier. Et signal med f_s=4000Hz, altså 0.25ms mellom hver sample,

vil da få en maksimal usikkerhet i tidsdifferanse mellom to like signaler på 0.5·0.25ms=125μs. For signaler med svært liten tidsdifferanse kan denne usikkerheten gi store feil i videre beregninger.

En metode å redusere denne usikkerheten på, er ved å oppsample signalet. Med oppsampling menes at man approksimerer hvordan signalet hadde sett ut gitt en høyere samplingsfrekvens enn den originale [5]. Å oppsample et signal med original samplingsfrekvens f_s=4000Hz 16 ganger, gir en ny samplingsfrekvens $f_{s,new}$=64000Hz. Med det blir den maksimale usikkerheten i tidsdifferansen på 7.8μs, altså over ti ganger mindre enn originalt.

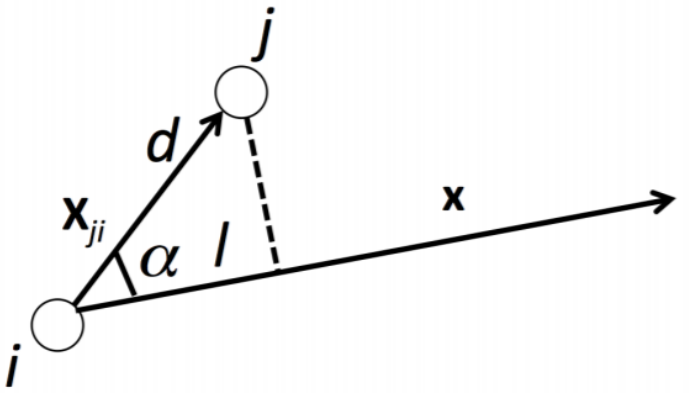
Beregning av innfallsvinkel

For å beregne innfallsvinkelen til lydsignalet er det avgjørende å organisere mikrofonene på en hensiktsmessig måte. Figur 4 viser et mulig oppsett for tre mikrofoner. Her er mikrofon 1, 2 og 3 plassert i hvert av hjørnene i en likesidet trekant med sideflater av lengde d. a er lengden fra punktet (0,0) i koordinatsystemet til mikrofon 1, mens vinkelen θ er avviket fra x-aksen og den vinkelen som ønskes beregnet.



Figur 4: Organisering av mikrofon 1,2 og 3 [4].

Ved å organisere de tre mikrofonene slik som i Figur 4 vil et innkommende lydsignal treffe hver av mikrofonene på ulike tidspunkt. Ved å bruke krysskorrelasjon beskrevet over kan vi beregne tidsdifferansen lyddsignalet treffer hver av mikrofonene med. Hvorfor dette er nyttig kan illustreres ved å se på et enkelt par med mikrofoner, mikrofon i og mikrofon j som i Figur 5. Her er vektoren $\vec{X}$ det innkommende lydsignalet med lengde l, $\vec{X_{ji}}$ er den kjente vektoren som går fra mikrofon i til mikrofon j med lengde d og α er vinkelen lydsignalet treffer mikrofon i med relativt til vektoren $\vec{X_{ji}}$.



Figur 5: To sensorer i og j [4].

For et slikt mikrofonpar som i Figur 5 kan α enkelt beregnes med (4) [4]. Her er c lydens hastighet og τ_{ji} er tidsdifferansen lydssignalet treffer mikrofon j med sammenliknet med mikrofon i beregnet ved krysskorrelasjon.

$$\cos\alpha = \frac{-c\tau_{ji}}{\mid \vec{X_{ji}} \mid} \tag{4}$$

Problemet med (4) er at det finnes to vinkler, α, som oppfyller ligningen. Følgelig trengs en tredje mikrofon for å avgjøre hvilken vinkel som er den korrekte, og en oppkobling som i Figur 4 er hensiktsmessig. Videre viser det seg at det er lettere å identifisere vektoren $\vec{X}$ istenden for vinkelen α, ettersom $\vec{X}$ vil være felles for alle tre mikrofoner mens α vil være ulik fra en mikrofon til en annen. Et utrykk for $\vec{X}$ kan bli utledet fra definisjonen av prikkproduktet mellom to vektorer.

$$\vec{X_{ji}} \cdot \vec{X} = \mid \vec{X_{ji}} \mid\mid \vec{X} \mid \cos\alpha \tag{5}$$

(4) innsatt i (5) gir

$$\vec{X_{ji}} \cdot \vec{X} = \mid \vec{X_{ji}} \mid\mid \vec{X} \mid \frac{-c\tau_{ji}}{\mid \vec{X_{ji}} \mid} \tag{6}$$

Ved å definere $\mid \vec{X} \mid$=1, kan (6) skrives om til

$$\vec{X_{ji}} \cdot \vec{X} + c\tau_{ji} = 0 \tag{7}$$

Med mikrofonene koblet opp som i Figur 4 kan tre vekotorer som går fra mikrofon til mikrofon defineres: $\vec{X_{21}}$, $\vec{X_{31}}$ og $\vec{X_{32}}$. Med de korresponderende tidsdifferansene τ_{21}, τ_{31} og τ_{32} kjent, blir (7) til et ligningssett bestående av tre ligninger med en ukjent, nemlig $\vec{X}$.

$$\vec{X_{ji}} \cdot \vec{X} + c\tau_{ji} = 0, \quad ji = 21, 31, 32 \tag{8}$$

Ved å finne løsninger for $\vec{X}$ kan vektoren dekomponeres til en x-komponent og en y-komponent. Innfallsvinkelen til lydssignalet, θ. kan beregnes fra sammenhengen mellom x,y og θ gitt ved

(9).

$$\theta = \arctan\frac{y}{x} \tag{9}$$

Ligningssettet (8) kan løses ved minste kvadraters metode. Med det introduseres et mål på feil, ϵ, som indikerer hvor godt estimatene for x og y passer de målte verdiene τ_{21}, τ_{31} og τ_{32} [4]. Følgelig er det da ønskelig å finne verdier for x og y som minimerer ϵ. Dette gir

$$\epsilon^2 = \sum_{ji}(\vec{X_{ji}} \cdot \vec{X} + c\tau_{ji})^2 = \sum_{ji}(x_{ji}x + y_{ji}y + c\tau_{ji})^2 \tag{10}$$

Ved å derivere (10) med hensyn på x og y, og så sette begge derivatene lik null, kan x-komponentene og y-komponentene som minimerer ϵ beregnes.
Å organisere mikrofonene som i Figur 4, gir at

$$\vec{X_{21}} = [-\frac{\sqrt{3}}{2}, -\frac{3}{2}]\text{a} \tag{11}$$

$$\vec{X_{31}} = [\frac{\sqrt{3}}{2}, -\frac{3}{2}]\text{a} \tag{12}$$

$$\vec{X_{32}} = [\sqrt{3}, 0]\text{a} \tag{13}$$

Fra (10) må x-komponenten og y-komponenten som minimerer ϵ være

$$x = \frac{2c}{9a^2}\sum_{ji} x_{ji}\tau_{ji}, \quad y = \frac{2c}{9a^2}\sum_{ji} y_{ji}\tau_{ji} \tag{14}$$

Ved å sette likningene (11),(12) og (13) inn i (14) og deretter (14) inn i (9), blir innfallsvinkelen til lydsignalet, θ, gitt av (15) [4].

$$\theta = \arctan(\sqrt{3} \cdot \frac{\tau_{21} + \tau_{31}}{\tau_{21} - \tau_{31} - 2\tau_{32}}) \tag{15}$$

I praksis oppereres det med samplede signaler. Av den grunn må tidsdifferansene behandles i antall sampler og ikke sekunder. Følgelig blir den tidsdiskrete versjonen av (15)

$$\theta = \arctan(\sqrt{3} \cdot \frac{n_{21} + n_{31}}{n_{21} - n_{31} - 2n_{32}}) \tag{16}$$

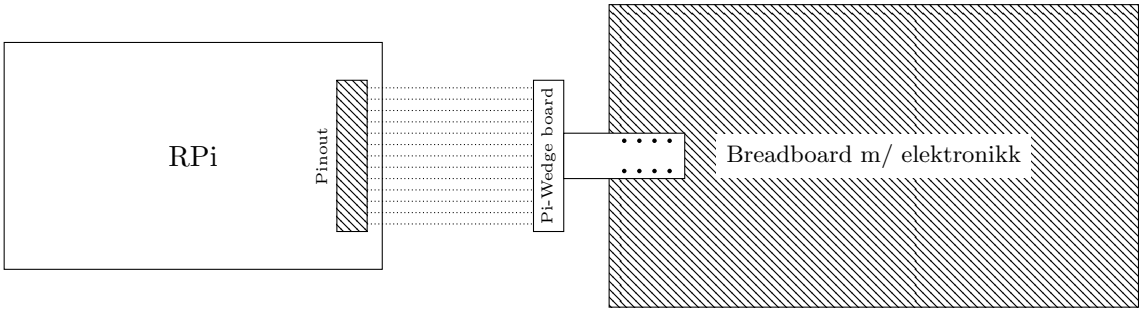
hvor n_{ji} er tidsdifferansen lyddsignalet treffer mikrofon j med sammenliknet med mikrofon i, nå gitt i antall sampler.

Noe verdt å merke seg ved ligning (16), er at en arctan funksjon kun gir vinkler i intervallet $[\frac{-\pi}{2}, \frac{+\pi}{2}]$. Med andre ord kan ikke innfallsvinkelen fra alle retninger bli funnet kun basert på (16). Ved å gå tilbake til uttrykket for x og y i (14), viser det seg at ved å legge til π i estimatet for vinkelen θ når $x < 0$, blir det mulig å beregne vinkler fra alle retninger [4].

Et siste poeng ved utregning av vinkelen θ, er at tidsdifferansen, n_{ji}, er gitt i hele samplingsverdier og kan kun ta et begrenset antall verdier. Med andre ord blir også nøyaktigheten på beregningen av vinkelen θ også begrenset. En større nøyaktighet på beregningene oppnås ved at n_{ji} kan ta et større intervall med verdier. Dette kan oppnåes ved å øke samplingsfrekvensen f_s, oppsampling eller å øke avstanden d mellom mikrofonene.

3 Realisering av systemet

Under realisering og testing av systemet benyttes 3 stk Microchip MCP3201 ADCer til å konvertere de analoge lydsignalene fra mikrofonene. Mikrofonene som blir brukt er Electret Microphone Amplifier - MAX4466. Mikrofonene, ADCene og lavpassfilterne kobles sammen på et breadboard i henhold til blokkskjemaet vist i Figur 2. En Raspberry Pi 3B blir brukt til å lagre samplet inndata samt å forsyne målesystemet med spenning. For å koble sammen Raspberry Pien til breadboardet benyttes et Pi-Wedge breakout board fra SparkFun. Dette "avbilder" pinnekonfigurasjonen til Pien fra pinnene til Pien til breadboardet. En illustrasjon av systemet med Wedge brettet er vist i Figur 6.

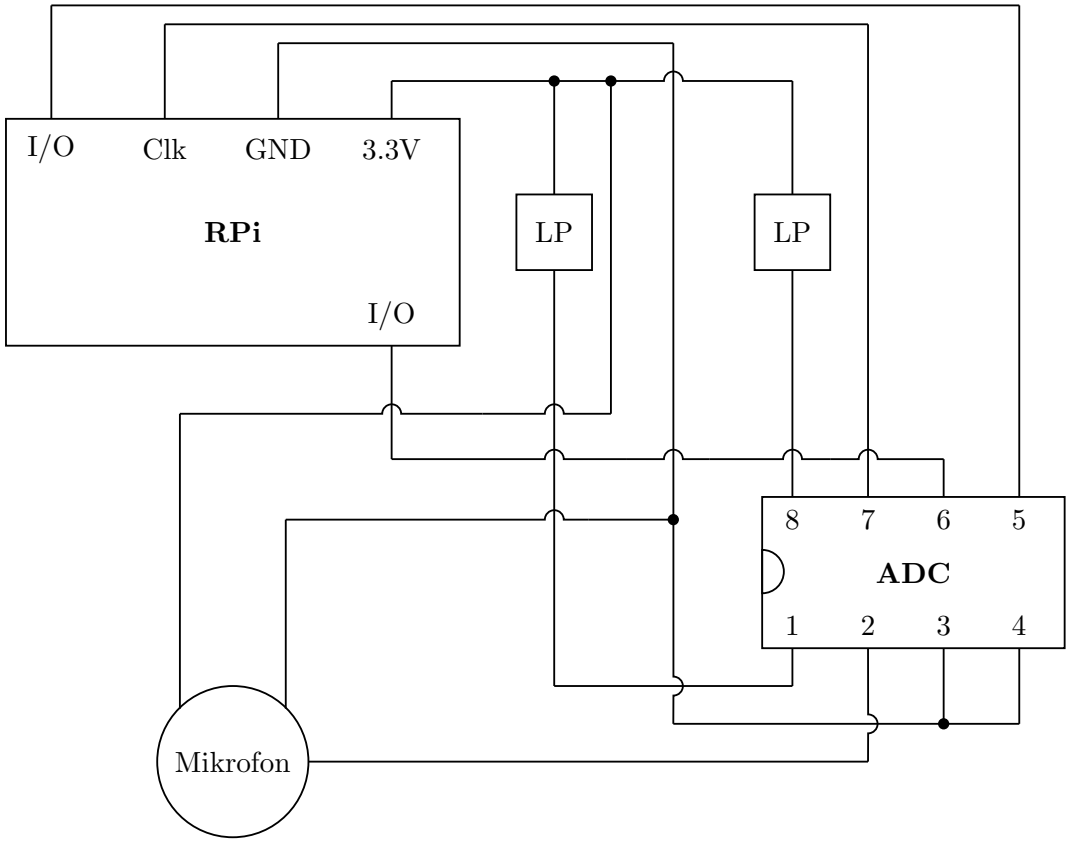


Figur 6: Blokkskjema av tilkobllingen mellom RPi og breadboardet via en Pi-Wedge Breakout board. Mikrofonene og instrumenteringssystemet plasseres på breadboardet og kobles til Pi-Wedge brettet.

Raspberry Pien forsyner ADCen med en forsyningspenning $V_{DD} = 3.3V$, i henhold til databladet til MCP601 som tilater en forsyningspenning mellom 2.7 og 6.0V [6]. Videre kobles de forskjellige portene på ADCen til sine respektive motparter på Pien i henhold til Pinouten til databladet til Pi-Wedge brettet [7]. Mikrofonene, som operer med en forsyningspenning mellom 2 og 5V [8], blir koblet til Piens 3.3V forssyningspenning. Portkoblingen mellom en mikrofon, en ADC og Pien er vist i Figur 7.

På grunn av tiltakene iverksatt i forbindelse med den pr april 2020 pågående Covid-19 pandemien har det ikke vært mulig å få et bilde av det realiserte systemet.

Samplingsfrekvensen som blir brukt til å sample lydsignalene fra mikrofonene er $f_s = 31250$Hz. Dette betyr at systemet ikke kan håndtere lydsignaler med frekvens høyere enn $\frac{f_s}{2} = 15625$ Hz, hvis ikke vil aliasing oppstå. Dette er derimot ikke noe problem ettersom lyden som produseres av at klapp er relativt lavfrekvent og mye mindre enn 15kHz. Videre oppsamples signalet med en faktor 5 for å gi større presisjon i beregningene. Mikrofonene plasseres slik at de danner en likesidet trekant i hht. Figur 4. Lengden på sidene i trekanten settes til 6 cm slik at mikrofonnettverket får plass på breadboardet. Komponentverdiene til dekoblingsnettverket vist i Figur 3 velges med hensyn til teorien diskutert i kapittel 2.1. Verdiene er gjengitt i Tabell 1.



Figur 7: Sammenkoblingsdiamgram av ADCen til Pien. I/O inngangene er pins på Pien som tar i mot samplet data. Under realiseringer benyttes et Pi-Wedge breakout board til å koble sammen breadboardet til Pien, men er utelatt for å simplifisere diagrammet.

Tabell 1: Kompomentverdier brukt under realiseringen av lavpassfilteret.

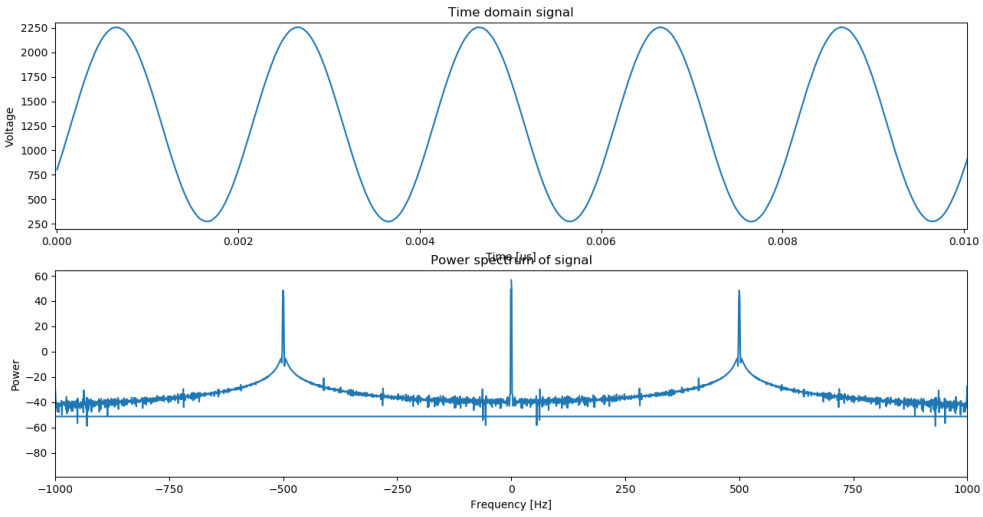
Navn	**Verdi**
C_1	100μF
C_2	100μF
C_3	100nF
L_1	1mH

4 Resultater

De samplede dataene som ble lagret på Raspberry Pien blir lastet ned på en ekstern datamaskin for behandling og beregning av innfallsvinkel. Python ble brukt for all behandling og beregning av og på data. Python koden brukt er vedlagt i Vedlegg A og Vedlegg B.

Testing av instrumenteringssystem

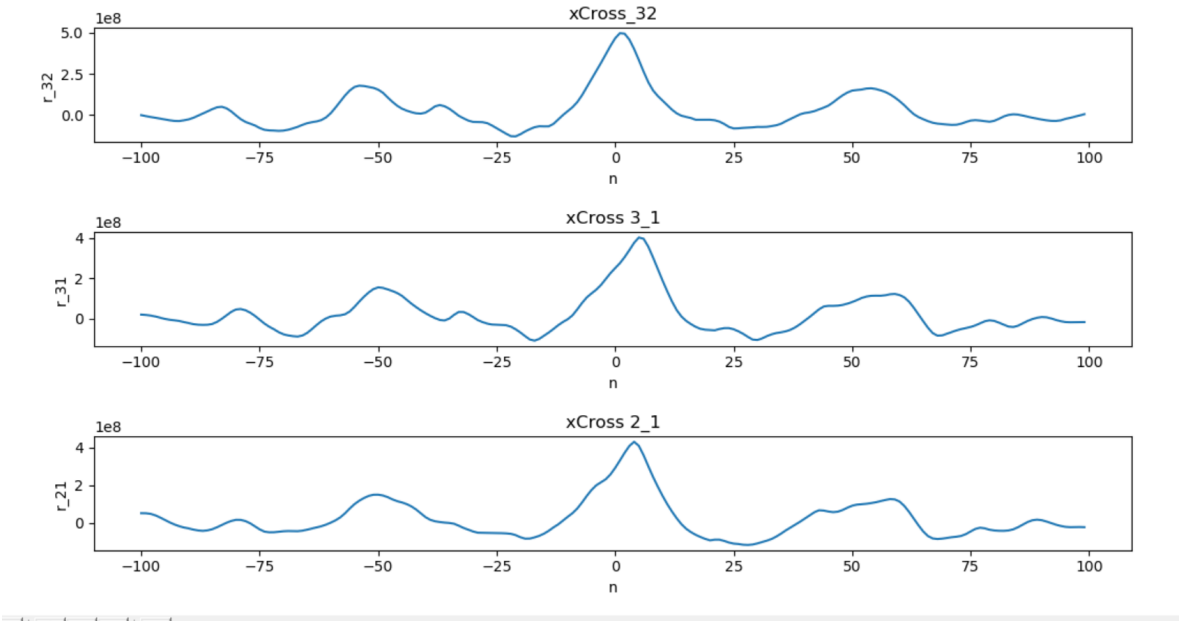
For å verifisere at instrumenteringssystemet fungerte som ønsket ble et rent 500Hz sinussingnal fra en signalgenerator sendt inn på ADCene og samplet med en samplingsfrekvens $f_s = 31250$Hz. Plots av det samplede signalet fra en av ADCene er vist i tidsdomenet og frekvensdomenet (ved sin Fouriertransform) i Figur 8. Resultatene fra denne enkle testen ga tilfredsstillende svar på at instrumenteringssystemet fungerte som ønsket.



Figur 8: Plot av en samplet 500Hz sinussignal i tidsdomenet (øvre plot) og frekvensdomenet (nedre plot). Samplet med en frekvens $f_s = 31250$Hz.

Måling av vinkel

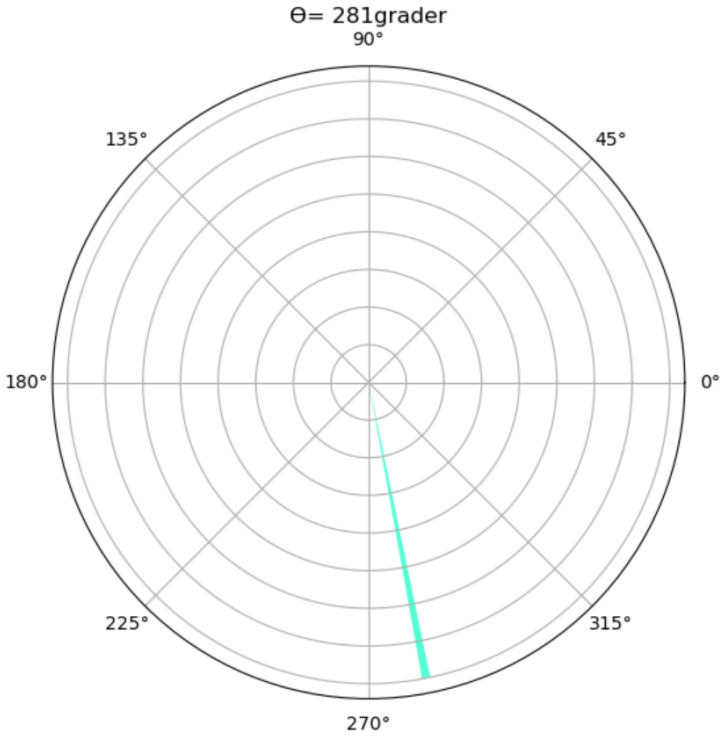
Det akustiske lokalisjonssystemet testes ved å klappe 1m unna sentrum av trekanten med mikrofoner, for og så sammenlikne den beregnede innfallsvinkelen, θ, med den faktiske vinkelen. Under produksjonen av lydsignalet fra et klapp beveges begge hendene omtrent 5cm ut fra startposisjonen. Med grunnleggende geometri gir dette, i 1 meters avstand fra målesystemet sentrum, en usikkerhet i referansevinkelen på 6°. 10 vinkler gjevnt fordelt på 360° rundt systemet blir målt. For hver vinkel blir det gjort 5 stk målinger. Figur 9 viser de beregnede krysskorrelasjonene fra en av målingene av vinkelen 288° relativt til x-aksen fra Figur 4.



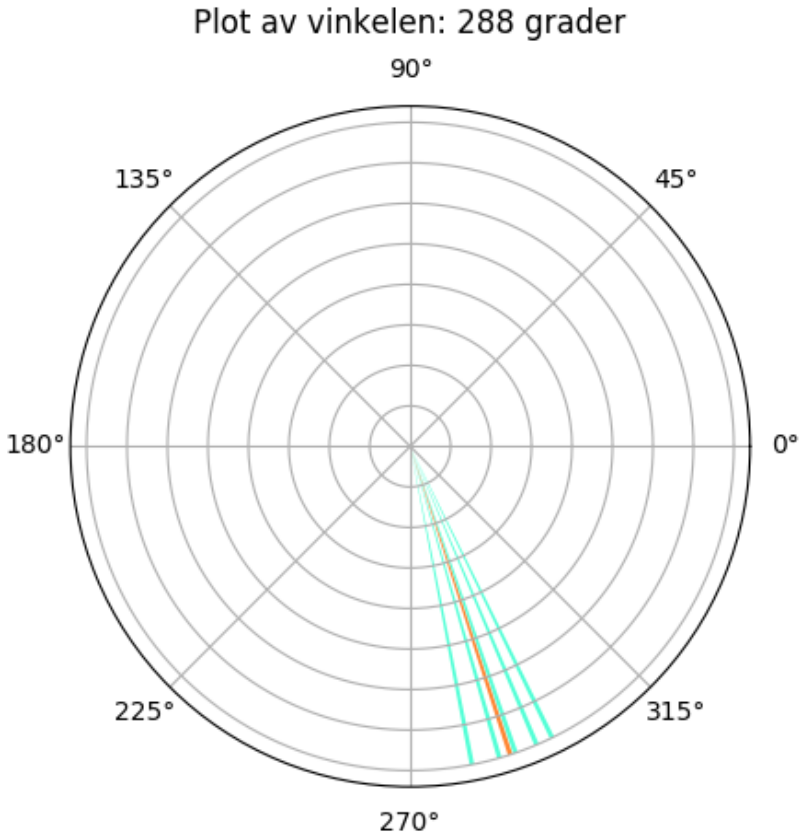
Figur 9: Krysskorrelasjonene r_{32}, r_{31} og r_{21} fra Figur 4 under målingen av vinkelen 288°.

De forkskjellige tidsdifferansene, n_{ji}, finnes så, som skrevet i teoridelen, ved å undersøke ved hvilken n hver av krysskorrelasjonene har sitt maksimum. I dette tilfellet, figur 9, hvor den faktiske innfallsvinkelen er 288°, er tidsforsinkelsene: n_{32}=1 sample, n_{31}=5 sampler og n_{21}=4 sampler. For et lydsignal med innfallsvinkel 288° er det naturlig at tidsdifferansen er minst mellom mikrofon 2 og 3, mens den for mikrofon 3 og 1 og mikrofon 2 og 1 vil være relativ lik, men større sammenliknet med mikrofon 3 og 2. Tar man dette i betrakning, gir verdiene for antall sampler tidsdifferanse god mening. Ved å sette disse verdiene inn i ligning (16), gir dette at den beregnede innfallsvinkelen, θ, blir -79° som tilsvarer 281°, vist i Figur 10. Figur 11 viser totalt 5 målinger for 288°, beregnet på tilsvarene måte.

Tabell 2 gjengir resultatene fra samtlige målinger, samt deres gjennomsnittsverdi og standardavvik. Fra tabellen framkommer det at vinkelestimatets gjenomsnittlige avvik er 1.87°. Det gjennomsnittlige standardavviket er 4.29°.



Figur 10: Beregnet vinkel $\theta = 281°$



Figur 11: Plot av 5 målinger av vinkelen 288°. Vinkelen 288° er markert med oransje og er ikke en del av målesettet.

Tabell 2: Fem målinger av 10 forskjellige vinkler, samt deres gjennomsnittsverdi og standardavvik.

Vinkel	**Måling 1**	**Måling 2**	**Måling 3**	**Måling 4**	**Måling 5**	**Gjennomsnitt**	**Standardavvik**
0°	0.09°	8.41°	2.25°	9.56°	-5.09°	3.044°	6.06°
36°	38.48°	29.02°	37.91°	32.85°	37.54°	35.16°	4.10°
72°	70.62°	74.9°	72.68°	75.52°	74.33°	73.61°	1.98°
108°	108.77°	105.17°	100.78°	111.62°	112.8°	107.83°	4.93°
144°	146.51°	142.19°	141.64°	145.68°	142.23°	143.65°	2.26°
180°	181.57°	179.19°	181.72°	187.85°	184.54°	182.97°	3.32°
216°	224.3°	220.64°	224.27°	220.69°	221.66°	222.31°	1.85°
252°	259.53°	266.85°	252.34°	245.48°	254.54°	255.75°	8.00°
288°	286.11°	289.61°	281.47°	296.03°	293.26°	289.30°	5.76°
324°	329.59°	322.6°	333.19°	333.65°	326.64°	329.13°	4.64°

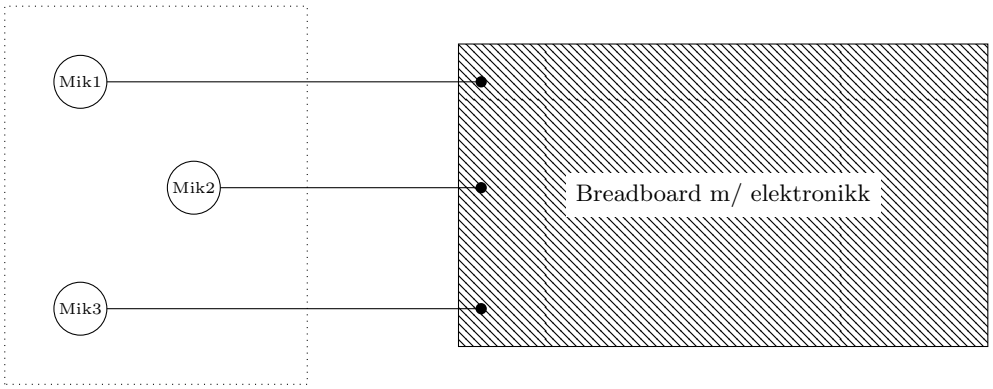
5 Diskusjon

For å vurdere ytelsen til instrumenteringssystemet hadde det vært hensiktsmessig å studere et plot av frekvensresponsen til lavpassfilteret som ble diskutert i 2.1. Et slikt plott ble ikke produsert under første testing av systemet. Av hensyn til Covid-19 situasjonen har systemet i etterkant blitt utilgjengelig og det er følgelig ikke mulig å produsere et slikt plot i etterkant. Ytelsen til intstrumenteringssystemet kan likevel vurderes utifra resultatene fra den enkle testen beskrevet i 4. Plottet av det samplede sinussignalet vist Figur 8 viser at instrumenteringssystemet er i stand til å lagre og gjenskape signalet uten tap av informasjon. Det kan altså konkluderes med at instrumenteringssysteme yter godt nok til at feilkilder i målingene hovedsaklig skyldes andre faktorer.

For å drøfte hvor godt systemet estimerer den faktiske vinkelen betraktes de grunnleggende egenskapene til en estimator, gjennomsnittlig avvik og standardavvik. Fra estimeringsteori er det kjent at en god estimator helst er forventningsrett, hvilket vil si at det gjennomsnittlige avviket er lik 0. Det også ønskelig at en god estimator har så lite standardavvik som mulig. Fra testene som ble gjort kom det fra at det gjennomsnittlige avviket er $1.87°$ og det gjennomsnittlige standardavviket er $4.29°$. Tatt i betraktning at usikkerheten til referansevinkelen ble anslått til å være $6°$ er dette et svært godt resultat.

En svakhet med systemet er måten det testes på. Det å bruke et klapp som lydkilde er i seg selv ikke en dårlig løsning, men det blir krevende å klappe fra nøyaktig samme avstand og eksakte vinkler. En stor del av avvikene skyldes antakelig dette. En annen svakhet ved testingen er at den ble gjort i et rom hvor andre gjorde tilsvarende tester. Med andre ord var systemet utsatt for ekstern støy. Ideelt sett burde testingen vært gjort i et rom fullstendig fritt for andre lydkilder enn det lydssignalet som var ønsket å detektere.

Videre blir også nøyaktigheten på målingene påvirkete av at oppsettet av mikrofonene blir begrenset ettersom de må passe inn på breadboardet. Som skrevet i teorien oppnår vi større presisjon i beregningene ved å la avstanden mellom mikrofonene, d, være så stor som mulig. På et breadboard vil denne størrelsen være begrenset. I tillegg til dette, er det også krevende å gjenskape en fullstendig likesidet trekant, som i figur 4, på et breadboard. Mikrofonene kan ikke plasseres hvor som helst ettersom de må plasseres i koblingspunktene på breadboardet. Av den grunn er det ikke sikkert at det i hele tatt er fysisk mulig å lage en eksakt likesidet trekant med mikrofoner på et breadboard. En forbedring til dette ville vært å bruke et større breadboard. Å bruke mikrofoner som kan plasseres fritt og kobles til breadbordet med ledninger vil også kunne en større trekant. En illustrasjon av dette er vist i Figur 12. Dette vil dog kreve relativt lange ledninger som kan kobles til breadboardet. Dette kan være vanskelig å opprive, tillegg til at lange ledninger ikke er ideelt med tanke på støy.



Figur 12: Oppsett med mikrofonen plassert utenfor breadboardet og kobles med eksterne ledninger. Vil kunne gi større trekant enn om de kobles rett på brettet.

6 Konklusjon

I denne rapporten har design og implementering av et akustisk lokalisjonssystem som beregner vinkelen på et innkommende lydsiganl blitt diskutert. Systemet består av tre mikrofoner som tar opp det innkommende lydssignalet. I tilegg består det av et instrumenteringssystem som sampler lydsignalene og lagrer dem digitalt. Det er biltt brukt en Raspberry Pi 3B til å lagre dataene. Vinkelen har blitt beregnet ved å sammenlinkne krysskorrelasjonen mellom lydsignalene til mikrofonene. Dette er blitt gjort med et Python script.

Resultatene fra testingen av systemet gir at det gjennomsnittlige avviket fra referansevinkelen (vinkelen som blir forsøkt estimert) er 1.87°. Det gjennomsnittlige standardavviket for alle målinger av alle vinklene er 4.29. Dette er et svært god resultat tatt i betraktning av at usikkerheten til referansevinkelen ble anslått til 6°. Instrumenteringssystemet yter i stor nok grad til at feilkildene hovedsaklig kommer fra usikkerhet i målemetoden og liten avstand mellom mikrofonene.

Referanser

[1] *Mikrofon - SNL*. Store Noreske Leksikon. URL: https://snl.no/mikrofon.

[2] *Lyd - SNL*. Store Norske Leksikon. URL: https://snl.no/lyd.

[3] *Cross-correlation - Wikipedia*. Wikipedia. URL: https://en.wikipedia.org/wiki/Cross-correlation.

[4] *Finding Dircetion to an Audio Source Using a Microphone Array*. NTNU.

[5] *Upsampling - Wikipedia*. Wikipedia. URL: https://en.wikipedia.org/wiki/Upsampling.

[6] *MCP3201 Datasheet*. Microchip.

[7] *Pi Wedge B Plus$_v$*11. Sparkfun.

[8] *ELECTRET MICROPHONE AMPLIFIER - MAX4466*. Adafruit.

A Kode til plotting og beregning av vinkel

```
import numpy as np
import matplotlib.pyplot as plt
import scipy as sci
import importdata as ipd
import buggesmatteland as bml
import getangle as ga

# Plot innstillinger
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.4, hspace=0.4)


# Systemspesifikasjoner for realisering av systemet:
x_12 = 0.065
x_13 = 0.065
x_23 = 0.06
a = 0.0562916512
f_s = 31250

channels = 5

# Filnavn til samplet data
path = 'radarData/radar72.bin'

x = np.linspace(-np.pi, np.pi, 201)

[nomTp, rawData] = ipd.raspi_import(path,channels)

rawData_up = ipd.preProc(rawData,5)

d1 = rawData_up[100:,0]
d2 = rawData_up[100:,1]
d3 = rawData_up[100:,2]
d4 = rawData_up[100:,3]
d5 = rawData_up[100:,4]

plt.subplot(5, 1, 1)
plt.title("Raw data from ADC 1")
plt.xlabel("Sample")
plt.ylabel("Conversion value")
plt.plot(np.arange(len(d1))*32e-6, d1)

plt.subplot(5, 1, 2)
plt.title("Raw data from ADC 2")
plt.plot(np.arange(len(d2))*32e-6, d2)
```

```
45  plt.xlabel("Sample")
46  plt.ylabel("Conversion value")
47
48  plt.subplot(5, 1, 3)
49  plt.title("Raw data from ADC 3")
50  plt.plot(np.arange(len(d3))*32e-6, d3)
51  plt.xlabel("Sample")
52  plt.ylabel("Conversion value")
53
54  plt.subplot(5, 1, 4)
55  plt.title("Raw data from ADC 4")
56  plt.plot(np.arange(len(d4))*32e-6, d4)
57  plt.xlabel("Sample")
58  plt.ylabel("Conversion value")
59
60  plt.subplot(5, 1, 5)
61  plt.title("Raw data from ADC 5")
62  plt.plot(np.arange(len(d5))*32e-6, d5)
63  plt.xlabel("Sample")
64  plt.ylabel("Conversion value")
65
66  plt.show()
67
68  # Henter rekker for corsscorr
69  seq1 = rawData[5:,0]
70  seq2 = rawData[5:,1]
71  seq3 = rawData[5:,2]
72
73  # Trekke fra DC
74  sum_seq1 = 0
75  sum_seq2 = 0
76  sum_seq3 = 0
77
78  for sample in seq1:
79      sum_seq1 += sample
80  for sample in seq2:
81      sum_seq2 += sample
82  for sample in seq3:
83      sum_seq3 += sample
84
85  mean_seq1 = sum_seq1 / len(seq1)
86  mean_seq2 = sum_seq2 / len(seq2)
87  mean_seq3 = sum_seq3 / len(seq3)
88
89  seq1_mod = []
90  seq2_mod = []
91  seq3_mod = []
```

```
92
93  for sample in seq1:
94      new = sample - mean_seq1
95      seq1_mod.append(new)
96  for sample in seq2:
97      new = sample - mean_seq2
98      seq2_mod.append(new)
99  for sample in seq3:
100     new = sample - mean_seq3
101     seq3_mod.append(new)
102
103
104 # Plotter samples
105 plt.subplot(3, 1, 1)
106 plt.title("seq 1")
107 plt.xlabel("bla")
108 plt.ylabel("bla")
109 plt.plot(seq1_mod)
110
111 plt.subplot(3, 1, 2)
112 plt.title("seq 2")
113 plt.xlabel("bla")
114 plt.ylabel("bla")
115 plt.plot(seq2_mod)
116
117 plt.subplot(3, 1,3)
118 plt.title("seq 3")
119 plt.xlabel("bla")
120 plt.ylabel("bla")
121 plt.plot(seq3_mod)
122
123 plt.show()
124
125 # Beregner xCorr
126 corr_12 = np.correlate(seq1_mod,seq2_mod,'full')
127 corr_13 = np.correlate(seq1_mod,seq3_mod,'full')
128 corr_23 = np.correlate(seq2_mod,seq3_mod,'full')
129
130 # Bruker auto corr til aa finne rett time shift
131 corr_auto = np.correlate(seq3_mod,seq3_mod,"full")
132
133 corr_auto_mod = []
134 for sample in corr_auto:
135     corr_auto_mod.append(sample)
136
137 m = max(corr_auto_mod)
138 i = corr_auto_mod.index(m)
```

```
139
140 print("Mid is located at " + str(i))
141
142 # Alle autocorr har samme midtverdi
143 corr_23_short = corr_23[31144:31344]
144 corr_12_short = corr_12[31144:31344]
145 corr_13_short = corr_13[31144:31344]
146
147 # Python klarer ikke indexer pånumpy lister tydeligvis. Lager nye identiske "vanlige" li
148 corr_23_short_mod = []
149 for sample in corr_23_short:
150     corr_23_short_mod.append(sample)
151
152 corr_13_short_mod = []
153 for sample in corr_13_short:
154     corr_13_short_mod.append(sample)
155
156 corr_12_short_mod = []
157 for sample in corr_12_short:
158     corr_12_short_mod.append(sample)
159
160 #Regner ut antall samples time delay
161 print("*----------------------------------------*")
162
163 print("The lengt of corr_23_short is " + str(len(corr_23_short)))
164 print("The max value of corr_23 is "+ str(max(corr_23_short)))
165 print("It is located at index " + str(corr_23_short_mod.index(max(corr_23_short_mod))))
166 print("That corresponds to a delay of " + str(corr_23_short_mod.index(max(corr_23_short_
167
168 print("*----------------------------------------*")
169
170 print("The lengt of corr_13_short is " + str(len(corr_13_short)))
171 print("The max value of corr_13 is "+ str(max(corr_13_short)))
172 print("It is located at index " + str(corr_13_short_mod.index(max(corr_13_short_mod))))
173 print("That corresponds to a delay of " + str(corr_13_short_mod.index(max(corr_13_short_
174
175 print("*----------------------------------------*")
176
177 print("The lengt of corr_12_short is " + str(len(corr_12_short)))
178 print("The max value of corr_12 is "+ str(max(corr_12_short)))
179 print("It is located at index " + str(corr_12_short_mod.index(max(corr_12_short_mod))))
180 print("That corresponds to a delay of " + str(corr_12_short_mod.index(max(corr_12_short_
181
182 print("*----------------------------------------*")
183
184 # Plotter alle xCorr
185 plt.subplot(3, 1, 1)
```

```
186 plt.title("xCross 2_3")
187 plt.xlabel("bla")
188 plt.ylabel("bla")
189 plt.plot(range((-100),(-100)+len(corr_23_short)),corr_23_short)
190
191 plt.subplot(3, 1, 2)
192 plt.title("xCross 1_3")
193 plt.xlabel("bla")
194 plt.ylabel("bla")
195 plt.plot(range((-100),(-100)+len(corr_13_short)),corr_13_short)
196
197 plt.subplot(3, 1, 3)
198 plt.title("xCross 1_2")
199 plt.xlabel("bla")
200 plt.ylabel("bla")
201 plt.plot(range((-100),(-100)+len(corr_12_short)),corr_12_short)
202
203 plt.show()
204
205
206 #plt.subplot(2, 1, 1)
207 plt.title("Corr auto")
208 plt.xlabel("bla")
209 plt.ylabel("bla")
210 plt.plot(range((-31244),(-31244)+len(corr_auto)),corr_auto)
211 plt.show()
212
213
214 # Beregner vinkel
215
216 # Delays
217 n_23 = corr_23_short_mod.index(max(corr_23_short_mod)) - (len(corr_23_short_mod))/2
218 n_13 = corr_13_short_mod.index(max(corr_13_short_mod)) - (len(corr_13_short_mod))/2
219 n_12 = corr_12_short_mod.index(max(corr_12_short_mod)) - (len(corr_12_short_mod))/2
220
221 x_t = bml.determineX(x_12,x_13,x_23, n_12, n_13,n_23, f_s,a)
222 vinkel = bml.calculateAngle(n_21 = n_12, n_31 = n_13, n_32 = n_23)
223 print("x er: " + str(x_t))
224 print("raw vinkel: " + str(vinkel))
225 print("raw deg: " + str(vinkel * 180 / np.pi))
226
227 if x_t < 0:
228     vinkel = bml.calculateAngle(n_21 = n_12, n_31 = n_13, n_32 = n_23) + np.pi
229
230 print("*----------------------------------------*")
231 print("Vinkelen er: " + str(bml.makeAnglePositive(vinkel * 180 / np.pi)) +  " grader.")
232 print("Vinkelen er: " +  str(vinkel) + " radianer.")
```

```
233
234  bml.circlePlot(bml.makeAnglePositive(vinkel * 180 / np.pi))
```

B Kode funksjoner brukt til plotting og beregning av vinkel

```
def determineX(x_12,x_13,x_23, n_12, n_13,n_23, f_s,a):
c = 3*10**8

x_21 = x_12
x_31 = x_13
x_32 = x_23

t_12 = n_12/f_s
t_13 = n_13/f_s
t_23 = n_23/f_s

t_21 = t_12
t_31 = t_13
t_32 = t_23

#sum = (x_12*t_12 + x_13*t_13 + x_21*t_21 + x_23*t_23 + x_31*t_31 + x_32*t_32)
#x = ((2*c) / (9*a**2))*sum

x = (-(np.sqrt(3)*a/3)*t_12)+((np.sqrt(3)*a/3)*t_13)+(np.sqrt(3)*a*t_23)
return x

def calculateAngle(n_21,n_31,n_32):
    arg = (np.sqrt(3)*(n_21 + n_32)) / (n_21 - n_31 - 2*n_32)

    angle = np.arctan(arg)

    return angle

def makeAnglePositive(ang):
    if ang < 0:
        ang = 360 + ang
    return ang

def circlePlot(angle):
    N = 360
    bottom = 0
    max_height = 4

    theta = np.linspace(0.0, 2 * np.pi, N, endpoint=False)
    radii = max_height*np.random.rand(N)
    width = ((2*np.pi) / N) + 0.01

    for i in range(N):
        radii[i] = 0
```

```
45          if i == int(round(angle)):
46              radii[i] = (max_height*1)
47
48      ax = plt.subplot(111, polar=True)
49      bars = ax.bar(theta, radii, width=width, bottom=bottom)
50      ax.set_yticklabels([])
51      ax.set_title("Plot av vinkelen: " + str(int(round(angle))) + " i grader:")
52
53      # Spicy farger
54      for r, bar in zip(radii, bars):
55          bar.set_facecolor(plt.cm.jet(r / 10.))
56          bar.set_alpha(0.8)
57
58      plt.show()
```