NTNU
Det skapende universitet

Radar målfølging

Jan Sigurd Karlsen

Master i teknisk kybernetikk
Oppgaven levert: Juni 2008
Hovedveileder: Tor Engebret Onshus, ITK
Biveileder(e): Øivind Ødeskaug, Kongsberg Defence and Aerospace AS

Norges teknisk-naturvitenskapelige universitet
Institutt for teknisk kybernetikk

Oppgavetekst

Oppgaven tar for seg radarer i moderne luftvernsystemer med vekt på målfølging. Punktene nedenfor beskriver de gjøremål som inngår i oppgaven.

- Litteraturstudie om radarsystemer og målfølging skal gjennomføres.

- Alternative prinsipper for målfølging med radar skal beskrives.

- Algoritme for målfølging basert på data fra radar skal utvikles.

- Målscenarioer med forskjellig manøvermiljø skal lages for simulering av målfølging.

Oppgaven gitt: 15. januar 2008
Hovedveileder: Tor Engebret Onshus, ITK

Forord

Dette er en diplomoppgave for undertegnede student og arbeidet er utført ved Institutt for Teknisk Kybernetikk ved NTNU våren 2008.

Oppgaven tar for seg radarer i Kongsberg Defense & Aerospace (KDA) sitt luftvernssystem med vekt på målfølging.

Institutt for Teknisk Kybernetikk har tidligere bidratt med prosjekt og masteroppgave rundt luftvernssystemet hvor fusjonering av distribuerte radarer ble behandlet. Dette har vekket interesse for hvordan målfølging kan forbedres.

Rapporten beskriver en løsning på denne problemstillingen ved å foreta målfølging med forskjellige metoder tilpasset tilgjengelige måldata. Neste skritt i prosjektet vil være å utbedre algoritmen for målfølging og å benytte det utviklede systemet i målfølging med fusjonering av distribuerte radarer.

Jeg vil takke min faglærer Tor Onshus for gode råd og hjelp underveis i prosjektet. Ved KDA vil jeg takke min veileder Svein Fagerlund for en utfordrende oppgave og for hjelp både ved besøk og på telefon.

Trondheim, 3/6-2008

Jan Sigurd Karlsen

Oppsummering og konklusjon

Kongsberg Defense & Aerospace (KDA) benytter radarer av typen ”phased array” i deres luftvernsystemer. På bakgrunn av nødvendige egenskaper som elektronisk styring av radarstråle tas denne radartypen også i bruk i denne oppgaven. En tilgjengelig signalkilde fra radaren er SNR data som brukes for utbedring av målfølgingen.

På bakgrunn av valgt radar er det fremlagt beskrivelser for hvordan oppstart og avslutning av målfølging kan gjennomføres.

Metoder for generering av observasjonsdata og observasjonsstøy er tilpasset SNR data innhentet fra radaren. Observasjonsdata er generert ved monopulsbasert og vinkelbasert metode i form av *”Amplitude Comparison Monopulse (ACM)”* og *”Sequential Lobing (SL)”*. Observasjonsstøy er generert ved Bartons, NOAH’s og vinkelstøybasert metode, der sistnevnte er utledet fra SL algoritmen. Samtlige metoder er beskrevet matematisk og funksjonelt.

To forskjellige typer algoritmer basert på Kalman-filter (KF) er brukt for estimering. Disse benytter forskjellige metoder for å følge manøvrerende mål. Utvidet KF, *”Extended Kalman-filter (EKF)”*, er satt i sammen med en manøverdeteksjonsalgoritme og Samvirkende multiple modeller, *”Interacting Multiple Modell (IMM)”*, filter benytter sannsynlighetsberegninger for å skille mellom en hastighetsmodell og en akselerasjonsmodell.

Sammensetninger av målfølgealgoritmer basert på modeller for observasjonsdata, observasjonsstøy og estimering er implementert i Matlab 2007a og simulert i manøvrerende miljø med ulik grad av akselerasjon og manøvrerbarhet.

Det er vist at det mulig å generere tilstrekkelig nøyaktige observasjonsdata ved å benytte SNR data fra radar med vinkelbasert metode. Monopulsbasert metode gir større avvik i form av bias og er dermed uegnet for bruk med filtrene som brukes for estimering.

Observasjonsstøy lar seg tilnærme med tilstrekklig nøyaktighet ved NOAH’s og vinkelstøybasert metode. Bartons metode avhenger for lite av differansen i SNR dataene og genererer dermed for unøyaktige støydata for estimatorene.

På bakgrunn av resultater fra ulike projeksjoner og statistiske beregninger er det vist at EKF og IMM filteret fungerer godt som estimatorer. IMM filteret gir noe bedre resultater ved manøvre grunnet bedre manøverhåndteringsmetoder. Allikevel fremgår det av konsistenstesten at IMM filteret er optimistisk ved harde manøvre, noe som kan skyldes en dårlig tilpasset akselerasjonsmodell.

Både EKF og IMM filteret ble realisert med variabel målestøy i form av generert observasjonsstøy, og dette viste seg å gi vesentlig større nøyaktighet i estimatene.

Det er altså vist at en fullstendig målfølgealgoritme lar seg realisere med tilgjengelige SNR data fra radarer av typen ”phased array”, og at nøyaktigheten kan forbedres ved å innføre variabel målestøy basert på de samme SNR dataene.

Innholdsfortegnelse

1 Innledning

Dette kapitlet gir en kortfattet oversikt over de tema som inngår i oppgaven med motivasjon, målsetning og beskrivelse av tidligere rapporter om emnet. Til sist følger en oversikt over rapportens struktur med beskrivelse av hvert kapittel i tillegg til forklaring av definisjoner.

1.1 Motivasjon

Kongsberg Defense & Aerospace (KDA) er et verdensledende selskap innen utvikling av luftvernsystemer. En viktig del av disse systemene er radarer som overvåker luftrommet og detekterer fiendtlige fly eller raketter. For at et motangrep skal kunne iverksettes må det detekterte målet følges. Nøyaktigheten av målfølgingen er avgjørende for om motangrepet er vellykket eller ikke. Dette utgjør motivasjonen for denne oppgaven. Altså å studere hvilken nøyaktighet som det er mulig å oppnå ved målfølging.

1.2 Målsetning

Oppgaven baserer seg på KDA sine radarsystemer og tar sikte på å vurdere nye løsninger for målfølging. Med sammensetninger av forskjellige metoder er målsetningen å vise at tilgjengelige SNR data fra radarmålingene kan forbedre målfølgingens nøyaktighet.

Dette krever i all hovedsak en tredelt algoritme hvor det først genereres observasjonsdata og observasjonsstøy på bakgrunn av SNR data. Denne informasjonen kan så tilføres et filter som estimerer posisjon, hastighet og akselerasjon. Vanligvis holdes målestøyen i et slikt filter konstant, basert på spesifikasjon av radarens målenøyaktighet. Men i denne oppgaven benyttes den genererte observasjonsstøyen som variabel målestøy for å undersøke om nøyaktigheten blir bedre.

1.3 Tidligere arbeid

Tor Erik Strand skrev i 2001 masteroppgave for ”Data Fusjon I Distribuert Luftvern System”, [1, Strand T. E.]. Dette arbeidet tok for seg fusjonering av estimater i distribuerte luftvernsystemer. Altså estimering med radarsignaler fra flere antenner spredd utover et større geografisk område.

Radarsignalene ble realisert ved virkelig målbane med tillagt hvit støy. Alpha/Beta-filter, Utvidet Kalman-filter, *”Extended Kalman filter (EKF)”*, og Multiple samvirkende modeller filter, *”Interacting multiple modell (IMM)”* filter inngikk som estimatorer.

Det ble konkludert med at de to sistnevnte filtrene fungerte best og svært likt ved rettlinjede målbevegelser. IMM filteret var bedre ved manøvre. Videre ble det konkludert med at fusjonering gav vesentlig reduksjon i usikkerheten av estimatene.

1.4 Rapportstruktur

Innholdet i rapportens kapitler blir her kortfattelig beskrevet.

1. Innledning: Kort innføring i oppgaven med beskrivelse av motivasjon og målsetning.

2. Bakgrunn: Opphavet til oppgaven utredes.

3. Teori: Teori som følger av litteraturstudiet presenteres**.**

4. Utvikling av systemet: Metoder og algoritmer fremlegges matematisk og forklares.

5. Simulering og resultater: Resultater fra simulering fremlegges og kommenteres.

6. Diskusjon: Resultater diskuteres og forslag gis for utbedring av systemet.

1.5 Definisjoner

ACM	Amplitude Comparison Monopulse
ADT	Automatic detection and track
EKF	Extended Kalman Filter
FDC	Fire Distribution Center
IMM	Interacting Multiple Model
KDA	Kongsberg Defense & Aerospace
KF	Kalman Filter
MSE	Mean Square Estimate
NEES	Normalized Estimation Error Squared
NOAH	Norwegian Adapted Hawk
NTNU	Norges Teknisk-Naturvitenskapelige Universitet
PCM	Phase Comparison Monopulse
Radar	Radio Detection and Ranging
RCS	Radar Cross Section
RMS	Root Mean Square
SL	Sequential Lobing
SNR	Signal to Noise Ratio
TWS	Track While Scan

2 Bakgrunn

KDA har siden begynnelsen av 70-tallet utviklet våpenkontrollsystemer for det norske forsvaret og et bredt internasjonalt marked. Disse luftvernsystemene er som regel basert på et nettverk av radarer i kombinasjon med andre forsvarskomponenter i et distribuert bakke til luft forsvarssystem. En av samarbeidspartnerne til KDA er radarleverandøren Raytheon [2, Thales Raytheon Systems] som har utviklet radaren AN/MPQ-64 (Sentinel) i figur 1.

Figur 1: Raytheon AN/MPQ-64 radar

Radarsystemet utgjør en viktig del av KDA sine luftvernsystemer ved deteksjon og følging av mål. I dag foretas all målfølging i radarsystemet som sender måldata til et ”Fire Distribution Center (FDC)” for videre prosessering. Denne diplomoppgaven er definert for å se på mulige forbedringer av målfølgealgoritmen basert på bruk av tilgjengelige SNR data.

3 Teori

Denne oppgaven tar som tidligere nevnt for seg problemstillinger rundt radar og målfølging. Teorien rundt disse emnene er omfattende så her vil kun det mest aktuelle for oppgaveløsningen bli presentert. Først blir radar beskrevet generelt, så følger en beskrivelse av radartyper og metoder for målfølging.

3.1 Radar

"Radio Detection and Ranging (*Radar)"* bruker elektromagnetiske pulser for å detektere og lokalisere reflekterende objekter. Avstand til objektet blir beregnet ut ifra hvor lang tid, $T_{R,}$ en utsendt puls bruker på å reise til et mål og tilbake ved å benytte *Doppler effekten*.

Dersom man passeres av en racerbil i høy hastighet endrer lyden seg, "Vrrroooommmm". Det samme prinsippet gjelder for *doppler frekvensen* som benyttes i radarsystemer.

Radarligningen

Radarens evne til å motta et reflektert signal avhenger av støyen som til enhver tid påvirker signalet. Støyen som ligger i samme frekvensområde som det reflekterte signalet utgjør et stort problem hvis det benyttes for lav forsterkningen under sending. Vanligvis finnes en nedre grense for hvor sterkt det reflekterte signalet må være for å detekteres. Ved å justere opp forsterkningen vil det reflekterte signalet lettere detekteres. Den beste måten å fremstille dette forholdet på er ved SNR (Signal to Noise Ratio). *Radarligningen* i 3-1 er hentet fra [3, Skolnik, M. I] og tar hensyn til en rekke faktorer som spiller inn på radarens stråling. SNR oppgis ofte i dB.

$$SNR = \frac{P_t G^2 \lambda^2 \sigma}{(4\pi)^3 k T_e B F L R^4} \tag{3-1}$$

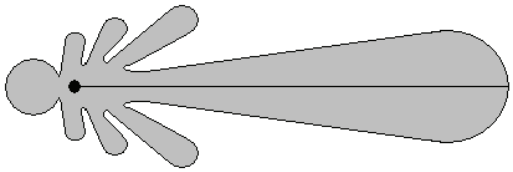
Tabell 1 gir en kort beskrivelse av de variable og konstanter som inngår.

Symbol	Beskrivelse
P_t	Maksimal effekt
G	Antenneforsterkning
T_e	Effektiv støytemperatur
B	Radarens strålebredde
L	Radar tap
R	Målavstand
F	Forholdet mellom støy på inngangen og utgangen av sender/mottaker.
λ	Bølgelengde
σ	Radar Cross Section (RCS)
k	Boltzman konstant

Tabell 1: Variable og konstanter i radarligningen

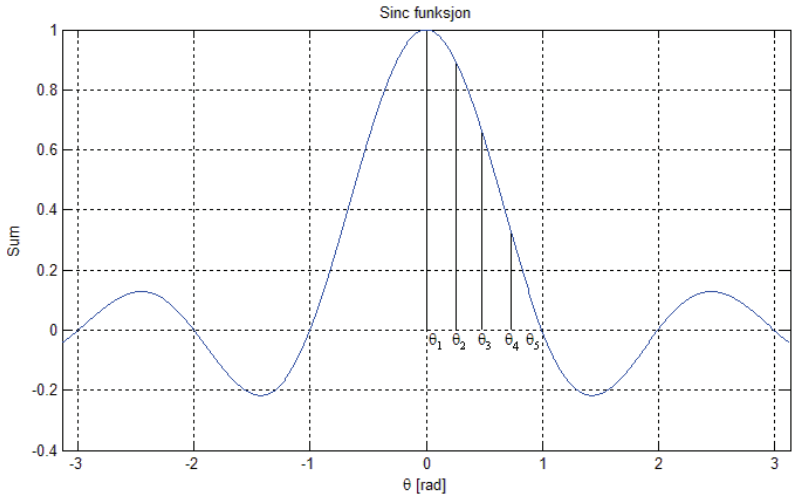
Stråling

Selv om det er ønskelig at radarer konsentrerer strålingen som en rett linje gjennom rommet er dette vanskelig og heller urealistisk å oppnå. Strålingen spres i utgangspunktet i alle retninger så den kan i beste fall til dels samles i en retning.



Figur 2: Stråling fra retningsorientert antenne

Figur 2 gir et todimensjonalt bilde på stråleeffekten fra en retningsorientert antenne. Den største og ytterste sløyfen kalles *hovedlobe* og de på sidene kalles *sidelober*. Senterlinja i hovedloben kalles vanligvis ”Boresight” men vil heretter kun kalles *senterlinje*. Denne formen for stråling refereres ofte til som diffraksjon og kan enklest beskrives ved en lineær oppstilling av sender-/mottakerelement. I figur 3 er det vist et eksempel ved en sinc-funksjon. Dersom $\theta = 0$ vil alle bølgene virke parallelt med samme fase og gi en stor sum feltstyrke. I andre tilfeller hvor $\theta \neq 0$ vil denne summen minke som følge av at bølgene ikke er i fase.



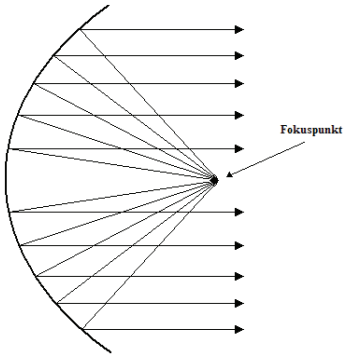
Figur 3: Sinc-funksjon

Tykkelsen på hovedloben blir ofte referert til som *radarstrålebredde*. Ved bruk av tredimensjonal radar må naturligvis strålebredden for asimut og elevasjon være kjent. Denne gis ofte som bredden av hovedloben ved -3 dB.

Antenner

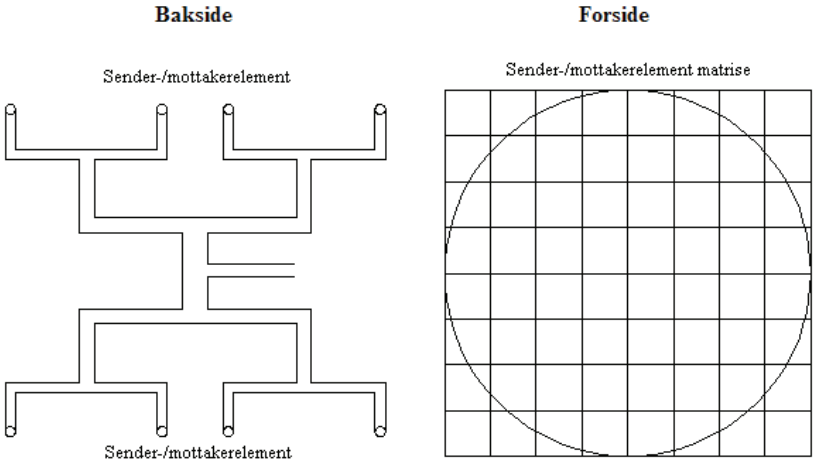
I [4, Stimson, G. W.] presenteres parabolsk reflektor antenne og ”planar array” – antenne som er de to vanligste radarantennene som benyttes i luftvernssystemer. Hver for seg representerer forskjellige prinsipper for bruk av radar.

Parabolske reflektorantenner fungerer ved at det stråles fra et fokuspunkt ut til en paraboloide, slik som vist i figur 4. Paraboloiden er utformet slik at strålingen virker i en og samme retning. En vesentlig konsekvens er også at fasen i vært punkt av planet forblir den samme. Strålingen forplanter seg dermed som en rett stråle gjennom rommet. Hvis det er ønskelig å endre retning på strålen må antennen flyttes i retning av målet.



Figur 4: Parabolsk reflektorantenne

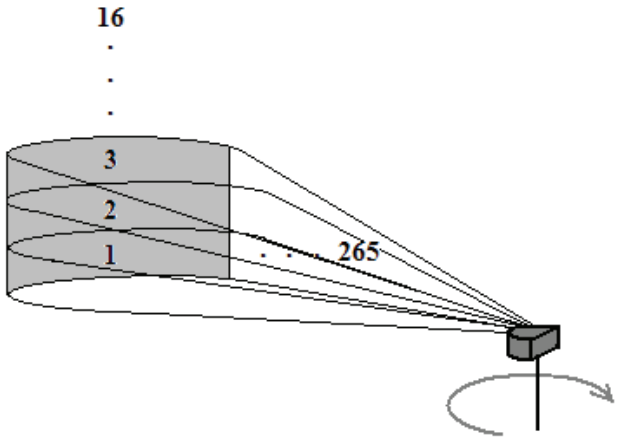
Navnet ”planar array” kommer av radarantennens plane flate med sender-/mottakerelementer oppsatt i hva som kan kalles et ”array” eller en matrise slik som vist i figur 5. Disse elementene sender ut samtidige pulser med stråling slik at det dannes en retningsbestemt stråle. Ved å skifte fasen til elementene kan retningen på strålingen styres uten fysisk å bevege antennen, derav navnet ”phased array”. Dette vil heretter kalles elektronisk styring av stråle. Endring av stråleretning tar mikrosekunder. En ulempe er at store vinkler gir dårligere ytelse som følge av at strålearealet fra antennen reduseres.



Figur 5: "Phased array" – antenne

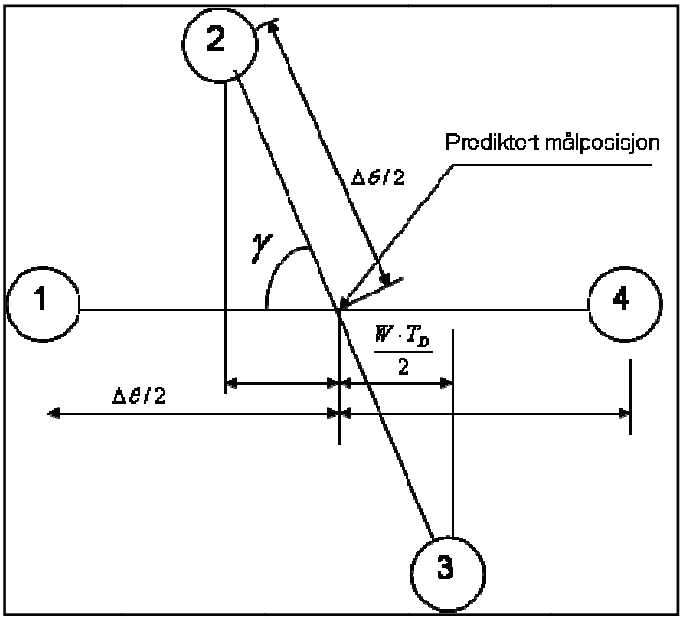
Radaren som arbeidet i denne masteroppgaven bygger på er Raytheon AN/MPQ-64 (Sentinel), [2]. Dette er en roterende "phased- array"-radar som brukes i flere av KDA sine distribuerte luftvernssystemer.

Antennen roteres slik at den dekker 256 sektorer i asimut. Hver av disse sektorene har også en rekke elevasjonsinndelte sektorer. Den laveste elevasjonssektoren har senter 0,75 grader over horisonten deretter følger det 15 sektorer med 1,5 graders mellomrom. I figur 6 er dette illustrert.



Figur 6: Sektordiagram for radarantenne

Dersom et mål detekteres i en av disse sektorene retter radaren fire stråler sekvensielt rundt målet ved elektronisk strålestyring. Figur 7 fra [5, Fagerlund, S.] viser hvordan det tar form. Årsaken til at stråle 2 og 3 ikke er normale på asimutplanet er at antennen har rotert litt imellom de to målingene.



Figur 7: Mønster for fire stråler rundt oppdatert målposisjon

Raytheon har ut ifra denne formasjonen kunnet forutsi plassering og bevegelse av mål slik at målfølging kan oppnås.

3.2 Målfølging

Målfølging er en viktig del av moderne luftvernsystemer. Radaren bestemmer da ikke bare tilstedeværelsen av et objekt men følger også objektet over tid. På denne måten kan radaren angi målbanen. Ved målfølging av flere mål brukes hovedsakelig to typer følgemetoder:

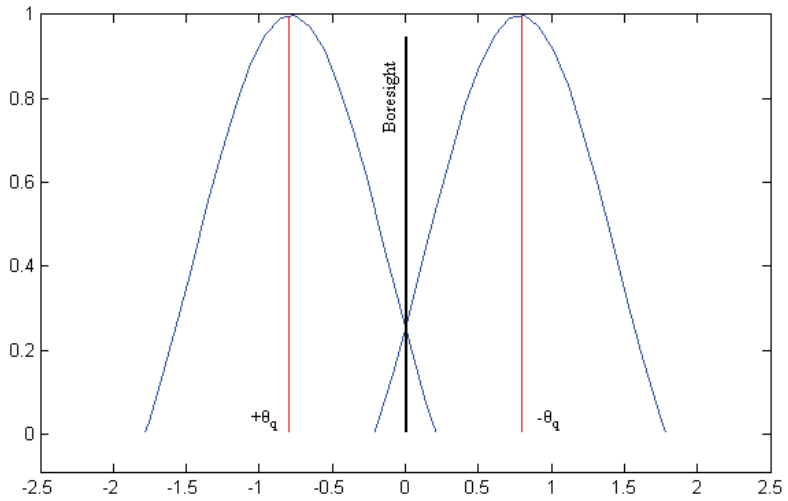
Ved *Automatisk deteksjon og følging, "Automatic detection and track (ADT)"*, styres radaren i åpen sløyfe. Selv om oppdateringshastigheten er lav kan opptil flere tusen mål følges av gangen. Dersom et mål skannes ved flere enn 2 rotasjoner initieres en assosiasjonsalgoritme som følger målet og estimerer hastighet og posisjon.

Elektronisk styrt stråling innebærer at flere mål kan følges sekvensielt med stor hastighet. Nøyaktigheten i posisjonsmålingene er her større og dette gjør målfølgingen langt bedre. AN/MPQ-64 som benyttes av KDA passer utmerket til dette formålet og vil dermed benyttes til å generere o*bservasjonsdata* med følgende metoder.

Vinkelbasert følging

Ved deteksjon vil vinkelkoordinatene til et mål kunne anslås å ligge innen radarstrålebredden. For kontinuerlig å holde målet innen strålebredden er det nødvendig å innhente mer informasjon. I et todimensjonalt system slik som vist i figur 8 er det tilstrekkelig å ha to radarstråler pekende i delt vinkel $-\theta_q$ og $+\theta_q$. Her kan en stråle veksle mellom å peke i hver vinkel eller det kan pekes samtidig i hver vinkel med to stråler. Krysningen mellom lobene fra disse strålene utgjør senterlinjen og målet er å la denne peke mot målobjektet. Når strålingen returneres til mottakeren sammenlignes den med amplituden i de to signalene og ut i fra disse bestemmes målretningen. Det samme kan gjøres tredimensjonalt men da er det nødvendig med ytterligere en eller to ortogonale stråler. De to vanligste formene for vinkelfølging er sekvensiell og konisk lobing.

"Seqential Lobing (SL)" er en teknikk som sekvensielt skifter en stråle mellom to forutbestemte vinkler. Forskjellen i returnert stråleamplitude utgjør en spenningsforskjell og dersom denne er lik null er målet på senterlinjen. Hvis den er ulik null antyder det at målet ligger mot den siden hvor amplituden er størst. Nøyaktigheten begrenses av radarstrålebredden og støyen fra elektronisk og mekanisk skiftemekanisme.



Figur 8: Eksempel på SL

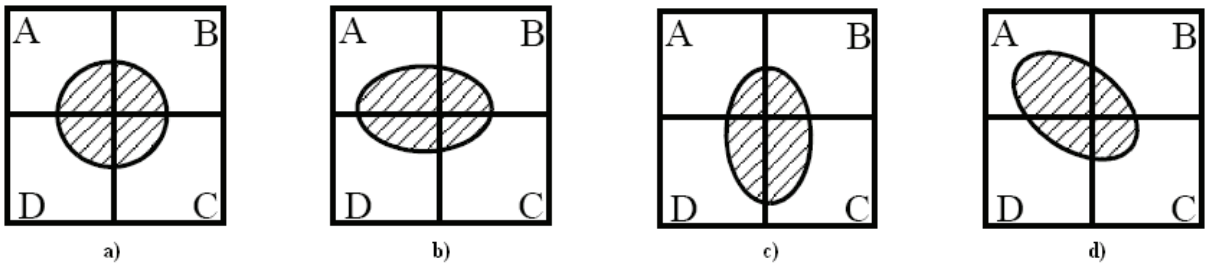
"Conical Lobing (CL)" er en utvidelse av SL. Her roteres loben kontinuerlig rundt senterlinjen i en konstant vinkel. Her vil den returnerte stråleamplituden gi seg utslag som en kontinuerlig varierende spenning. Dette er en lite brukt og komplisert målfølgingsmetode som på grunn av funksjonalitet er uegnet for denne oppgaven.

Monopulsbasert følging

Monopulsbasert følging har store likhetstrekk med SL, men skiller seg ut ved at den benytter flere uavhengige og samtidige stråler for å bestemme asimut og elevasjonsvinkler. Denne metoden for målfølging er mer nøyaktig enn sekvensielt tidsdelte metoder som kun benytter en stråle. Dette skyldes at amplitudevariasjoner som følge av ekko fra målet unngås siden det kun brukes en enkelt puls. De to vanligste formene for monopulsbasert følging er basert på amplitudesammenligning og fasesammenligning.

Amplitudesammenlignings-monopuls "Amplitude Comparison Monopulse, (*ACM)"* benytter fire stråler med lik fase som legges rundt et mål. Den returnerte strålingen avgir amplituder som kan benyttes for å beregne posisjonen i vinkelkoordinater. Dette kan enklest illustreres med diagrammet i figur 9 som er hentet fra [6, Mahafza, B. R. & Elsherbeni, A. Z.]. Her følger en forklaring av tilfellene i figur 9 fra a) til d).

a) Energimengden er jevnt fordelt på kvadrantene og målet ligger da i sentrum.
b) Energimengden i kvadrant A og D gir større utslag og målet ligger dermed mot venstre.
c) Energimengden i kvadrant D og C gir større utslag og målet ligger dermed nedover.
d) Energimengden i kvadrant A gir størst utslag og målet ligger dermed i retning skrått opp mot venstre.



Figur 9: Eksempler på ACM energirespons

Fasesammenlignings-monopuls "Phase-Comparison Monopulse, (PCM)" er en mindre brukt metode for målfølging. Den har det tilfelles med ACM at sum og differanse i asimut og elevasjon innhentes for posisjonsbestemmelse av målet. I motsetning til ACM er det her fasen som endres i strålene og dette gjør denne metoden lite egnet for denne oppgaven da fase data ikke inngår i mottatte signaler.

Estimator

Etter at observasjonsdata er generert innføres algoritmer for å glatte observasjonene. Disse algoritmene estimerer tilstander som posisjon, hastighet og akselerasjon. Eksempler på slike algoritmer er Alpha/Beta filter, Kalman-filter og Multiple Samvirkende Modeller filter. I [1] er det konkludert med at førstnevnte yter betraktelig dårligere enn de to andre, så denne metoden tas dermed ikke med videre i denne oppgaven. I kapittel 4 behandles Kalman-filter og Multiple Samvirkende Modeller filter i detalj.

4 Utvikling av systemet

Ved å se på responsen i mottatte radarsignaler kan det bestemmes om et mål er i søkesonen til radaren. Dersom det dukker opp unormalt store topper i signalet kan dette være en indikasjon på at et mål er i søkesonen og at det bør indikeres på monitor. I de fleste sammenhenger brukes radarer for å detektere mål som fly, raketter og andre fartøy. Det er kjent at disse følger relativt rette baner uten store tilfeldige hopp i posisjon. Hvis en radar først detekterer et mål ved en rotasjon vil den dermed kunne assosiere dette målet ved neste rotasjon. Basert på dette grunnlaget skal det i dette kapitlet utvikles metoder for oppstart, gjennomføring og avslutning av målfølging.

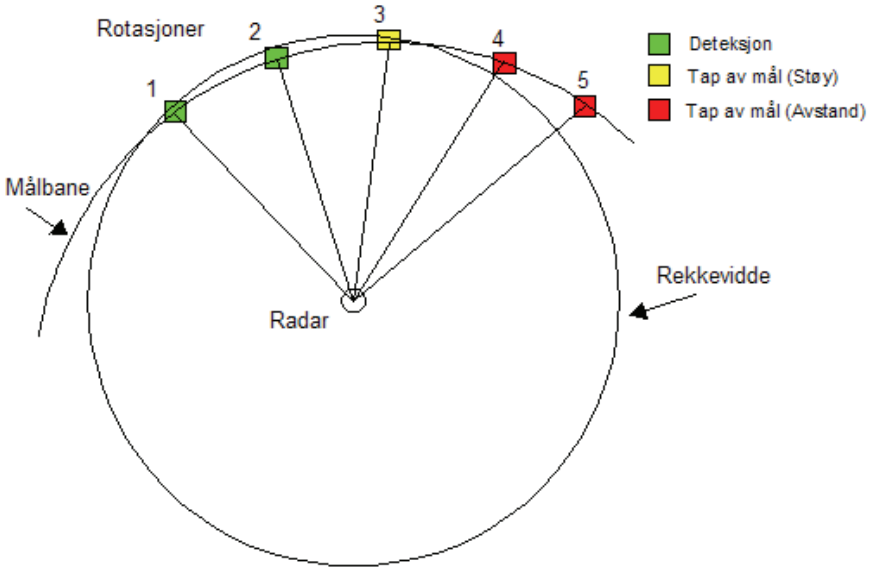
Målfølgingen er først beskrevet matematisk ved sammensatte metoder og deretter implementert med programkode i appendiks B. Hovedfilen i appendiks B.1 utgjør samlepunktet for alle metodene.

4.1 Oppstart og avslutning av målfølging

Radaren, Sentinel, sveiper 360 grader rundt med en vinkelhastighet på 180 grader/s og dekker 256 sektorer i asimut og 16 sektorer i elevasjon. Dersom et mål skulle bevege seg inn i radarens deteksjonsområde vil det med stor sannsynlighet detekteres i nedre sektorer først. Av den grunn sjekkes disse også oftere enn sektorer som ligger høyere i elevasjon. Det går altså ikke mer en 2 sekunder mellom hver gang sektorene nederst i elevasjon sjekkes. Sektorer som ligger over sjekkes kun annen hver gang og sjeldnere desto høyere man går i elevasjon.

Man skiller mellom søk og målfølging. Radaren søker kontinuerlig gjennom alle sektorer uavhengig av om et eller flere mål detekteres. Dersom et mål detekteres vil radaren først iverksette ”backscanning” som rett og slett er elektronisk styring av radarstrålen. Ved neste rotasjon vil søket fortsette som normalt, men ved samme punkt som tidligere vil man sette ut elektronisk styrte radarstråler. På denne måten kan man foreta målfølging ved å fortolke de oppdaterte måledataene. Søket må naturligvis gå kontinuerlig for å holde oversikten over multiple mål. Dersom flere mål detekteres vil man foreta samme prosedyre med elektronisk strålestyring ved hver målposisjon.

I radarsammenheng opereres det ofte med usammenhengende data som følge av forstyrrelser og store målmanøver. Slik sett kan et mål komme inn og falle ut i forskjellige intervaller. For målfølging i denne oppgaven tas dermed følgende regel i bruk. Dersom et mål ikke detekteres ved mer enn tre radarrotasjoner droppes målet, slikt som vist i figur 10.



Figur 10: Avslutning av målfølging

I appendiks C er det tatt med pseudokode av en algoritme for realisering av oppstart og avslutning av målfølgingen. Denne benyttes ikke sammen med simulering av algoritmene som inngår i selve målfølgingen siden dette ikke er hensiktsmessig og heller ikke vesentlig for oppgaven.

4.2 Observasjonsdata

I kapittel 3 ble flere ulike metoder for å generere observasjonsdata presentert og SL og ACM viste seg å være aktuelle alternativer. Disse vil her bli tilpasset de tilgjengelige variablene fra radaren:

- Tidspunkt for måling
- Avstand
- Asimut retning
- Elevasjon retning
- SNR i hver av målingene

Implementering av disse algoritmene kan finnes i appendiks B.4.

Radartypen Sentinel som det tas utgangspunkt i legger fire stråler sekvensielt rundt detektert målposisjon. Forholdet i SNR mellom disse strålene vil så være avgjørende for generering av nye observasjonsdata.

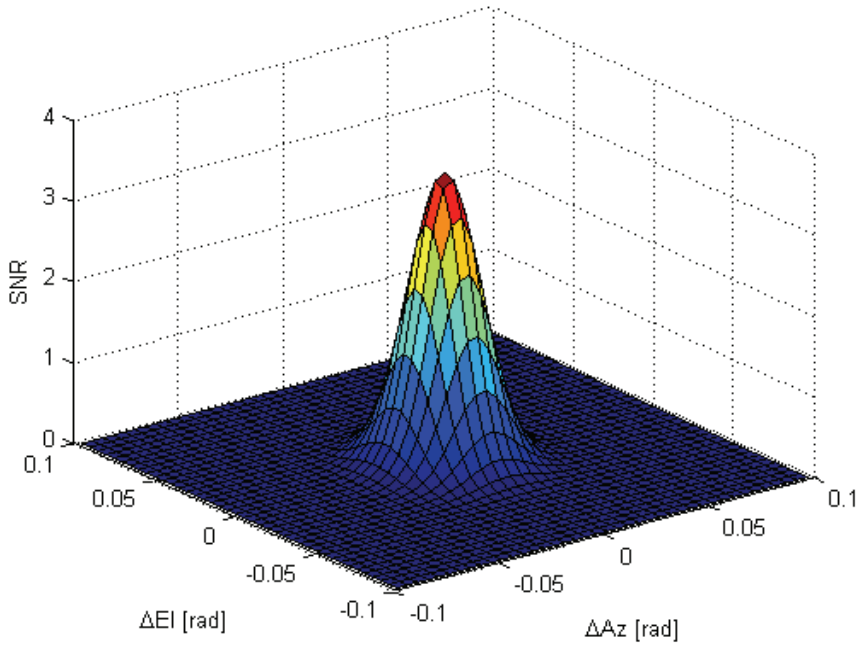
For at dette skal kunne simuleres og testes er det utviklet en modell for SNR i [5] som er vist i ligning 4-1 og 4-2. Denne modellen tar utgangspunkt i vinkelavvik fra målt posisjon i asimut og elevasjon. Tabell 2 beskriver funksjonens variable og konstanter. I appendiks B.2 er programkoden implementert og i figur 11 er et fordelingseksempel illustrert.

$$SNR_i = \frac{K_{SNR} \cdot RCS_i}{{R_i}^4 \cdot L_i} \tag{4-1}$$

$$L_i = 10^{1.2\left[\left(\frac{\Delta Az_i}{AzBw}\right)^2 + \left(\frac{\Delta El_i}{ElBw}\right)^2\right]} \tag{4-2}$$

Symbol	Beskrivelse
K_{SNR}	Konstant for riktig tilnærming av SNR.
RCS_i	Målets radarareal (Radar Cross Section) i kvadratmeter.
R_i	Avstand til målet i meter.
ΔAz_i	Asimut målavvik fra senter i radar strålen i radianer.
$AzBw$	Radarens strålebredde i asimut i radianer.
ΔEl_i	Elevasjons målavvik fra senter i radar strålen i radianer.
$ElBw$	Radarens strålebredde i elevasjon i radianer.

Tabell 2: Variable og konstanter i beregning av SNR



Figur 11: SNR som funksjon av feil i asimut og elevasjon

4.2.1 Monopulsbasert metode

Det finnes forskjellig typer av ACM hvor ulike antall målepunkt benyttes. Den mest relevante metoden for denne oppgaven er naturligvis den som baserer seg på amplitudeforholdet i spenningen mellom fire forskjellige målepunkter. Amplituden av spenningen må her skiftes ut med amplituden av SNR.

I ligning 4-3 til 4-5 vises det først hvordan sum og differanse beregnes av SNR i de forskjellige målepunktene. Amplitudeforholdet er da gitt av differansen i asimut og elevasjon delt på sum slik som vist i ligning 4-6 og 4-7. Dette forholdet er direkte proporsjonalt med vinkelfeilen til målet. Dersom et mål befinner seg innen området som er dekket av de fire sektorene vil dermed målposisjonen kunne estimeres med rimelig stor nøyaktighet ved å multiplisere sist oppdaterte målposisjon med dette forholdet slik som vist i ligning 4-8 og 4-9.

$$\Sigma = SNR_1 + SNR_2 + SNR_3 + SNR_4 \tag{4-3}$$

$$\Delta_{Az} = (SNR_2 + SNR_4) - (SNR_1 + SNR_3) \tag{4-4}$$

$$\Delta_{El} = (SNR_1 + SNR_2) - (SNR_3 + SNR_4) \tag{4-5}$$

$$\delta_{Az} = \frac{\Delta_{Az}}{\Sigma} \tag{4-6}$$

$$\delta_{El} = \frac{\Delta_{El}}{\Sigma} \tag{4-7}$$

$$Az_{\delta} = \delta_{Az} \cdot {}^{\Delta\theta}/_{2} \tag{4-8}$$

$$El_{\delta} = \delta_{El} \cdot {}^{\Delta\theta}/_{2} \tag{4-9}$$

Siden den ene aksen i det virkelige radarsystemet er forskjøvet med en vinkel γ vil den estimerte posisjonsfeilen ha et lite avvik. Dette kan korrigeres ved å dekomponere målepunktene som vektorer i asimut og elevasjon for så å foreta summering og differanseberegninger basert på disse verdiene. Ligning 4-10 til 4-13 settes da inn i ligning 4-3 til 4-5.

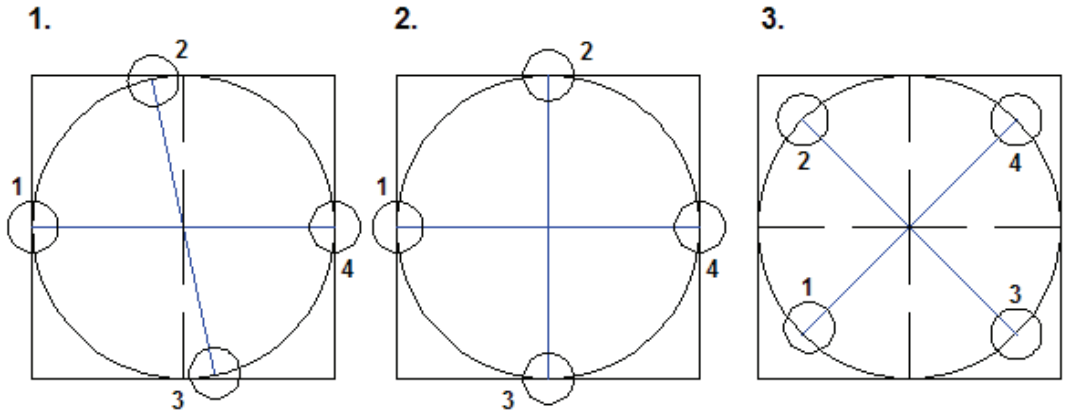
$$SNR_1 = SNR_{målt,1} + \left(SNR_{målt,2} - SNR_{målt,3}\right) \cdot \cos(\gamma) \tag{4-10}$$

$$SNR_2 = SNR_{målt,2} \cdot \sin(\gamma) \tag{4-11}$$

$$SNR_3 = SNR_{målt,3} \cdot \sin(\gamma) \tag{4-12}$$

$$SNR_4 = SNR_{målt,4} + \left(SNR_{målt,3} - SNR_{målt,2}\right) \cdot \cos(\gamma) \tag{4-13}$$

Siden aksene for vinkel i asimut og elevasjon i ACM normalt representerer et skille mellom de fire målepunktene og dette ikke samsvarer med plasseringen av strålesektorene i vårt radarsystem er det nødvendig å rotere systemet 45 grader. Dette gjøres ved bruk av en rotasjonsmatrise slik som vist i ligning 4-14 og 4-15. Figur 12 viser de tre rotasjonstrinnene som må til for å tilpasse systemet i denne oppgaven til metoden i [7, Barton, D. K.].



Figur 12: Rotasjonstrinn for tilpasning av monopulsbasert metode

$$R_{\frac{\pi}{4}} = \begin{bmatrix} \cos\frac{\pi}{4} & -\sin\frac{\pi}{4} \\ \sin\frac{\pi}{4} & \cos\frac{\pi}{4} \end{bmatrix} \tag{4-14}$$

$$\begin{bmatrix} Az_\varepsilon \\ El_\varepsilon \end{bmatrix} = R_{\frac{\pi}{4}} \begin{bmatrix} Az_\delta \\ El_\delta \end{bmatrix} \tag{4-15}$$

Et estimat av målets posisjonsavvik beregnes fra den detekterte målposisjonen basert på SNR. Feilen kan så legges til den sist oppdaterte målposisjonen for å finne nye oppdaterte observasjonsdata slik som vist i ligning 4-16 og 4-17.

$$Az_{k+1} = Az_k + K_{Az} \cdot Az_\varepsilon \tag{4-16}$$

$$El_{k+1} = El_k + K_{El} \cdot El_\varepsilon \tag{4-17}$$

Vektleggingen av den estimerte feilen, Az og El, multipliseres med en forsterkning for å oppnå best mulig tilnærming av observasjonsdata. Forsterkningen er i denne oppgaven tilnærmet slik som i ligning 4-18 og 4-19.

$$K_{Az} = 1.1 - 10^{\delta_{Az}-1} \tag{4-18}$$

$$K_{El} = 1.5 - 10^{\delta_{El}-1} \tag{4-19}$$

I figur 13 til 16 er det gitt eksempler på hvordan amplitudebasert metode fungerer med ulike vinkelfeil. Det røde merket er virkelig feil og det turkise merket er estimert feil. De blå feltene er radarstråleposisjon.

4.2.2 Vinkelbasert metode

I likhet med ACM tilpasses SL slik at amplituden til SNR kan brukes for å oppdatere observasjonsdata i asimut og elevasjonsvinkel. Den største forskjellen er at SL metoden benytter logaritmiske SNR data slik som vist i ligning 4-20.

$$SNR_{dB,i} = 10 \cdot \log_{10} SNR_i \quad for \quad i = 1, \dots, 4 \tag{4-20}$$

For å finne vinkeloppdatering i asimut multipliseres differansen mellom SNR_{dB} i stråle 1 og 4 med forsterkningen, K_{Az}. Det samme gjøres også for elevasjons vinkeloppdatering men her multipliseres forsterkningen, K_{El}, med differansen mellom SNR_{dB} i stråle 2 og 3. Differansen mellom asimut vinkel i forrige og nåværende oppdatering legges også til slik at det kompenseres for vinkelen γ. Dette er vist i ligning 4-21 og 4-22.

$$Az_\varepsilon = K_{Az} \cdot \left[SNR_{db,4} - SNR_{db,1}\right] \tag{4-21}$$

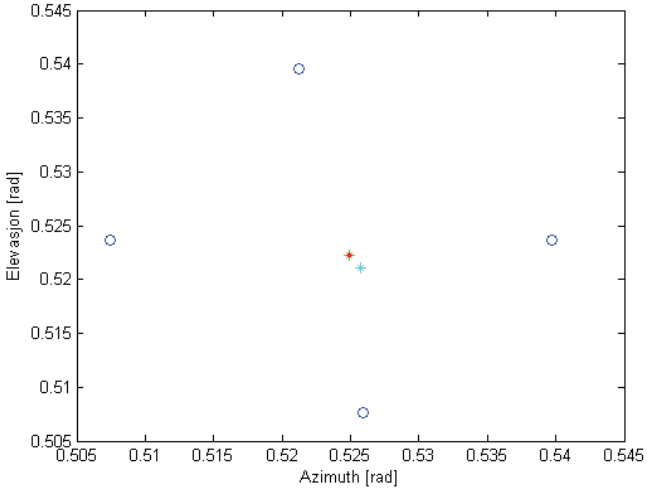
$$El_\varepsilon = \frac{K_{El} \cdot \left[SNR_{db,2} - SNR_{db,3}\right] + \left[Az_{k+1} - Az_k\right] \cdot \cos\gamma}{\sin\gamma} \tag{4-22}$$

Vinkelfeilen legges så til forrige vinkelestimat som i ligning 4-23 og 4-24, slik at oppdaterte observasjonsdata genereres.

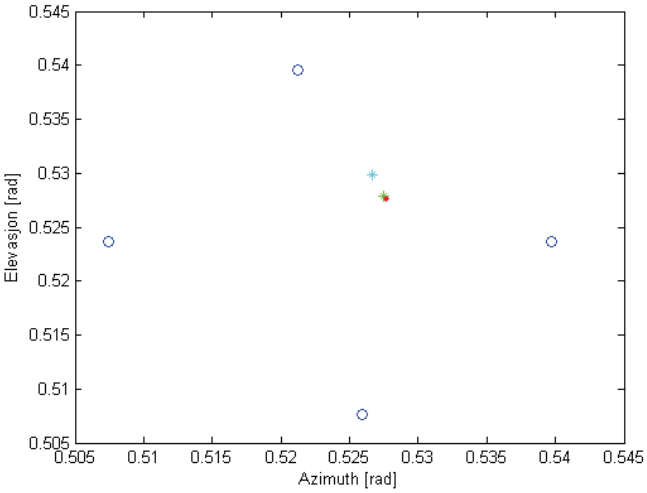
$$Az_{k+1} = Az_k + K_{Az} \cdot Az_\varepsilon \tag{4-23}$$

$$El_{k+1} = El_k + K_{El} \cdot El_\varepsilon \tag{4-24}$$

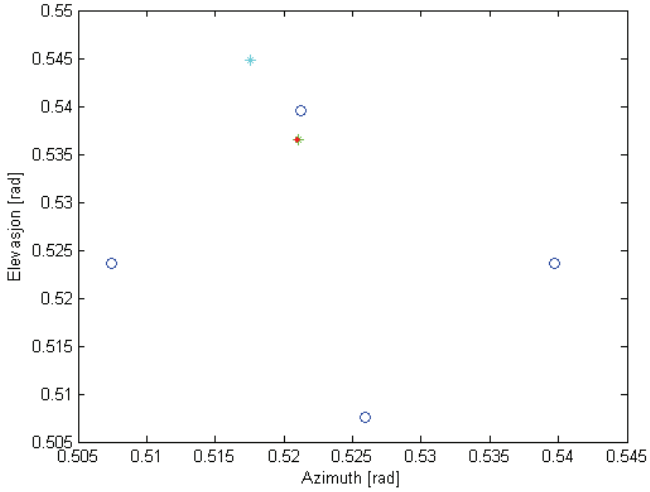
I figur 13 til 16 er det også gitt eksempler på hvordan denne metoden fungerer med ulike vinkelfeil. Her er estimert vinkelposisjon markert med et grønt punkt.



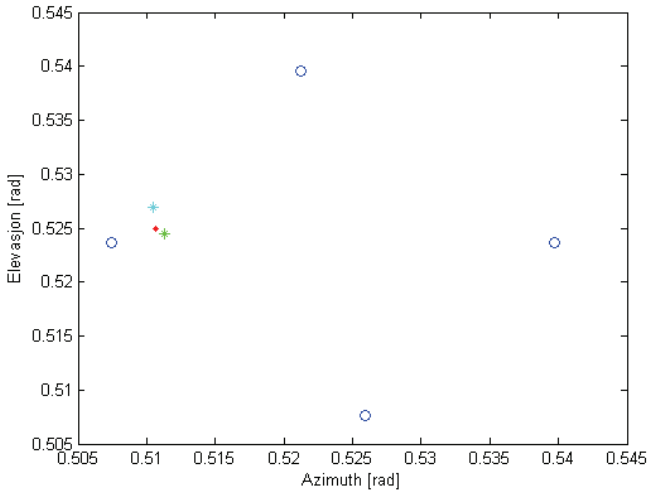
Figur 13: Vinkelfeil Az = Δθ/24, El = -Δθ/24



Figur 14: Vinkelfeil Az = Δθ/8, El = Δθ/8



Figur 15: Vinkelfeil Az = -Δθ/24, El = Δθ/2.5



Figur 16: Vinkelfeil Az = -Δθ/2.5, El = Δθ/24

Figur 13 til 16 viser at det følger større usikkerhet med ACM enn SL. I kapittel 5.1 skal begge metodene for oppdatering av observasjonsdata simuleres slik at de kan tas nærmere i betraktning.

4.3 Observasjonsstøy

Termisk støy for radarer omfatter støyen som blir generert av den termiske endringen i ledningsevne i den ohmske delen av radarens mottakerfase. En av utfordringene i denne oppgaven er å beregne støyen som funksjon av SNR i målepunktene i figur 7. Støyen kan da settes inn som målestøyens kovariansmatrise i estimatorer for å utbedre estimatene. Det vil her bli lagt frem tre måter å foreta oppdatering av observasjonsstøy i asimut og elevasjonsvinkel. Implementeringen av disse algoritmene kan finnes i appendiks B.5.

Radarstrålebredde ved -3 dB

Strålebredden for asimut, $\theta_{A,3}$, og elevasjon, $\theta_{El,3}$, inngår i samtlige av algoritmene som presenteres i dette kapitlet. I figur 17 er et eksempel illustrert. En måte å beregne dem på er ved ligning 4-25 til 4-26:

$$\Sigma_{Az}(\theta) = SNR_1(\theta) + SNR_4(\theta) \tag{4-25}$$

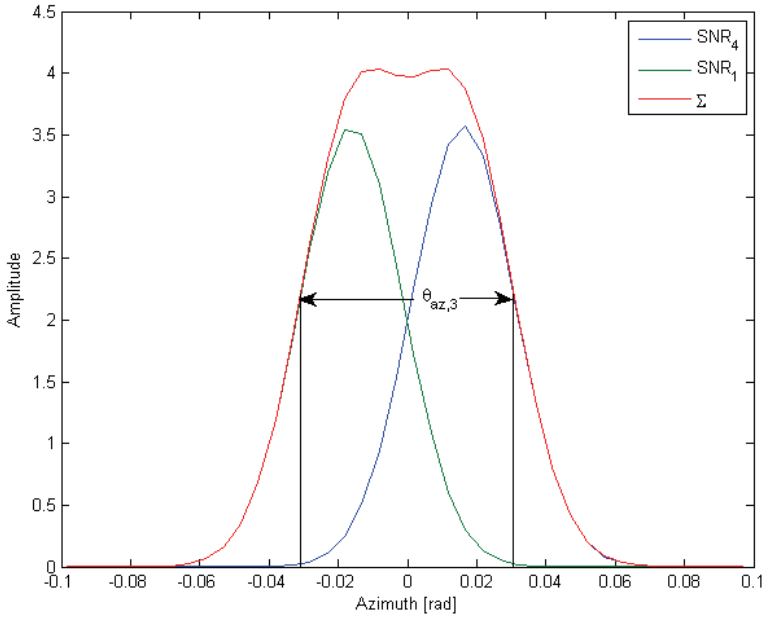
$$\Sigma_{El}(\theta) = SNR_2(\theta) + SNR_3(\theta) \tag{4-26}$$

Først må denne funksjonen deriveres med hensyn på vinkelen i asimut eller elevasjon og deretter settes lik null for å finne maksimumsverdien $\Sigma_{\max}$. Fra denne verdien kan da $\Sigma_{-3\mathrm{dB}}$ beregnes. θ_3 hentes så ut ved å sette inn $\Sigma_{-3\mathrm{dB}}$ for Σ.

θ_3 kan også beregnes fra strålebredden til de enkle SNR funksjonene og vinkelen mellom dem slik som vist i ligning 4-27 og 4-28 og figur 17. Dette er en enklere og tilstrekkelig presis tilnærming og vil dermed brukes videre.

$$\theta_{A,3} = AzBw + \Delta\theta \tag{4-27}$$

$$\theta_{El,3} = ElBw + \Delta\theta \tag{4-28}$$



Figur 17: Radarstrålebredde ved SL

SNR for asimut og elevasjon

SNR_{Az} og SNR_{El} inngår også i samtlige av algoritmene og kan tilnærmes ved gjennomsnittet av SNR_1 og SNR_4 for asimut, og SNR_2 og SNR_3 for elevasjon slik som i ligning 4-29 og 4-30.

$$SNR_{Az} = \frac{SNR_1 + SNR_4}{2} \tag{4-29}$$

$$SNR_{El} = \frac{SNR_2 + SNR_3}{2} \tag{4-30}$$

4.3.1 Bartons metode

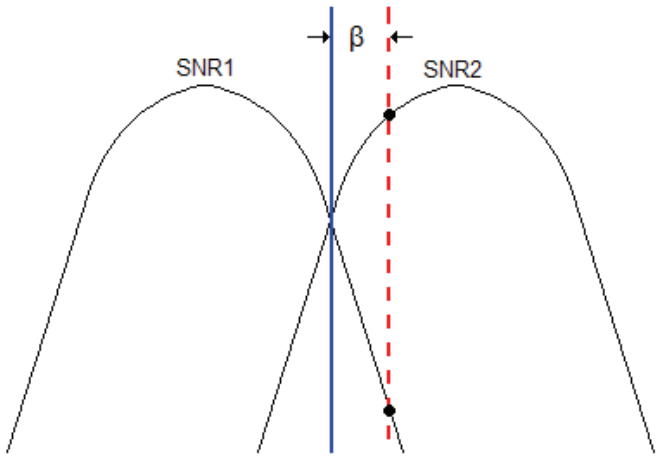
David K. Barton er en forsker som har skrevet en rekke bøker om radarteori hvor blant annet termisk støy inngår. I appendiks A.1 gjennomgås en metode fra [7] som er basert på monopulsbasert målfølging. Ligning 4-31 og 4-32 viser den endelige formen.

$$\sigma_{Az} = \frac{\theta_{Az,3}}{1.6\sqrt{2 \cdot SNR_{Az}}} \tag{4-31}$$

$$\sigma_{El} = \frac{\theta_{El,3}}{1.6\sqrt{2 \cdot SNR_{El}}} \tag{4-32}$$

4.3.2 NOAH's metode

Hughes Aircraft Company har utviklet flere algoritmer for feilestimering i målfølging for forskjellige radarer. I en av deres artikler [8, Ground System Group] legges det blant annet frem en modell for spenningsstyrt generering av støy ved SL, der β er tilnærmet vinkelavvik og $F = \theta_3/\Delta\theta$ er "stack factor". Forholdet mellom spenningen i målepunktene tilnærmes ved SNR. Figur 18 er hentet fra [8] og illustrerer β i SL.



Figur 18: Vinkelavvik ved SL

I appendiks A.2 er ligningene for asimut, σ_{Az}, og elevasjon, σ_{El}, utledet slik at de kan settes opp som i 4-33 til 4-36. Metoden justeres til slutt med en konstant forsterkning.

$$\beta_{Az} = \frac{F \cdot AzBw}{5.55} ln\left(\frac{V_1}{V_4}\right) \approx \frac{F \cdot AzBw}{5.55} ln\left(\frac{SNR_1}{SNR_4}\right) \tag{4-33}$$

$$\sigma_{Az} = E\left[\dot{\beta}_{Az}\right] = K_{\sigma_{Az}} \cdot \frac{F \cdot AzBw}{5.55\sqrt{2 \cdot SNR_{Az}}} \left[e^{5.55\left(\frac{1}{2F} - \frac{\beta_{Az}}{AzBw}\right)^2} - e^{5.55\left(\frac{1}{2F} + \frac{\beta_{Az}}{AzBw}\right)^2}\right]^{\frac{1}{2}} \tag{4-34}$$

$$\beta_{El} = \frac{F \cdot ElBw}{5.55} ln\left(\frac{SNR_2}{SNR_3}\right) \tag{4-35}$$

$$\sigma_{El} = E\left[\dot{\beta}_{El}\right] = K_{\sigma_{El}} \cdot \frac{F \cdot ElBw}{5.55\sqrt{2 \cdot SNR_{El}}} \left[e^{5.55\left(\frac{1}{2F} - \frac{\beta_{El}}{ElBw}\right)^2} - e^{5.55\left(\frac{1}{2F} + \frac{\beta_{El}}{ElBw}\right)^2}\right]^{\frac{1}{2}} \tag{4-36}$$

4.3.3 Vinkelstøybasert metode

Metodene for oppdatering av observasjonsstøy som er beskrevet hittil baserer seg på generelle utledninger som er funnet i ulike deler av radarlitteraturen. Disse kan i verste fall bli for lite konkrete for å generere korrekte støydata. Men ved å beregne $E[\dot{A}z]$ og $E[\dot{E}l]$ fra ligning 4-23 og 4-24 kan standardavviket, σ_{Az} og σ_{El}, bestemmes spesielt for vinkelbasert metode beskrevet i denne oppgaven. Med opphav i vinkelbasert metode kalles denne formen for generering av observasjonsstøy for vinkelstøybasert metode. Utledningene er vist i appendiks A.3 og resultatet er her gitt i ligning 4-37 og 4-38.

$$\sigma_{Az} = \left| \frac{K_{Az}}{\sqrt{2}} \cdot \sqrt{\frac{SNR_1 - SNR_4}{SNR_4 \cdot SNR_1}} \right| \tag{4-37}$$

$$\sigma_{El} = \left| \frac{K_{El}}{\sqrt{2}} \cdot \sqrt{\frac{SNR_3 - SNR_2}{SNR_3 \cdot SNR_2 \cdot \sin\gamma}} \right| \tag{4-38}$$

4.4 Estimator

Målfølging består hovedsakelig av å finne hastigheten og retningen til mål slik at de lar seg overvåke. En viktig del av dette er estimering av fremtidige måldata og i dette kapitlet presenteres to slike metoder. EKF og IMM filter representerer to ulike former for estimering i manøvrerende miljø. Implementeringen av filtrenes programmer er fremlagt i appendiks B.8 og B.9.

4.4.1 Utvidet Kalman-filter

Kalman-filteret (KF) [9, Brown, R. G. & Hwang P. Y.] er et rekursivt filter som estimerer tilstander i lineære, stokastiske og dynamiske systemer. Signaler påvirket av hvit støy estimeres ved minimalisering av middelkvadratfeil i målingen. KF har noen klare fordeler som gjør det spesielt velegnet for bruk i målfølging.

1. Forsterkningskoeffisientene blir beregnet dynamisk slik at filteret kan brukes i forskjellige målmanøvrerende miljøer.
2. Filteret adapterer seg til varierende deteksjonshistorikk.

Kalman-filter modell

I denne oppgaven følges utledningen av KF fra [9]. Den dynamiske *prosesslikningen* er gitt i ligning 4-39 og *måleligningen* er gitt i ligning 4-40.

$$x_{k+1} = \Phi_k x_k + \Delta_k u_k + \Gamma_k w_k \tag{4-39}$$

$$z_k = H_k(x_k) + G_k(x_k) v_k \tag{4-40}$$

Her er, w_k, *prosesstøy* og, v_k, *målestøy* gjensidig uavhengige sekvenser hvit gaussisk støy med gjennomsnitt lik null. Disse benyttes for å danne *kovariansmatrisene* Q og R slik som vist i ligning 4-41 og 4-42.

$$E[w_k {w_i}^T] = \begin{cases} Q_k, & i = k \\ 0, & i \neq k \end{cases} \tag{4-41}$$

$$E[v_k {v_i}^T] = \begin{cases} R_k, & i = k \\ 0, & i \neq k \end{cases} \tag{4-42}$$

Tilstandsvektoren er gitt ved ligning 4-43.

$$x_k = [x_k \quad y_k \quad z_k \quad v_{x,k} \quad v_{y,k} \quad v_{z,k}]^T \tag{4-43}$$

Systemmatrisen er gitt ved ligning 4-44.

$$\Phi_k = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4-44}$$

Prosesstøyens koeffisientmatrise er gitt ved ligning 4-45.

$$\Gamma_k = [0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1]^T \tag{4-45}$$

Pådragskoeffisientmatrisen er gitt ved ligning 4-46.

$$\Delta_k = [0 \quad 0 \quad 0 \quad \Delta t \quad \Delta t \quad \Delta t]^T \tag{4-46}$$

Kalman-filterligningene

For å finne kovariansmatrisen ved neste tidsskritt beregnes først estimeringsfeilen ved ligning 4-51. *Kovariansmatrisen* finnes så ved å sette ligning 4-47 i 4-48.

$$e_{k+1}^- = x_{k+1} - x_{k+1}^- \tag{4-47}$$

$$P_{k+1}^- = E\left[e_{k+1}^- {e_{k+1}^-}^T\right] = \Phi_k P_k \Phi_k + Q_k \tag{4-48}$$

Kalman-forsterkningen kan videre beregnes ved ligning 4-49.

$$K_k = P_k^- {H_k}^T \left(H_k P_k^- {H_k}^T + R_k\right)^{-1} \tag{4-49}$$

Forsterkningen brukes for å oppdatere tilstandsestimatet slik som i ligning 4-50.

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H_k \hat{x}_k^-) \tag{4-50}$$

Oppdatert kovariansmatrise er gitt ved ligning 4-51. Dette kalles *Joseph form* og er en alternativ kovariansberegning som er mindre sensitiv for avrundingsfeil og holder for alle forsterkninger K_k. Forøvrig kan den ikke gi negative egenverdier.

$$P_k = (I - K_k H_k) P_k^- (I - K_k H_k)^T + K_k R_k {K_k}^T \tag{4-51}$$

Utvidet Kalman-filter

KF baserer seg i utgangspunktet på lineære modeller og dette byr på problemer da målematrisen i systemmodellen er ulineær. Grunnen til ulinearitetet er at målematrisen utgjør koordinatkonverteringen fra kartesiske til sfæriske koordinater slik som vist i ligning 4-52 til 4-54.

$$R_k = \sqrt{{x_k}^2 + {y_k}^2 + {z_k}^2} \tag{4-52}$$

$$Az_k = \tan^{-1}(\frac{y_k}{x_k}) \tag{4-53}$$

$$El_k = \tan^{-1}(\frac{-z_k}{\sqrt{{x_k}^2 + {y_k}^2}}) \tag{4-54}$$

Det er dermed nødvendig å innføre et tiltak for å løse dette problemet. Et alternativ er *Utvidet Kalman-filter, "Extended Kalman-filter (EKF)"*.

For å forenkle uttrykket benyttes notasjonen gitt i ligning 4-55 og 4-56.

$$RSR_k = \sqrt{{x_k}^2 + {y_k}^2 + {z_k}^2} \tag{4-55}$$

$$RGR_k = \sqrt{{x_k}^2 + {y_k}^2} \tag{4-56}$$

Målematrisen lineariseres rundt et arbeidspunkt slik som vist i ligning 4-57.

$$\begin{aligned} H_k &= \begin{bmatrix} \frac{\partial R_k}{\partial x_k} & \frac{\partial R_k}{\partial y_k} & \frac{\partial R_k}{\partial z_k} & \frac{\partial R_k}{\partial v_{x,k}} & \frac{\partial R_k}{\partial v_{y,k}} & \frac{\partial R_k}{\partial v_{z,k}} \\ \frac{\partial Az_k}{\partial x_k} & \frac{\partial Az_k}{\partial y_k} & \frac{\partial Az_k}{\partial z_k} & \frac{\partial Az_k}{\partial v_{x,k}} & \frac{\partial Az_k}{\partial v_{y,k}} & \frac{\partial Az_k}{\partial v_{z,k}} \\ \frac{\partial El_k}{\partial x_k} & \frac{\partial El_k}{\partial y_k} & \frac{\partial El_k}{\partial z_k} & \frac{\partial El_k}{\partial v_{x,k}} & \frac{\partial El_k}{\partial v_{y,k}} & \frac{\partial El_k}{\partial v_{z,k}} \end{bmatrix}_{x=x^*} \\ &= \begin{bmatrix} \frac{x_k}{RSR_k} & \frac{y_k}{RSR_k} & \frac{z_k}{RSR_k} & 0 & 0 & 0 \\ -\frac{y_k}{{RGR_k}^2} & \frac{x_k}{{RGR_k}^2} & 0 & 0 & 0 & 0 \\ \frac{z_k x_k}{{RSR_k}^2 RGR_k} & \frac{z_k y_k}{{RSR_k}^2 RGR_k} & -\frac{RGR_k}{{RSR_k}^2} & 0 & 0 & 0 \end{bmatrix}_{x=x^*} \end{aligned} \tag{4-57}$$

Siden målingen er gitt i polare koordinater blir målestøymatrisen som i ligning 4-58. For å redusere matrisestørrelsene innføres heretter notasjonen av Matlab kommandoen, *A = diag(B)*, hvor B er en vektor med diagonalkomponentene til matrisen A.

$$G_k = \begin{bmatrix} \frac{\partial x_k}{\partial R_k} & \frac{\partial x_k}{\partial Az_k} & \frac{\partial x_k}{\partial El_k} \\ \frac{\partial y_k}{\partial R_k} & \frac{\partial y_k}{\partial Az_k} & \frac{\partial y_k}{\partial El_k} \\ \frac{\partial z_k}{\partial R_k} & \frac{\partial z_k}{\partial Az_k} & \frac{\partial z_k}{\partial El_k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = diag([1 \quad 1 \quad 1]) \tag{4-58}$$

I et EKF lineariseres det rundt siste oppdaterte tilstandsestimat. På den måten genereres det til en hver tid en så god tilnærming som mulig basert på forrige tidsskritt.

Initialisering

Kalman-filteret må initialiseres med et tilstandsestimat x_0 og en kovariansmatrise P_0 for å unngå store avvik i starten av simuleringene.

I denne oppgaven er det valgt å initialisere tilstandsvektoren slik som i [10, Bar-Shalom, Y., Li, X. R., & Kirubarajan, T.]. Posisjonen i x_0, y_0 og z_0 settes lik første posisjon i sann målvektor. Videre beregnes gjennomsnittet av de to første posisjonsmålingene som deles på Δt for å finne hastighetsvektorene $v_{x,0}$, $v_{y,0}$ og $v_{z,0}$. Initial tilstandsvektor blir da slik som i ligning 4-59.

$$x_0 = \begin{bmatrix} x_0 & y_0 & z_0 & \frac{x_1 - x_0}{\Delta t} & \frac{y_1 - y_0}{\Delta t} & \frac{z_1 - z_0}{\Delta t} \end{bmatrix}^T \tag{4-59}$$

Kovariansmatrisen initialiseres ved først å konvertere standardavviket i sfæriske koordinater til kartesiske koordinater slik som vist i ligning 4-60 til 4-62.

$$\sigma_x = \left(\frac{\partial x}{\partial R}\right)^2 {\sigma_R}^2 + \left(\frac{\partial x}{\partial Az}\right)^2 {\sigma_{Az}}^2 + \left(\frac{\partial x}{\partial El}\right)^2 {\sigma_{El}}^2 = \sqrt{\left(\frac{x_0}{R_0}\right)^2 {\sigma_R}^2 + (y_0\sigma_{Az})^2 + \frac{(y_0 z_0)^2}{(RGR)^2}{\sigma_{El}}^2} \tag{4-60}$$

$$\sigma_y = \left(\frac{\partial y}{\partial R}\right)^2 {\sigma_R}^2 + \left(\frac{\partial y}{\partial Az}\right)^2 {\sigma_{Az}}^2 + \left(\frac{\partial y}{\partial El}\right)^2 {\sigma_{El}}^2 = \sqrt{\left(\frac{y_0}{R_0}\right)^2 {\sigma_R}^2 + (x_0\sigma_{Az})^2 + \frac{(y_0 z_0)^2}{(RGR)^2}{\sigma_{El}}^2} \tag{4-61}$$

$$\sigma_z = \left(\frac{\partial z}{\partial R}\right)^2 {\sigma_R}^2 + \left(\frac{\partial z}{\partial Az}\right)^2 {\sigma_{Az}}^2 + \left(\frac{\partial z}{\partial El}\right)^2 {\sigma_{El}}^2 = \sqrt{\left(\frac{z_0}{R_0}\right)^2 {\sigma_R}^2 + (RGR)^2{\sigma_{El}}^2} \tag{4-62}$$

Standardavviket i kartesiske koordinater settes så på diagonalen av en 3x3 matrise.

$$w_{diag} = \begin{bmatrix} \sigma_x & 0 & 0 \\ 0 & \sigma_y & 0 \\ 0 & 0 & \sigma_z \end{bmatrix} \tag{4-63}$$

P_0 kan da bygges opp slik som i ligning 4-64.

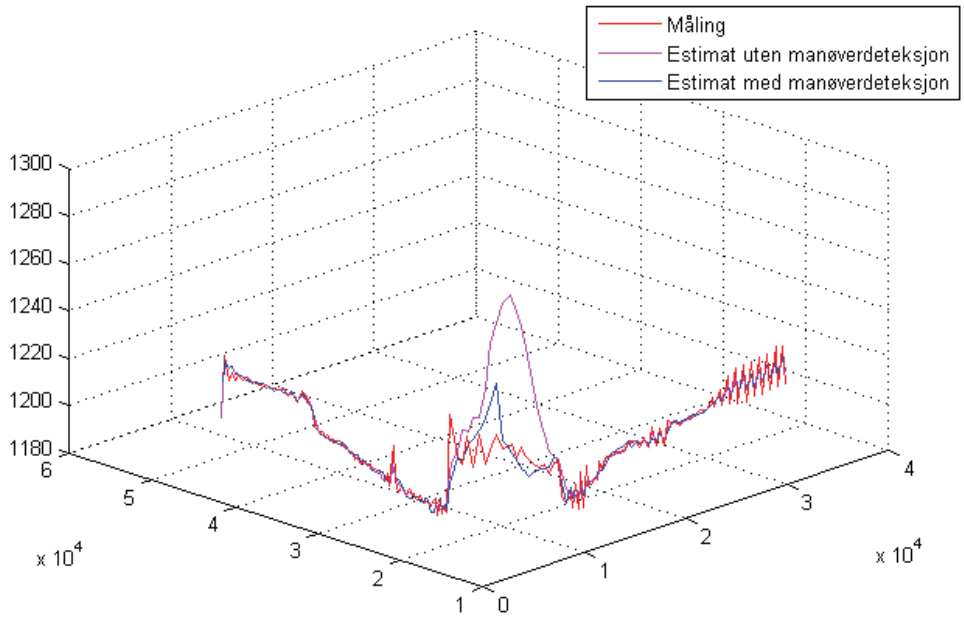
$$P_0 = \begin{bmatrix} w_{diag} & \frac{w_{diag}}{\Delta t} \\ \frac{w_{diag}}{\Delta t} & \frac{2 \cdot w_{diag}}{\Delta t} \end{bmatrix} \tag{4-64}$$

Denne formen for initialisering kalles *Topunkts differensiering* og en stor fordel ved denne fremgangsmåten er at den garanterer *konsistens* i initialiseringen. Konsistens er et mål på om en tilstand estimeres slik at den konvergerer til virkelig verdi. Implementeringen av initialiseringen er vist i appendiks B.6.

Manøverdeteksjon

Lineariseringen i ligning 4-57 gjør KF sårbart for akselerasjoner som følge av ulineariteter grunnet manøvrering og akselerasjon. Det vil da være naturlig å innføre en algoritme for å detektere og behandle slike manøvrer. I litteraturen finnes det forskjellige manøverdeteksjonsalgoritmer men mye overlates til den enkelte oppgave som skal løses. Det er av den grunn utviklet en ny algoritme slik som vist i appendiks B.8.

Dersom differansen mellom hastighetsmålingen i forrige og neste tidsskritt overstiger en gitt grenseverdi vil diagonalelementene kovariansmatrisen, Q, for prosesstøyen økes. Grunnen er at en økning i Q gir større P_k som igjen gir større Kalman-forsterkning, K_k. Forsterkningen vektlegger hensynet det skal tas til innovasjonen (måleresidualet) i tilstandsoppdateringen. Ved en manøver tillegges da innovasjonen større vekt slik at estimatoren henter seg inn i manøverpartiet. Da støykomponenter også kan medføre at grenseverdien overstiges uten at en reell manøver finner sted benyttes gjennomsnittet av residualet i de ti siste tidsskritt når det sammenlignes med grenseverdien. I figur 19 er et eksempel på et mål som estimeres med EKF med og uten manøverdeteksjon.



Figur 19: Manøverdeteksjon

Vinkeldeteksjon

En problemstilling som dukker opp ved konvertering mellom forskjellige koordinatsystemer er vinkelfeil. Dersom asimut vinkel er målt ved et positivt antall grader og estimert ved et negativt antall grader eller motsatt, vil residualet kunne gi gal vinkeloppdatering for konvertering til kartesiske koordinater. Dette vil medføre at estimatene divergerer og det er dermed nødvendig å innføre et system som foretar vinkeldeteksjon. Algoritmen for vinkeldeteksjon gitt i appendiks B.8 benytter sum i stedet for å trekke estimert fra målt vinkel dersom estimert og målt vinkel har ulikt fortegn.

Konvertering fra sfæriske til kartesiske koordinater

Utgangen til systemmodellen for KF, z_k, er gitt i sfæriske koordinater. Disse må konverteres til kartesiske koordinater før de kan illustreres i plot og lignende. Ligning 4-65 til 4-67 viser hvordan denne koordinatkonverteringen gjøres.

$$x_k = R_k \cdot \cos Az_k \cdot \cos El_k \tag{4-65}$$

$$y_k = R_k \cdot \sin Az_k \cdot \cos El_k \tag{4-66}$$

$$z_k = -R_k \cdot \sin El_k \tag{4-67}$$

4.4.2 Multiple samvirkende modeller filter

Multiple samvirkende modeller, "Interacting Multiple Model (IMM)", filter benytter et gitt antall modeller, r, som hver er tilpasset systemet i forskjellige situasjoner. Hver av modellene generer estimater basert på sin tilnærming av systemet og disse veies så opp mot hverandre i en Bayesiansk sannsynlighetsberegning. Resultatet av denne beregningen er IMM filterets oppdaterte estimat.

Den statistiske sammensetningen av modellene medfører at det ikke er nødvendig med en egen manøverdeteksjonsalgoritme slik som i EKF. For radarsignaler basert på virkelig målbane med tillagt hvit støy gir i følge [10] og [1] IMM filteret bedre estimering i manøvrerende miljø enn EKF.

Begge metodene tas allikevel med i denne oppgaven da observasjonsdata og observasjonsstøy genereres på en helt ny måte og kan ha innvirkning på hvem estimeringsalgoritme som fungerer best.

En ulempe ved IMM filteret er at flere modeller medfører større kompleksitet og dermed hardere regnebelastning.

IMM modeller

I denne oppgaven tilnærmes systemet med to modeller hvor hver er basert på KF med forskjellige egenskaper. Den første modellen, IMM1, har til hensikt å tilnærme mål med rettlinjet bevegelse uten variasjon i akselerasjon. Modellen er dermed gitt ved matrisene i ligning 4-68 til 4-71.

$$\Phi_k = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4-68}$$

$$\Gamma_k = [0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0]^T \tag{4-69}$$

$$\Delta_k = [0 \quad 0 \quad 0 \quad \Delta t \quad \Delta t \quad \Delta t \quad 0 \quad 0 \quad 0]^T \tag{4-70}$$

$$G_k = diag([1 \quad 1 \quad 1]) \tag{4-71}$$

Den andre modellen, IMM2, har til hensikt å tilnærme mål som foretar manøvre med varierende akselerasjon. Modellen er dermed gitt ved matrisene i ligning 4-72 til 4-75.

$$\Phi_k = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{\Delta t^2}{2} & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{\Delta t^2}{2} & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{\Delta t^2}{2} \\ 0 & 0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4-72}$$

$$\Gamma_k = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1]^T \tag{4-73}$$

$$\Delta_k = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \Delta t \quad \Delta t \quad \Delta t]^T \tag{4-74}$$

$$G_k = diag([1 \quad 1 \quad 1]) \tag{4-75}$$

I implementeringen av EKF i kapittel 5.3 tas det ikke hensyn til akselerasjon i modellen. Det gjør det derimot i IMM filterets modeller og modellene utvides dermed med tre tilstander for også å omfatte akselerasjon, slik som vist i ligning 4-76.

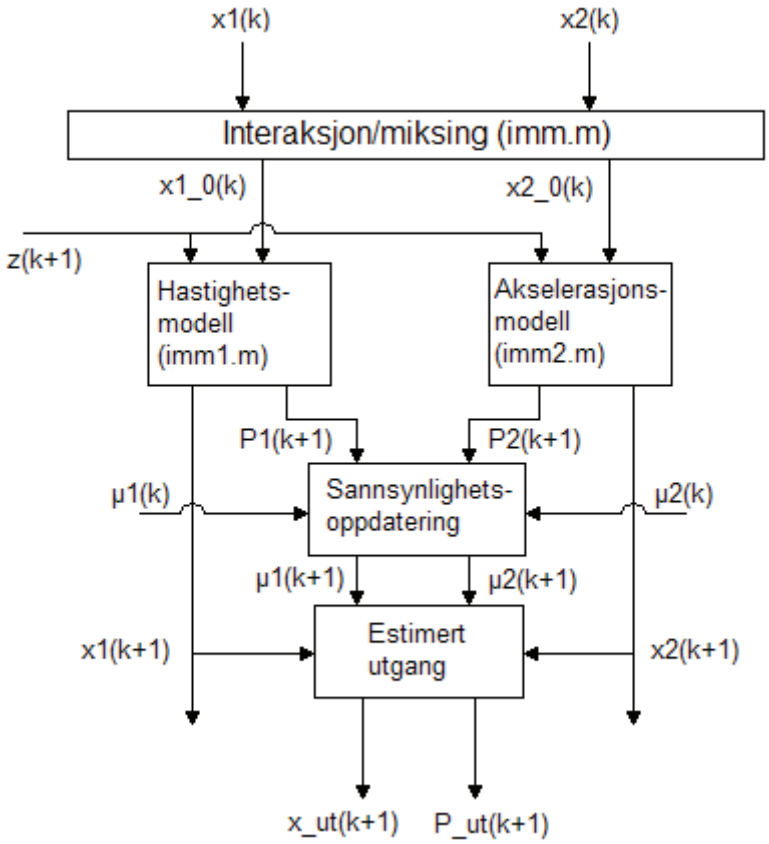
$$x_k = [x_k \quad y_k \quad z_k \quad v_{x,k} \quad v_{y,k} \quad v_{z,k} \quad a_{x,k} \quad a_{y,k} \quad a_{z,k}] \tag{4-76}$$

Målematrisen for begge modellene i ligning 4-77 er en utvidelse av ligning 4-57.

$$H_k = \begin{bmatrix} \frac{x_k}{RSR_k} & \frac{y_k}{RSR_k} & \frac{z_k}{RSR_k} & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{y_k}{{RGR_k}^2} & \frac{x_k}{{RGR_k}^2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{z_k x_k}{{RSR_k}^2 RGR_k} & \frac{z_k y_k}{{RSR_k}^2 RGR_k} & -\frac{RGR_k}{{RSR_k}^2} & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4-77}$$

IMM filterligningene

For å gi en illustrasjon av IMM filterets algoritme vises det i figur 20 en syklus av de rekursive beregningene. I tillegg beskrives algoritmen med tilhørende ligninger i trinnene nedenfor. Utledningen av filteret er mer detaljert beskrevet i [10].

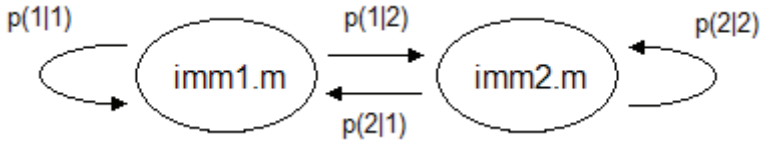


Figur 20: En syklus i IMM algoritmen

Ved bruk av Bayes formel kan det vises at modellenes korrekthet med dataene frem til tidsskritt k er gitt ved ligning 4-78. Dette kalles her *kombinasjonssannsynligheten* og vekter sammensetningssanssynligheten av de forskjellige modellene.

$$\mu_{i|j}(k-1|k-1) = \frac{1}{\bar{c}_j} p_{ij}\mu_i(k-1), \qquad i,j = 1,\dots,r \tag{4-78}$$

Matrisen p_{ij} er Markov-kjedens transisjonsmatrise. Denne bestemmer vektingen for når overgangen skal skje mellom modellene. I figur 21 er dette illustrert for modellene som brukes i denne oppgaven.



Figur 21: IMM filterets transisjon mellom 2 modeller

De normaliserte konstantene, $\overline{c}_j$, er gitt ved ligning 4-79.

$$\overline{c}_j = \sum_{i=1}^{r} p_{ij}\mu_i(k-1)\,, \qquad i,j = 1,\dots,r \tag{4-79}$$

IMM filteret genererer nye sammensatte initialverdier for tilstandsestimatet, $\hat{x}$, og kovariansmatrisen, P, ved hver syklus. Disse er gitt i ligning 4-80 og 4-81.

$$\hat{x}^{0j}(k-1|k-1) = \sum_{i=1}^{r} \hat{x}^{i}(k-1|k-1)\mu_{i|j}(k-1|k-1)\,, \qquad i,j = 1,\dots,r \tag{4-80}$$

$$\begin{aligned} P^{0j}(k-1|k-1) &= \sum_{i=1}^{r} \mu_{i|j}(k-1|k-1)\Big\{P^{i}(k-1|k-1) \\ &+ \left[\hat{x}^{i}(k-1|k-1) - \hat{x}^{0j}(k-1|k-1)\right] \\ &\cdot \left[\hat{x}^{i}(k-1|k-1) - \hat{x}^{0j}(k-1|k-1)\right]^{T}\Big\}, \qquad i,j = 1,\dots,r \end{aligned} \tag{4-81}$$

Initialverdiene for tilstandsvektoren og kovariansmatrisen utgjør da inngangen til de to EKF modellene som benyttes i denne oppgaven. Disse utvides nå til også å omfatte ”Likelihood”-funksjonen gitt i ligning 4-82. Dette er sannsynlighetstetthetsfunksjonen av den Gaussiske fordelingen til hver av modellene. I [11, Newman, P. & Leonard, J.] fremgår det denne funksjonen kan skrives som i ligning 4-83, der residualet, res, i ligning 4-84 er det samme som innovasjonen i KF.

$$\begin{aligned} \Lambda_{\mathrm{j}}(\mathrm{k}) = \mathcal{N}\Big[&\mathrm{x}(\mathrm{k}); \hat{\mathrm{z}}^{\mathrm{j}}\big[\mathrm{k}|\mathrm{k}-1; \hat{\mathrm{x}}^{0\mathrm{j}}(\mathrm{k}-1|\mathrm{k} \\ &-1)\big], \mathrm{S}^{\mathrm{j}}\big[\mathrm{k}; \mathrm{P}^{0\mathrm{j}}(\mathrm{k}-1|\mathrm{k}-1)\big]\Big] \qquad j = 1,\dots,r \end{aligned} \tag{4-82}$$

$$\Lambda_{\mathrm{j}}(\mathrm{k}) = \frac{1}{\sqrt{|2\pi \cdot \mathrm{S}^{\mathrm{j}}(\mathrm{k})|}} \mathrm{e}^{-0.5\left(\mathrm{res}^{\mathrm{j}}(\mathrm{k})^{\mathrm{T}} \mathrm{S}^{\mathrm{j}}(\mathrm{k}) \mathrm{res}^{\mathrm{j}}(\mathrm{k})\right)} \qquad j = 1, \dots, r \tag{4-83}$$

$$\mathrm{res}^{\mathrm{j}}(\mathrm{k}) = z^{j}(k) - H_j[\hat{x}^{j}(k-1)] \tag{4-84}$$

Den enkelte modells sannsynlighetsoppdatering gjøres som i ligning 4-85 der $\overline{c}_j$ er den samme som i ligning 4-79 og c er den normalisert konstanten gitt i ligning 4-86.

$$\mu_j(k) = \frac{1}{c} \Lambda_{\mathrm{j}}(\mathrm{k}) \overline{c}_j, \qquad j = 1, \dots, r \tag{4-85}$$

$$c = \sum_{j=1}^{r} \Lambda_{\mathrm{j}}(\mathrm{k}) \overline{c}_j\,, \qquad j = 1, \dots, r \tag{4-86}$$

Siste del av algoritmens syklus er oppdateringen av kombinerte tilstandsestimater og kovarians i ligning 4-87 og 4-88. Et viktig poeng er her at disse ikke inngår i de rekursive trinnene over, men kun utgjør utgangene til filteret.

$$\hat{x}(k|k) = \sum_{j=1}^{r} \hat{x}^{j}(k|k) \mu_j(k) \tag{4-87}$$

$$P(k|k) = \sum_{j=1}^{r} \mu_j(k) \left\{ P^{j}(k|k) + \left[\hat{x}^{j}(k|k) - \hat{x}(k|k)\right] \cdot \left[\hat{x}^{j}(k|k) - \hat{x}(k|k)\right]^{T} \right\} \tag{4-88}$$

Initialisering

I likhet med EKF må IMM filteret initialiseres med tilstander og kovariansmatrise. IMM filterets modeller er utvidet med tre tilstander for akselerasjon og disse settes lik 0 i initialtilstandenes vektor. Kovariansmatrisen initialiseres ved å sette de utvidede akselerasjonskomponentene lik 0. Initialverdier for tilstandsvektor og kovariansmatrise er dermed gitt i ligning 4-89 og 4-90, og implementeringen er gitt i appendiks B.7. Disse ligningene må ikke forveksles med ligning 4-80 og 4-81 som inngår i den rekursive oppdateringen av IMM filteralgoritmen.

$$x_0 = \begin{bmatrix} x_0 & y_0 & z_0 & \frac{x_1 - x_0}{\Delta t} & \frac{y_1 - y_0}{\Delta t} & \frac{z_1 - z_0}{\Delta t} & 0 & 0 & 0 \end{bmatrix}^T \tag{4-89}$$

$$P_0 = \begin{bmatrix} w_{diag} & \frac{w_{diag}}{\Delta t} & 0 \\ \frac{w_{diag}}{\Delta t} & \frac{2 \cdot w_{diag}}{\Delta t} & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4-90}$$

4.5 Scenario for mål

For å kunne vurdere egenskapene til de forskjellige metodene for målfølging er det nødvendig å bruke forskjellige målscenarioer som representerer ulike miljø. Det vil i denne oppgaven benyttes fire forskjellige målscenarioer slik som vist i figur 22 til 25.

Programmet for generering av målbaner er utviklet ved KDA og fungerer på følgende måte. Først legges et gitt antall koordinater inn i en initialiseringsfil for målbanen. I tillegg til å legge inn posisjon kan også hastigheten varieres fra punkt til punkt. Et program laster så inn filen og avrunder kurvene slik at banen blir realistisk med tanke på et flygende objekt. Til slutt sender den banedataene ut i en fil med både posisjons-, hastighets- og akselerasjonskoordinater.

Koordinatene kan så leses ut i Matlab med programkoden gitt i appendiks B.3. I denne oppgaven avleses kun posisjonsvektor. Hastighet og akselerasjon estimeres av KF.

En viktig del av denne oppgaven er nettopp å estimere tilstander basert på observasjonsdata. Antall tidsskritt er da avgjørende for å estimere en varians med en gitt nøyaktighet. I [10] er det presentert en metode for å bestemme hvor mange tidsskritt som er nødvendig for å oppnå tilstrekkelig god estimering, og denne vil bli fulgt her. Dersom det antas at variansen i samplingsdataene er normalfordelt rundt gjennomsnittet vil de med 95 % sikkerhet ligge innen 25 % av de oppdaterte observasjonsdataene med tidsskritt k ≥ 123. Dette kan ses i beregningene fra ligning 4-91 til 4-93.

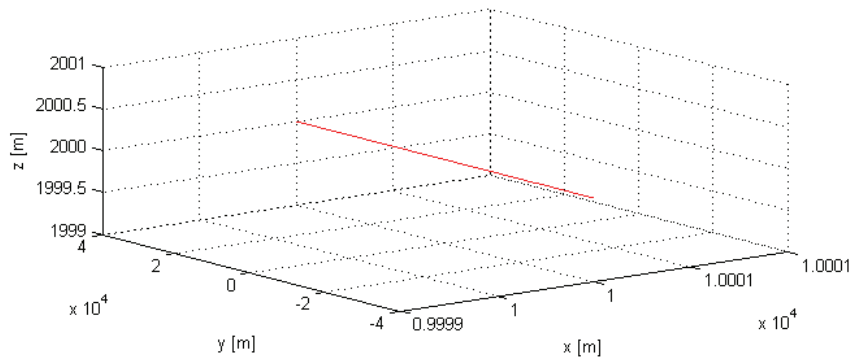
$$P\left\{|(\hat{\sigma}^{ML})^2 - \sigma^2| \leq 1.96\sigma^2\sqrt{2/k}\right\} = 0.95 \tag{4-91}$$

$$\frac{|(\hat{\sigma}^{ML})^2 - \sigma^2|}{\sigma^2} = 0.25 \tag{4-92}$$

$$1.96\sqrt{2/k} = 0.25 \Longrightarrow k \approx 123 \tag{4-93}$$

Samtlige av de fire målbanene vil dermed genereres slik at de har flere enn 123 tidsskritt.

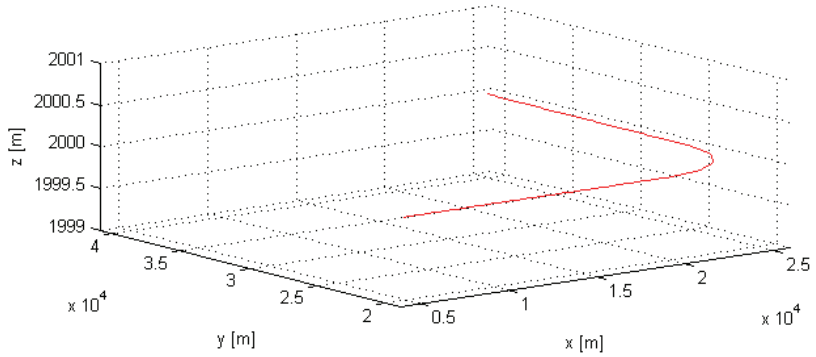
Målbane 1



Figur 22: Målbane 1

Målbane 1 i figur 22 strekker seg fra y_{start} = [10000 -40000 2000] til y_{slutt}= [10000 40000 2000]. Siden hastigheten er konstant og hvert tidsskritt er gitt ved $\Delta t = 2$ s utgjør hele banen 400 tidsskritt.

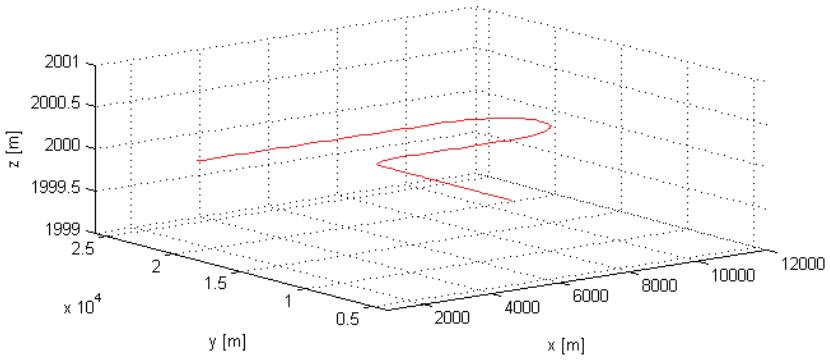
Målbane 2



Figur 23: Målbane 2

Målbane 2 i figur 23 går fra x_{start} = [5000 20000 2000] til x_{slutt}= [25000 40000 2000]. Det er implementert en treg manøver som følger en fjerdedels sirkel med radius R = 5000 m i x- og y-koordinater. Hastigheten er konstant slik at hele banen utgjør 190 tidsskritt.

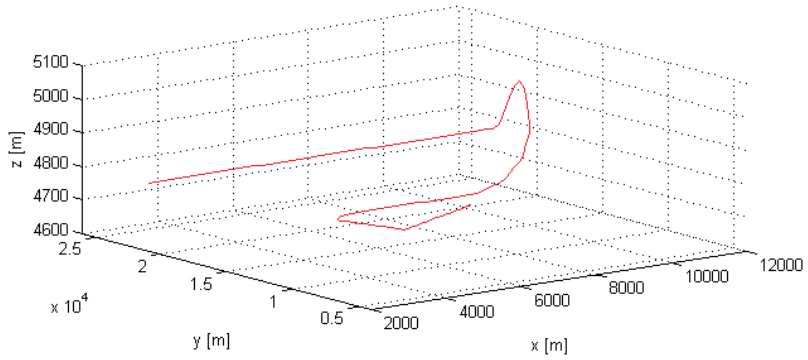
Målbane 3



Figur 24: Målbane 3

Målbane 3 i figur 24 går fra x_{start} = [5000, 5000, 2000] til x_{slutt}= [2000, 21000, 2000] og har to harde manøvre i x- og y-koordinater. I første manøver følger målet en fjerdedels sirkel med radius R = 1000 m. Neste manøver består av en halvsirkel med radius R = 2500 m. Hastigheten varierer fra 100 m/s til 150 m/s slik at hele banen utgjør 145 tidsskritt.

Målbane 4

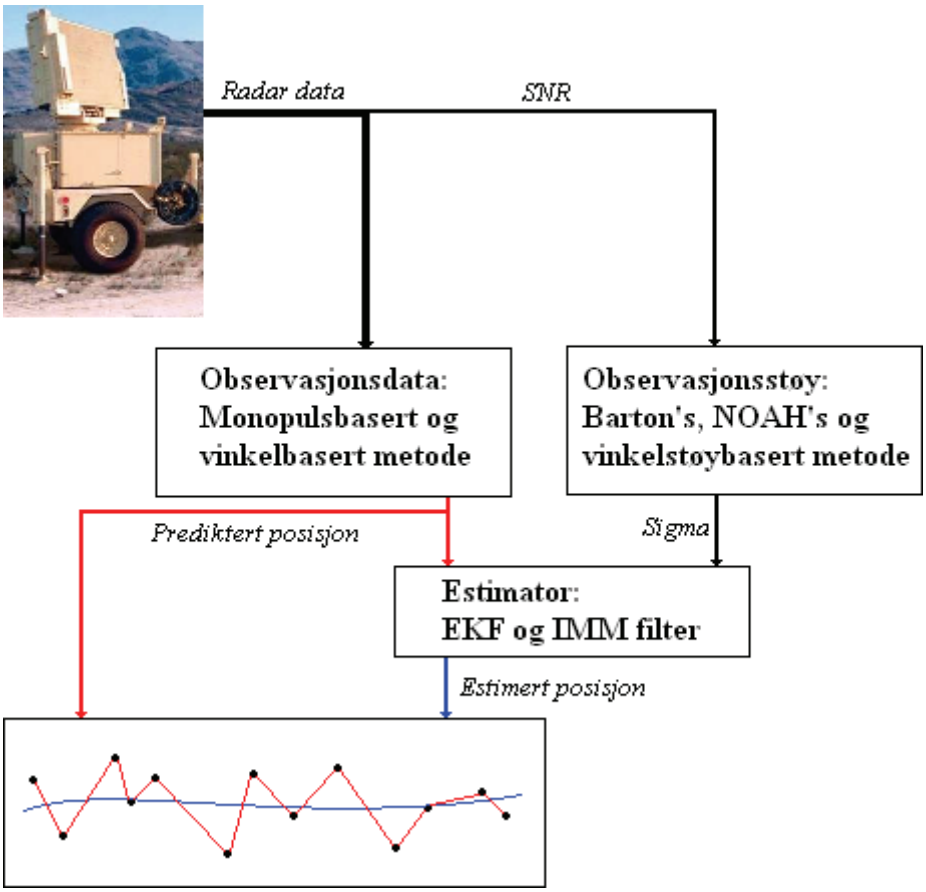


Figur 25: Målbane 4

Målbane 4 i figur 25 går fra x_{start} = [5000, 5000, 4850] til x_{slutt}= [2000, 21000, 4803.3] og har flere harde manøverpartier i x-, y- og z-koordinater. I første manøver følger målet en fjerdedels sirkel med radius R = 1000 m. Neste manøver består av en halvsirkel med radius R = 2500 m. I tillegg foretas flere harde manøver i høyde. Hastigheten varierer i tre trinn, 100 m/s, 150 m/s og 200 m/s slik at hele banen utgjør 145 tidsskritt.

4.6 Sammensetning til målfølgealgoritme

Målfølgealgoritmen skal settes sammen med de forskjellige metodene representert i foregående kapitler. I figur 26 er dette illustrert ved blokkdiagram. Her skilles det mellom prediktert og estimert posisjon. Den *predikterte* posisjonen er fra algoritmene for oppdatering av observasjonsdata og *estimert* posisjon er fra estimatorer.



Figur 26: Helhetlig målfølgealgoritme

5 Simulering og resultater

I denne delen gjennomføres simulering av de utviklede algoritmene for målfølging slik at de kan evalueres. Formålet med å simulere algoritmene er også å vise deres funksjonalitet og å provosere frem eventuelle feil.

Først gjennomgås simulering av de forskjellige algoritmene for generering av observasjonsdata og observasjonsstøy. Deretter knyttes disse opp mot estimatorer og den fullstendige målfølgealgoritmen simuleres. Ved å benytte statistiske metoder og analyse av forskjellige projeksjoner vil den sammensatte algoritmen vurderes.

5.1 Generering av observasjonsdata

Ved å simulere oppdatering av observasjonsdata med monopulsbasert og vinkelbasert metode kan deres egenskaper vurderes og veies opp mot hverandre.

For at metodene for oppdatering av observasjonsdata skal simuleres på best mulig måte må noen variable i ligningssettet bestemmes. Avstanden til målet blir generert ved å legge hvit støy med standardavvik $\sigma_R = 25$ m til sann avstand R_{true}. Asimut og elevasjonsvinkel genereres på tilsvarende måte med hvit støy med standardavvik, $\sigma_{Az} = \sigma_{El} = 0{,}001$ rad. Denne tilnærmingen er basert på hva det i virkeligheten kan oppstå av usikkerhet i avstandsmåling.

Konstantene fra ligning 4-1 og 4-2 får verdiene gitt i tabell 3.

Beskrivelse (Symbol)	**Konstanter**
Radar Cross Section (RCS)	= 10 m
Asimut radarstrålebredde (AzBw)	$= 2 \cdot \pi / 180 = 0{,}0349$ rad
Elevasjon radarstrålebredde (ElBw)	$= 1.65 \cdot \pi / 180 = 0{,}0314$ rad
Konstant for riktig tilnærming av SNR (K_{SNR})	$= 2{,}29 \cdot 10^{18}$ m^2

Tabell 3: Konstantverdier for beregning av SNR

Disse tilnærmingene gjelder simulering av begge metodene for oppdaterting av observasjonsdata.

5.1.1 Betraktningsfaktorer for observasjonsdata

En ideell algoritme for oppdatering av observasjonsdata følger målbanen kontinuerlig uavhengig av manøvermiljø. Dette kan være vanskelig da algoritmens ytelse i stor grad avhenger av tilgjengelig måldata og hvor godt disse utnyttes. En av utfordringene i denne oppgaven blir da å utnytte SNR data slik at korrekte posisjonsdata genereres.

Metodene for oppdatering av observasjonsdata som er beskrevet i kapittel 4.2 vil her simuleres og vurderes ut ifra egenskapene til den enkelte algoritme. Nedenfor er to aktuelle betraktningsfaktorer som tas med i kommentering av simuleringsresultatene.

SNR

Dersom algoritmene genererer observasjonsdata med voksende variasjoner kan blant annet utviklingen av SNR være årsak til det. Ved å se på beregningen av SNR i ligning 4-1 og 4-2 er det mulig forstå hvorfor SNR i målepunktene endres slik de gjør. SNR beregningene avhenger i stor grad av tre variable. Disse er avstand, R, asimut vinkelavvik, ΔAz, og elevasjons vinkelavvik, ΔEl og gir grenseverdiene i 5-1 og 5-2 for beregningen av SNR.

$$\lim_{R\to\infty} SNR \to 0 \tag{5-1}$$

$$\lim_{\Delta Az,\Delta El\to\infty} SNR \to 0 \tag{5-2}$$

SNR faller altså dersom avstanden og vinkelavviket vokser. Genererte observasjonsdata avhenger dermed i stor grad av SNR i de fire målepunktene og forholdet mellom dem. Dermed kan algoritmene til en viss grad vurderes på bakgrunn av hvordan de påvirkes av disse faktorene.

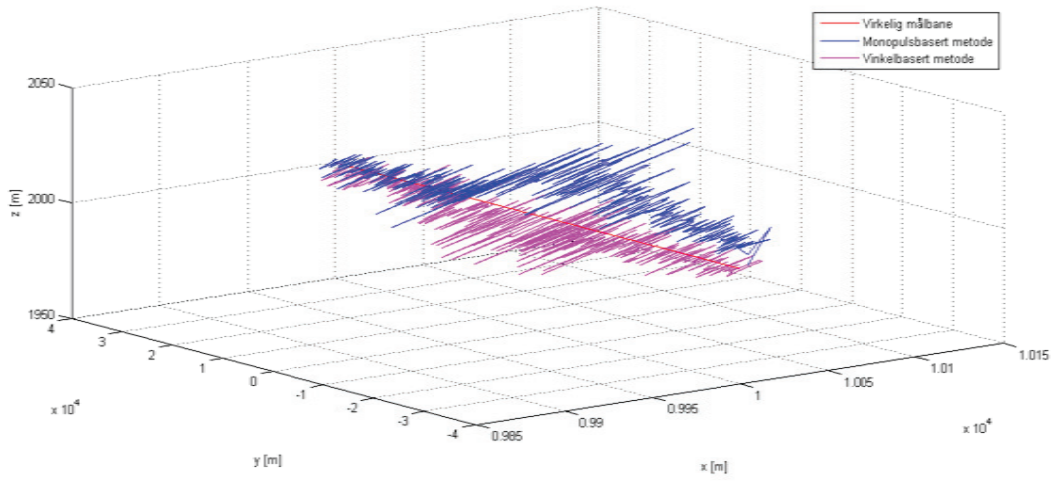
Samvirkende funksjoner

Siden algoritmene for oppdatering av observasjonsdata er rekursive vil innholdet i de forskjellige funksjonene være sterkt avhengige av hverandre. Dette kan gi opphav til komplikasjoner. Dersom en del av algoritmene svikter kan dette ha stor innvirkning på målfølgealgoritmen i sin helhet.

5.1.2 Genererte observasjonsdata

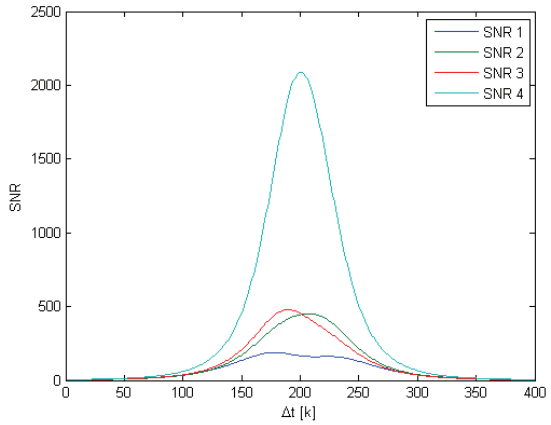
I denne delen vil monopulsbasert og vinkelbasert metode for oppdatering av observasjonsdata simuleres. Ved å se på utviklingen av SNR måldata og projeksjoner av målbanene kan metodene kommenteres.

Observasjonsdata for målbane 1

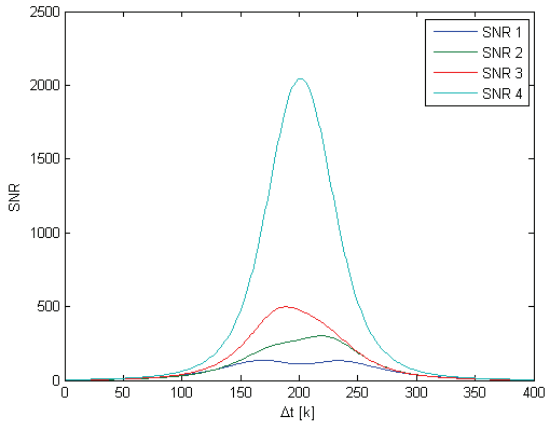


Figur 27: Observasjonsdata fra målbane 1

Figur 27 viser oppdaterte observasjonsdata for målbane 1 med monopulsbasert og vinkelbasert metode. Denne simuleringen gir oss en god indikasjon på den generelle oppførselen til metodene da målbanen følger en rettlinjet bevegelse. Algoritmenes feilestimat har i intervallet y = [$-2\cdot10^4$, $2\cdot10^4$] voksende avvik. For å forstå hvorfor variasjonene tar av må blant annet utviklingen av SNR i figurene 28 og 29 betraktes. Her er det tydelig at SNR i begge figurene vokser og når et toppunkt ved k = 200 før de minker igjen.



Figur 28: SNR ved monopulsbasert metode for målbane 1



Figur 29: SNR ved vinkelbasert metode for målbane 1

Avstanden, R, til målet er størst ved ytterpunktene $y_{min} = -4 \cdot 10^4$ og $y_{maks} = 4 \cdot 10^4$ slik at SNR også vil være lav i dette området. I intervallet y = [$-2 \cdot 10^4$, $2 \cdot 10^4$] minker den, og det er dermed ikke uventet at SNR når et maksimum i dette området.

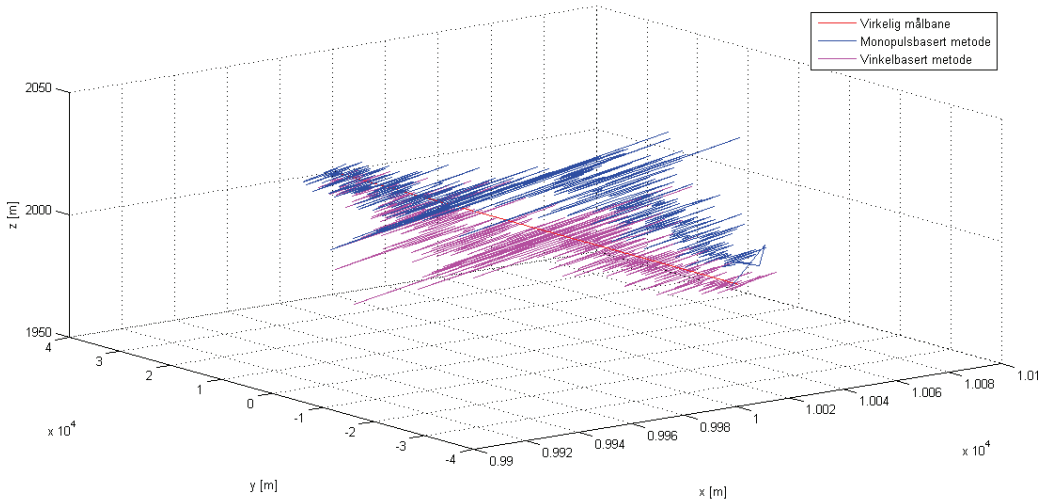
Et naturlig spørsmål er da om avstanden er årsaken til avviket i figur 27. I appendiks A.4 fremgår det at vinkeloppdateringen er uavhengig av R både for monopulsbasert og vinkelbasert metode.

Vinkeloppdateringen tar altså ikke hensyn til at kortere avstand i virkeligheten gir større utslag i Az_{k+1} og El_{k+1} enn ved lengre avstand.

For å utbedre denne feilen i vinkelbasert metode innføres konstantene K_{Az} og K_{El} som funksjoner av R, slik som vist i ligning 5-3 og 5-4. Oppdaterte observasjonsdata blir da som vist i figur 30.

$$K_{Az}(R) = 0.00157 \cdot \left[1 + 0.02 \cdot \left(-\log\left(\frac{R}{4 \cdot 10^4}\right)\right)\right] \tag{5-3}$$

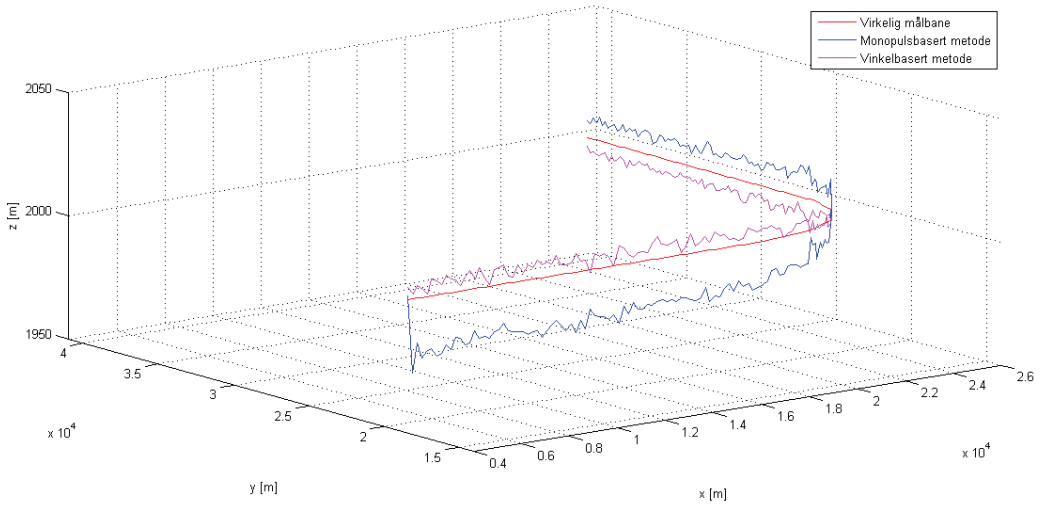
$$K_{El}(R) = 0.001116 \cdot \left[1 + 0.1 \cdot \left(-\log\left(\frac{R}{4 \cdot 10^4}\right)\right)\right] \tag{5-4}$$



Figur 30: Observasjonsdata fra målbane 1 med variable forsterkningsfaktorer

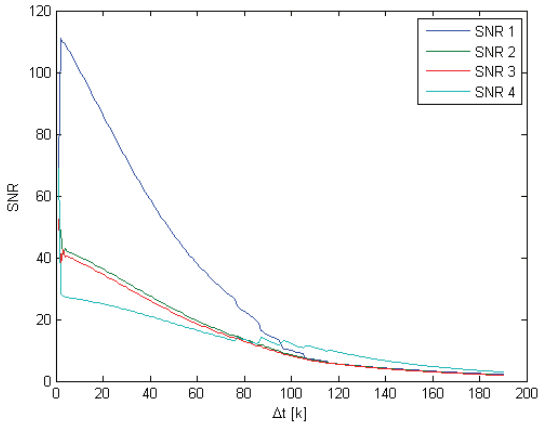
Det er her tydelig at vinkelbasert metode er mindre sensitiv for endring i R enn tidligere.

Observasjonsdata for målbane 2

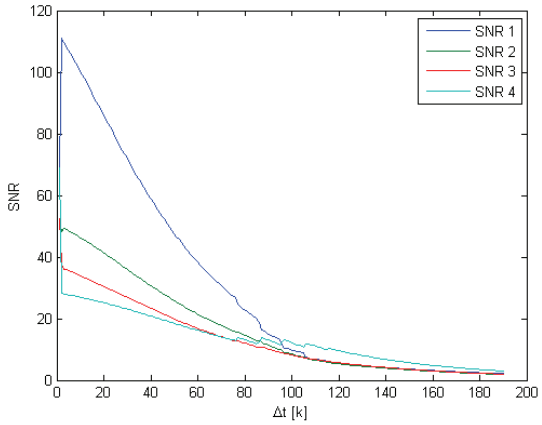


Figur 31: Observasjonsdata fra målbane 2

Figur 31 viser hvordan monopulsbasert og vinkelbasert metode fungerer med manøver i x- og y-retning. For monopulsbasert metode er det klart at avviket i observasjonsdata er størst i starten av simuleringen. Etter manøveren avtar avviket og målbanene avslutter med et mindre konstant avvik. Selve manøveren ser ut til å ha innvirkning både på monopulsbasert og vinkelbasert metode da begge skifter konstant avvik (bias).



Figur 32: SNR ved monopulsbasert metode for målbane 2

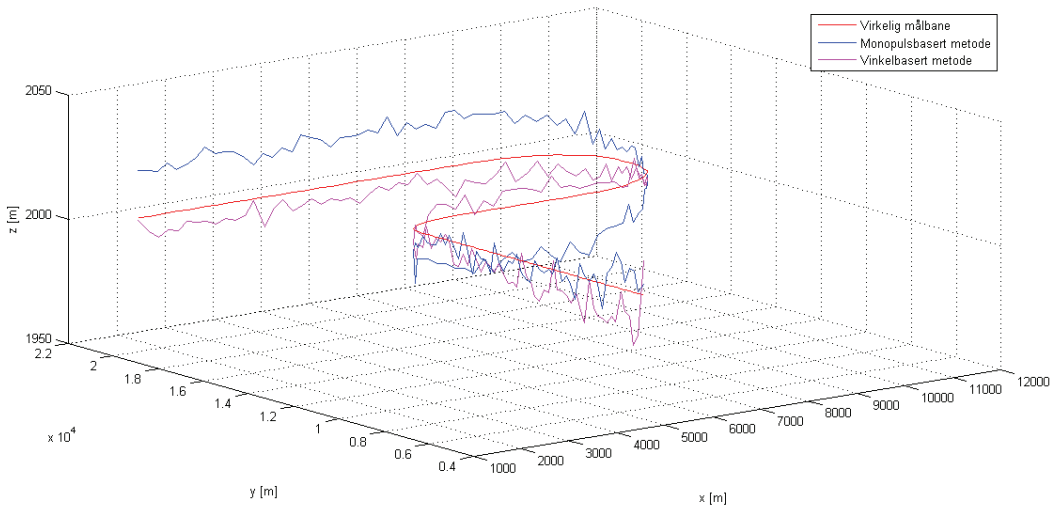


Figur 33: SNR ved vinkelbasert metode for målbane 2

Ved å se på figurene 32 og 33 er det tydelig at SNR i de fire målepunktene avtar raskt som følge av at avstanden øker. I figur 33 er det også tydelig at det i starten inntrer en økt respons av SNR i målepunkt 3. Noe som kan settes i sammenheng med det kraftige avviket monopulsbasert metode har i starten av simuleringene. Grunnen til dette avviket kan være at forsterkningen K_{Az} og K_{El} som benyttes i monopulsbasert metode er for dårlig dimensjonert.

I begge figurene er SNR i målepunkt 1 stor i starten fordi målet kontinuerlig beveger seg til venstre i asimut vinkel. Deretter øker SNR i målepunkt 4 fordi manøveren forårsaker en målbevegelse mot høyre.

Observasjonsdata for målbane 3

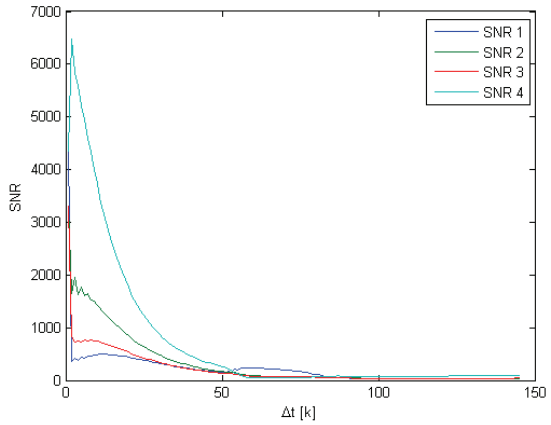


Figur 34: Observasjonsdata fra målbane 3

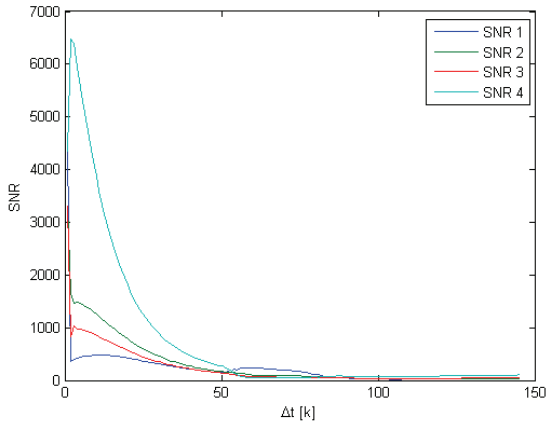
I figur 34 vises monopulsbasert og vinkelbasert metode for målbane 3 med to manøvre i x- og y-retning, hvorav den første er hardere enn den andre. Dette gir forskjellig akselerasjon i manøverens ulike partier.

Monopulsbasert metode har forholdsvis store variasjoner i intervallet etter første manøver. De to påfølgende manøvere utgjør ikke en stor avviksforskjell, men en skiftning i det konstante avviket.

For vinkelbasert metode er det i første del av målbanen et mindre konstant avvik med små variasjoner. Etter første manøver holdes avviket lite og konstant. De to påfølgende manøvere utgjør ingen stor forskjell i variasjon og gir kun en skiftning i det konstante avviket. Oppdaterte observasjonsdata er dermed nøyaktigere enn ved monopulsbasert metode.



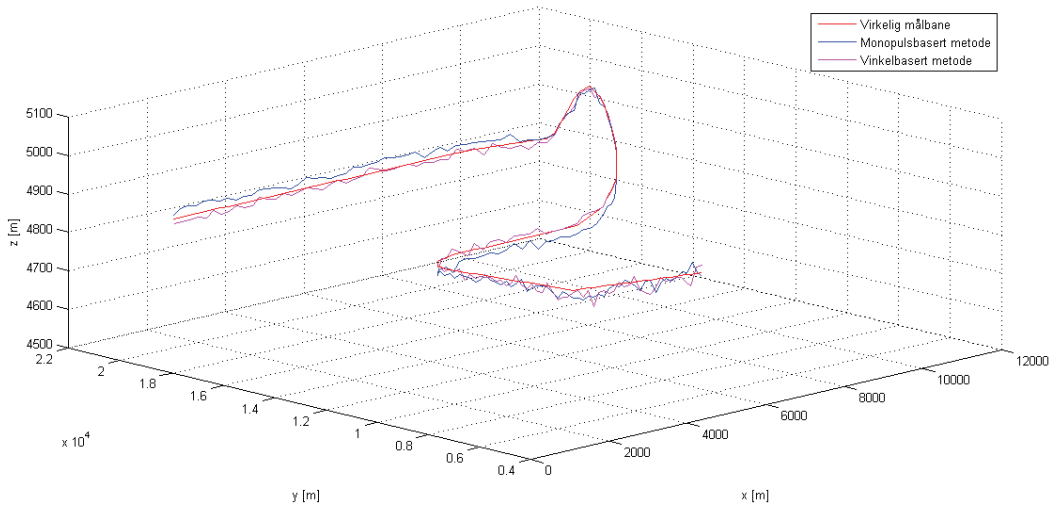
Figur 35: SNR ved monopulsbasert metode for målbane 3



Figur 36: SNR ved vinkelbasert metode for målbane 3

I likhet med forrige SNR måling oppstår det i starten av figurene 35 og 36 store verdier av SNR i alle målepunktene. Noe som igjen er indikasjon på at avstanden øker gjennom simuleringene. Gjennom simuleringene oppstår det store differanser mellom SNR i målepunkt 1 og 4 grunnet bevegelser i asimut vinkel. SNR i målepunkt 2 og 3 har større differanse ved monopulsbasert enn vinkelbasert metode og dette kan tolkes slik at monopulsbasert metode har mindre presise observasjonsdata i elevasjonsvinkel.

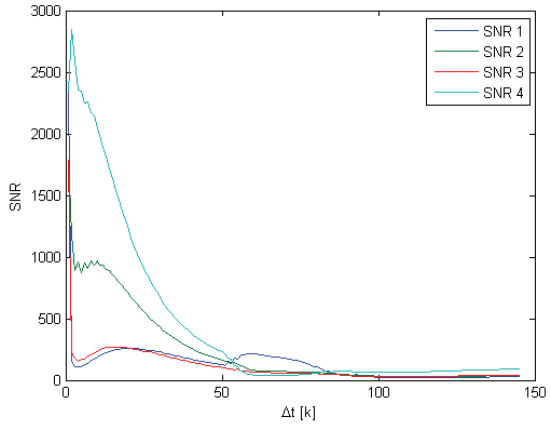
Observasjonsdata for målbane 4



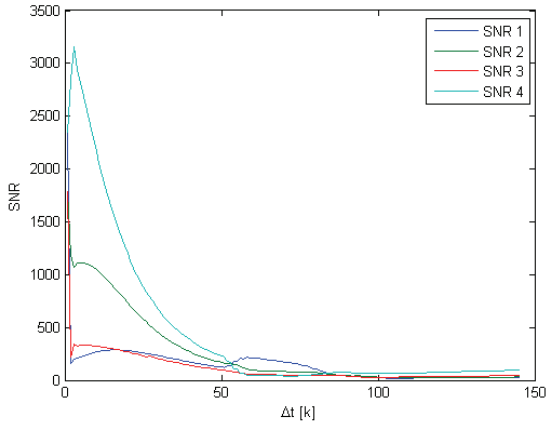
Figur 37: Observasjonsdata fra målbane 4

Figur 37 viser hvordan monopulsbasert og vinkelbasert metode fungerer med akselerasjon grunnet manøver i x-, y- og z-retning. Scenarioet setter algoritmene på en stor prøve da det oppstår kraftige manøvre i tre dimensjoner.

Monopulsbasert metode gir i fra starten av simuleringene større konstante avvik i form av skiftende bias. Ved vinkelbasert metode fremstår de oppdaterte observasjonsdataene som mindre varierende avvik rundt virkelig målbane.



Figur 38: SNR ved monopulsbasert metode for målbane 4

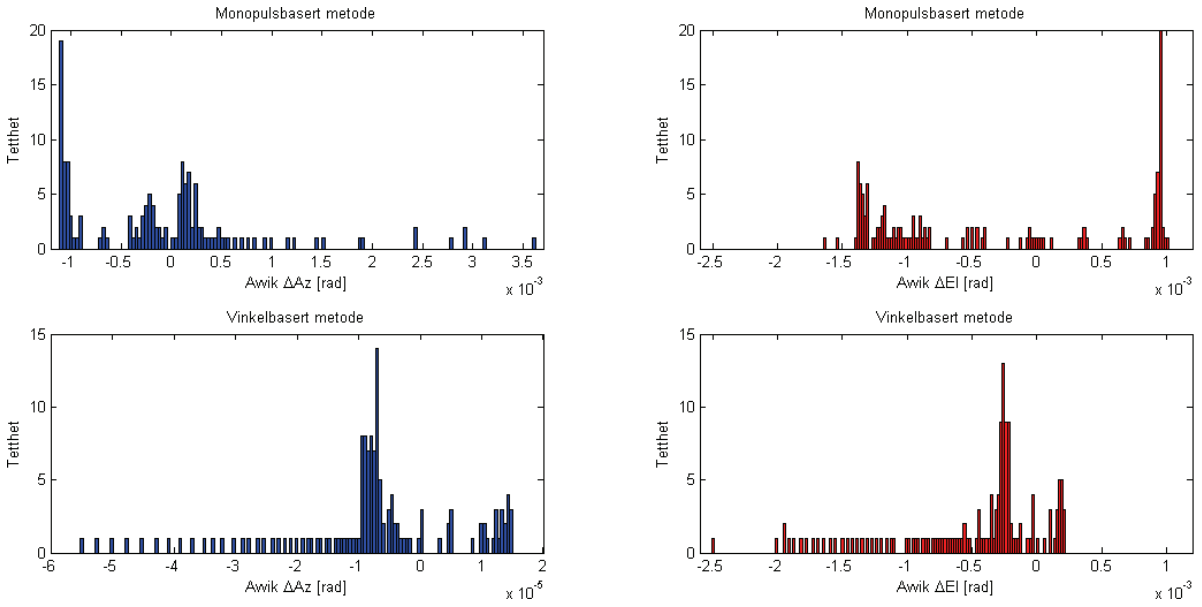


Figur 39: SNR ved vinkelbasert metode for målbane 4

I figur 38 og 39 går det frem at det er større variasjoner i SNR i målepunkt 2 og 3 ettersom målbanen også har manøvre i elevasjon. Forøvrig kan slutningene for avstanden, og vinkelavvik fra målbane 3 også trekkes her.

5.1.3 Vurdering av algoritmer for generering av observasjonsdata

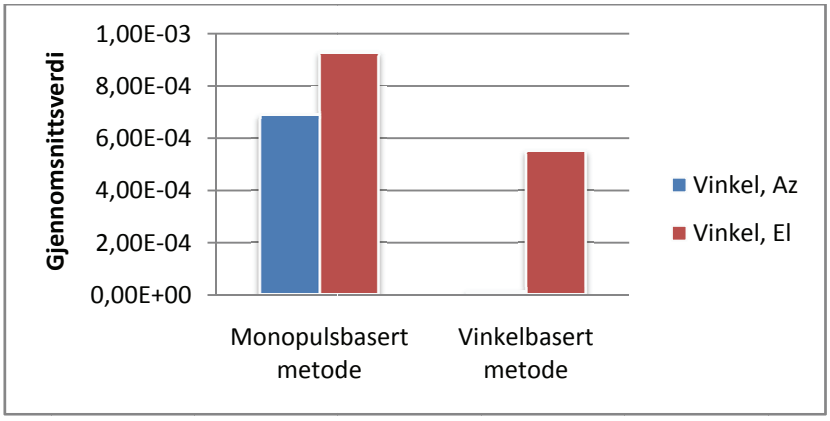
Simuleringene av monopulsbasert og vinkelbasert metode vil her vurderes ved hjelp av histogram. Histogrammet i figur 40 viser avvikstettheten i oppdaterte observasjonsdata for målbane 4. Figuren er også representativ for målbane 1, 2 og 3 da de har flere likhetstrekk. Disse tas dermed ikke mer i detalj her, men kan betraktes i appendiks A.5.



Figur 40: Histogram av oppdaterte observasjonsdata for målbane 4

Av figur 40 fremgår det at vinkelavviket med monopulsbasert metode er spredt over et større område enn med vinkelbasert metode. Dette kan skyldes at metoden ikke er nøyaktig nok, slik at den ikke klarer å kompensere for avviket. Avviket har også stor tetthet i enkelte avviksområder, noe som kan være en indikasjon på at monopulsbasert metode er dårlig proporsjonert i forhold til målingene av SNR.

Gjennomsnittlig avvik i oppdaterte observasjonsdata for målbane 4 er gitt i stolpediagrammet i figur 41. Monopulsbasert metode gir her større gjennomsnittlig avvik både i asimut- og elevasjonsvinkel. Dette samsvarer med at det i figur 40 er større spredning i avvik med denne metoden.



Figur 41: Gjennomsnittlig vinkelavvik i målbane 4

Diagrammer med gjennomsnittet av vinkelavvik for målbane 1,2 og 3 kan finnes i appendiks A.5

Ved målbanene fremsatt i denne oppgaven er det tydelig at vinkelbasert metode fungerer bedre enn monopulsbasert metode. Dette kan blant annet sees fra avviksforskjellen i figur 40 og 41. Histogrammene og gjennomsnittsmålingene indikerer at monopulsbasert metode er dårlig dimensjonert for de innkommende SNR dataene.

Siden monopulsbasert metode har vist seg å gi større konstante avvik i form av skiftende bias fra målbanene er det heller ingen grunn til å ta den med videre i testing av estimatorer. Dette skyldes at estimatorene som benyttes i denne oppgaven er basert på sekvenser med hvit støy med forventningsverdi lik 0. Monopulsbasert metode er dermed uegnet.

5.2 Generering av observasjonsstøy

Hensikten med EKF og IMM filteret er å estimere tilstander i systemer hvor målestøy og prosesstøy inngår som ukorrelerte sekvenser hvit støy med forventningsverdi lik 0. I utgangspunktet er ikke disse variablene kjent, men ved å finne deres kovariansmatriser Q og R, kan deres virkning tilnærmes. Estimatorene tar da hensyn til at det inngår forstyrrelser med en forutbestemt varians i prosessen. Siden det benyttes konstant kovarians er det fremdeles rom for utbedring av filteret ved å benytte variabel oppdatering av støyens kovarians.

Det er vanskelig å tilnærme prosesstøyen, men det finnes metoder for å tilnærme målestøyen. Dersom den kan tilnærmes ved hvert tidsskritt vil estimater være nærmere virkelig verdi, og nettopp dette er formålet med algoritmene for oppdatering av observasjonsstøy i kapittel 4.3.

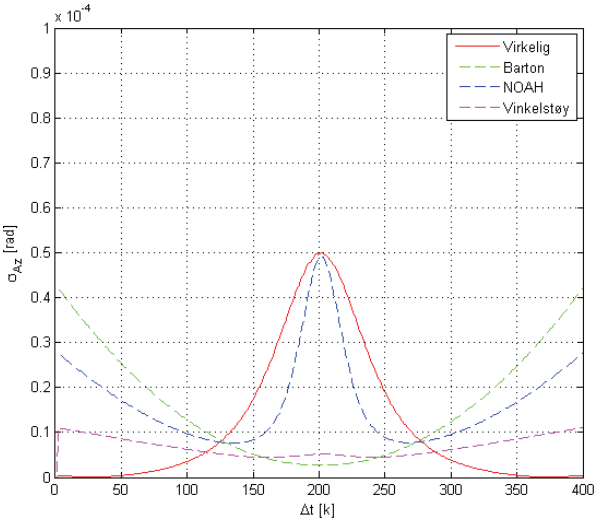
Siden målavstanden er virkelig avstand påvirket av hvit støy med kjent kovarians, genererer ikke metodene for estimering av observasjonsstøy standardavviket til dette estimatet. Det holdes konstant med $\sigma_R = 25$.

5.2.1 Generert observasjonsstøy

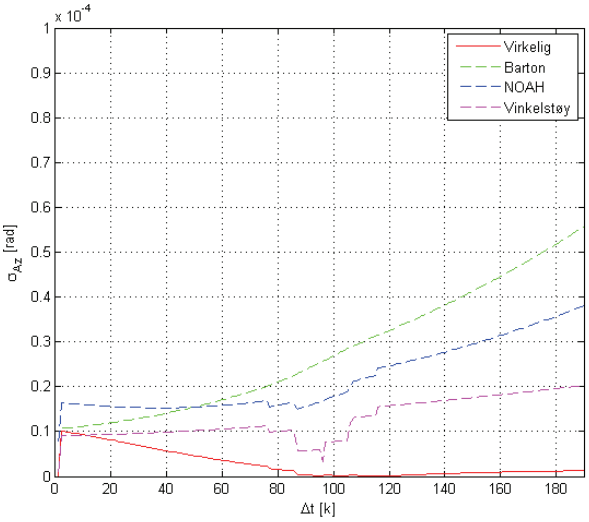
De forskjellige metodene for generering av observasjonsstøy vil her simuleres og vurderes mot virkelig støy.

Observasjonsstøy i asimut vinkel

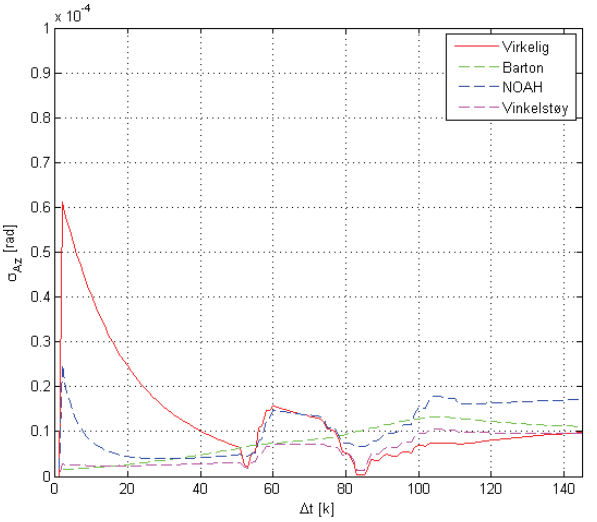
I figur 42 til 45 vises oppdaterte støydata og virkelig støy i asimut vinkel med hensyn til samplingstiden for målbane 1 til 4 med vinkelbasert metode. Den virkelige støyens standardavvik beregnes ved å ta absoluttverdien til virkelig vinkelposisjon minus oppdaterte observasjonsdata.



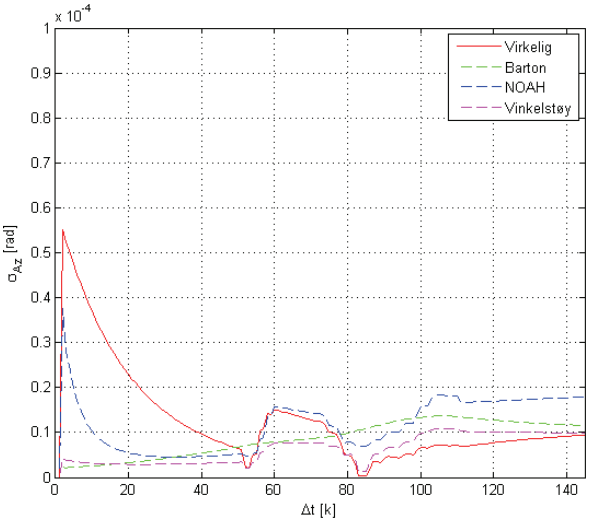
Figur 42: Støy i asimut vinkel ved målbane 1



Figur 43: Støy i asimut vinkel ved målbane 2



Figur 44: Støy i asimut vinkel ved målbane 3

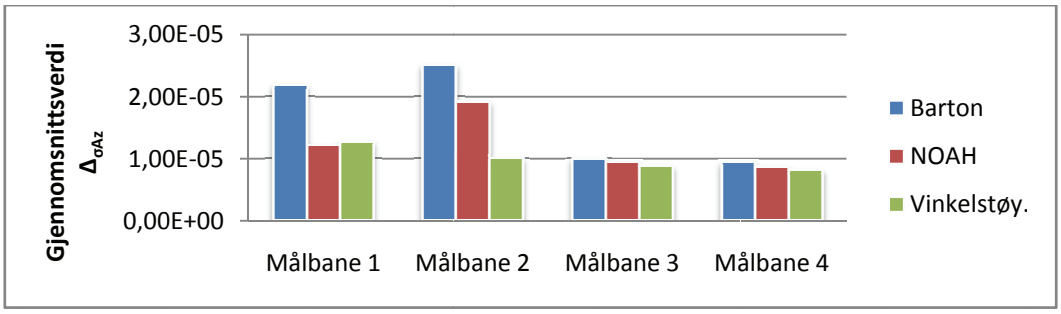


Figur 45: Støy i asimut vinkel ved målbane 4

Samtlige av metodene for generering av observasjonsstøy er basert på SNR forholdet i målepunktene, men fra ligning 4-34, 4-36, 4-37 og 4-38 fremgår det at NOAH's og vinkelstøybasert metode i større grad tar hensyn til differansen i vinkelavviket. Dette samsvarer med figurene 42 til 45 der det gjennomsnittlige avviket fra virkelig støy er vesentlig mindre.

I målbane 2 er det en stor manøver som vil gi økt avvik i algoritmen for oppdatering av observasjonsdata. Dette avviket får innvirkning på de mest avvikssensitive metodene for generering av observasjonsstøy, og kan gi en indikasjon på metoden som er best egnet i manøvrerende miljø. Vinkelstøybasert metode gir noe bedre tilnærming av støyen enn NOAH's metode.

Figur 44 og 45 har relativt likt mønster da målbane 3 og 4 er like sett fra asimut vinkel. NOAH's og vinkelstøybasert metode har bedre tilnærming av observasjonsstøy enn Bartons metode. Dette bekreftes av gjennomsnittverdien av avviket mellom virkelig og estimert observasjonsstøy i asimut vinkel i figur 46.

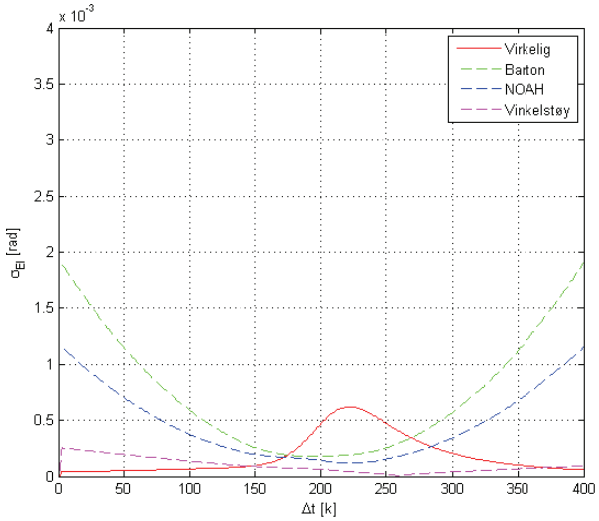


Figur 46: Gjennomsnittlig støyavvik i asimut vinkel

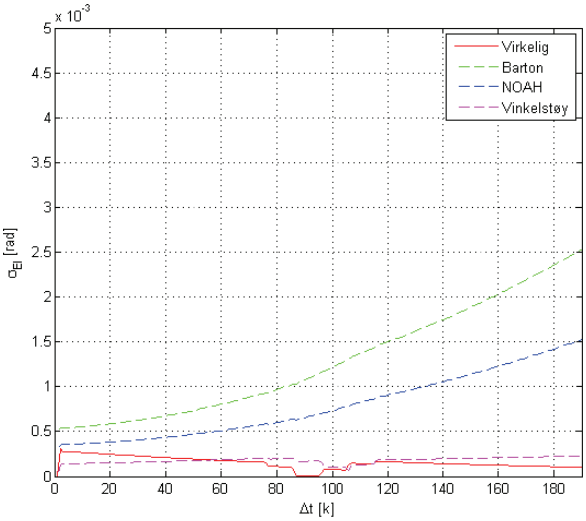
En tendens i figur 46 er at den gjennomsnittlige forskjellen i støyavviket ved asimut vinkel er svært lik både for NOAH's og vinkelstøybasert metode, mens Bartons metode gir større gjennomsnittlig støyavvik. Dette skyldes nok at de to førstnevnte metodene er spesielt utledet for vinkelbaserte målfølgemetoder.

Observasjonsstøy i elevasjonsvinkel

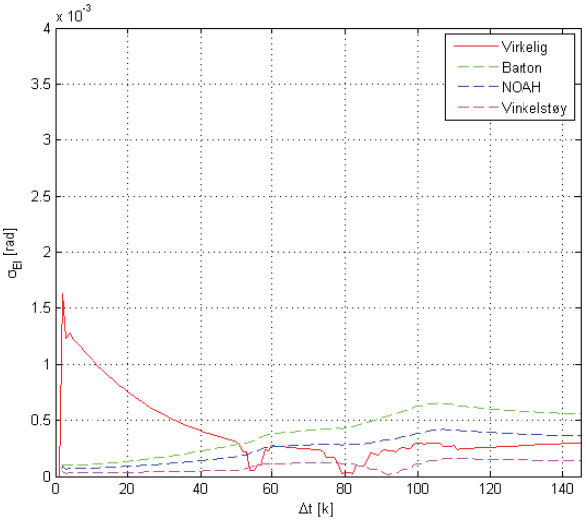
I figur 47 til 50 vises oppdaterte støydata og virkelig støy i elevasjonsvinkel med hensyn til samplingstiden for målbane 1 til 4.



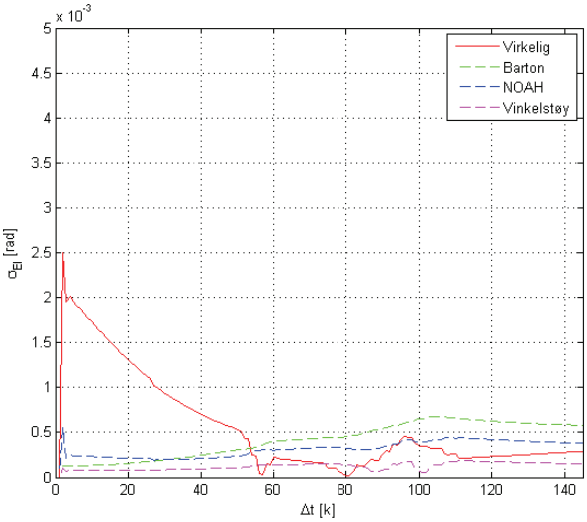
Figur 47: Støy i elevasjonsvinkel ved målbane 1



Figur 48: Støy i elevasjonsvinkel ved målbane 2

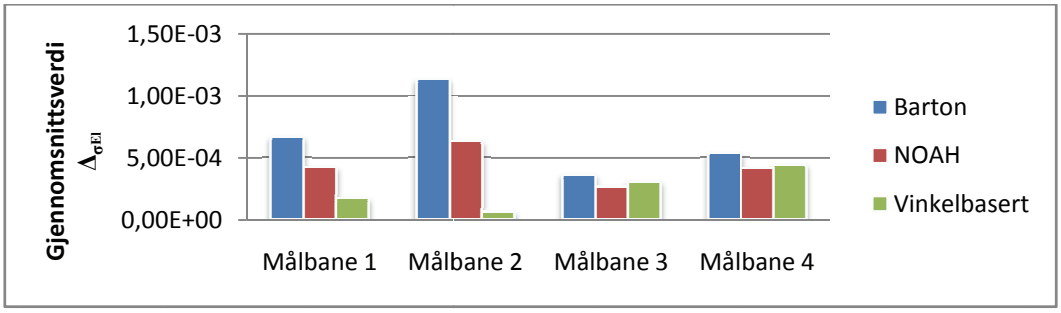


Figur 49: Støy i elevasjonsvinkel ved målbane 3



Figur 50: Støy i elevasjonsvinkel ved målbane 4

Vinkelstøybasert metode gir noe bedre tilnærming til virkelig observasjonsstøy i elevasjonsvinkel ved målbane 1 og 2. I figur 49 og 50 er allikevel Bartons og NOAH's metode noe bedre, selv om tilnærmingen er lite gjennomgående. Vinkelstøybasert metode gir noe bedre følging av konjunkturene i den virkelige observasjonsstøyen selv om avviket jevnt over er stort.



Figur 51: Gjennomsnittlig støyavvik i elevasjonsvinkel

Av figur 51 fremgår det at gjennomsnittet av avvik mellom virkelig og estimert observasjonsstøy i elevasjonsvinkel for målbane 1 og 2 er minst for vinkelstøybasert metode. Gjennomsnittet av avviket er størst for Bartons metode ved alle målbanene. I målbane 3 og 4 har NOAH's metode minst gjennomsnittlig avvik. Dette kan forstås ved å se på figur 49 og 50 der NOAH's metode i større grad tar hensyn til avviket i elevasjonsvinkel.

5.2.2 Vurdering av algoritmer for generering av observasjonsstøy

Etter å ha simulert de forskjellige metodene for oppdatering av observasjonsstøy med vinkelbasert metode kan noen enkle slutninger trekkes. I asimut og elevasjonsvinkel er støyavviket størst for Bartons metode. Det skyldes i stor grad at metoden ikke tar tilstrekkelig hensyn til avviket mellom SNR i målepunktene.

NOAH's og vinkelstøybasert metode gir langt bedre tilnærming av den virkelige observasjonsstøyen både for asimut og elevasjonsvinkel. Forskjellen mellom støyavviket ved de to metodene er forøvrig relativt liten.

Estimering av målbane 1 vil gi forholdsvis likt avvik både med konstant og variabel observasjonsstøy fordi den variable støyen ikke tilnærmes på en tilstrekkelig god måte. Målbane 2 bør kunne tilnærmes bedre med vinkelstøybasert metode enn med NOAH's metode men i målbane 3 og 4 bør metodene gi forholdsvis lik estimering.

Siden Bartons metode ikke klarer å følge den virkelige observasjonsstøyen er det ikke hensiktsmessig å ta den med i simulering av estimatorer med variabel observasjonsstøy.

5.3 Estimering

Estimeringsalgoritmene som er beskrevet under kapittel 4.4 vil her simuleres og vurderes. I estimering med målfølging skilles det mellom rettlinjede målbevegelser og manøvrerende mål med akselerasjon. Algoritmene vil simuleres i begge disse miljøene. Inngangen til filtrene vil som tidligere nevnt være oppdaterte observasjonsdata fra algoritmene i kapittel 4.2 med tillagt hvit støy. Målestøyen vil først holdes konstant og deretter genereres med algoritmene for observasjonsstøy.

5.3.1 Betraktningsfaktorer for estimat

Når egenskapene til estimatorene skal undersøkes er det vesentlig å ha gode verktøy for å analysere estimeringsalgoritmenes ytelse og nøyaktighet. Det vil i dette kapitlet fremlegges metoder for å analysere de statistiske egenskapene til estimatorene.

Projeksjoner

En god måte å fremstille egenskapene til algoritmene for tilstandsestimering i denne oppgaven er å vise projeksjoner av posisjons- og hastighetsestimatene. På den måten vil avviket mellom virkelig målbane og estimatene i forskjellige retninger fremstilles på en enkel måte.

Varians

Det kan være interessant å se på størrelsen til variansen for å anslå hvor spredtliggende estimatene er. Variansen er forventningsverdien til estimerte tilstander minus forventningsverdien av dem kvadrert, slik som vist i ligning 5-5.

$$var(\hat{x}) = E[(\hat{x} - E(\hat{x}))^2] \tag{5-5}$$

Siden simuleringene i denne oppgaven gjøres i tre dimensjoner vil ikke variansberegningen kunne gjennomføres på hele datasettet. Ved manøvre vil variansen til aksene som er berørt ikke gi fornuftige svar på grunn av leddet, $E(\hat{x})$. Dermed benyttes middelkvadratsfeilen.

Middelkvadratfeil

Dersom inngangene er signaler påvirket av hvit støy vil estimatorens varians kunne tilnærmes ved å beregne *middelkvadratsfeilen, "Mean Square Error (MSE)",* slik som vist i ligning 5-6.

$$MSE(\hat{x}) = var(\hat{x}) + B(\hat{x}) = E[(\hat{x} - x)^2] \tag{5-6}$$

Der $B(\hat{x})$ er bias.

Algoritmene for oppdatering av observasjonsdataene genererer en tilnærming av virkelig målbane før estimeringsalgoritmene tas i bruk. Observasjonen utgjør da farget støy i form av en hvit sekvens med små jevnlige bias fra virkelig målbane da tilnærmingen ikke er perfekt. Siden EKF og IMM filter er utviklet for signaler påvirket av hvit støy oppstår det dermed et problem.

I de fleste tilfeller utgjør allikevel ikke avviket i form av bias mellom genererte observasjonsdata og virkelig målbane et så stort problem at man ikke kan benytte middelkvadratsfeilen som et mål på estimeringsalgoritmenes ytelse.

Standardavvik

For å beregne estimatorens standardavvik tas kvadratroten til variansen slik som i ligning 5-7. Dette kalles *standardfeilen* til estimatoren og utgjør et mål på dens nøyaktighet ved hver tilstand. I en estimator der $\hat{x}$ er påvirket av bias kan variansen settes lik middelkvadratfeilen.

$$\sigma_{\hat{x}} = \sqrt{var(\hat{x})} \tag{5-7}$$

Konsistens

Kjikvadratfordeling kan brukes for å undersøke om estimatorer er *konsistent*. Altså om virkelig feil er konsistent med variansen som beregnes i estimatoren. Konsistens er nødvendig for estimatorens (filterets) optimalitet og er gitt ved følgende krav:

1. Estimeringsfeilen må ha gjennomsnitt lik null.
2. Estimeringsfeilens kovariansmatrise må være slik som beregnet av filteret.

I [10] er det fremlagt en konsistenstest for estimatorer. Dersom den normaliserte kvadrerte estimeringsfeilen *"Normalized estimation error squared (NEES)"* gitt ved ligning 5-8 ligger innen intervallet, $\epsilon(k) \in [r_1, r_2]$, vil krav 1 og 2 bekreftes.

$$\epsilon(k) = \tilde{x}^T(k|k)P(k|k)^{-1}\tilde{x}(k|k) \tag{5-8}$$

På bakgrunn av antall frihetsgrader og akseptert sannsynlighetsintervall velges $r_1 \approx 0$ og r_2 fra en kjikvadratssannsynlighetstabell. Den ensidige 97,5 % sannsynlighetsregionen for målbane 1 til 4 er da gitt ved en øvre grense $r_2 = 9{,}35$.

Absoluttfeil

I [10] fremlegges en måte å beregne filtrenes absolutte ytelse ved kvadratisk middelverdiberegning, "Root mean square (RMS)", slik som i ligning 5-9. I sammenheng med målfølging kan posisjon, hastighet, kurs osv inngå i disse beregningene. I stedet for at det som i standardfeil beregnes nøyaktighet for hver tilstand beregnes da den totale ytelsen i alle dimensjonene som en enhet ved å ta kvadratroten av samlede posisjons-, hastighets- og akselerasjonstilstander til sist.

$$RMS(\tilde{\xi}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\tilde{\xi}_i^2} \tag{5-9}$$

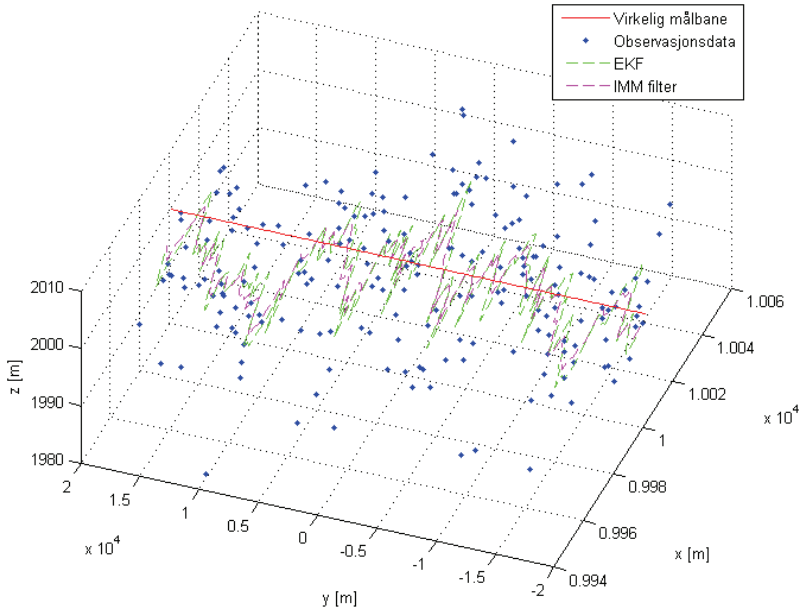
Her er $\tilde{\xi} = \hat{x} - x$ og N er antall tidsskritt. For hastighet byttes $\tilde{\xi}$ ut med $\dot{\tilde{\xi}}$.

5.3.2 Estimerte måldata

Dette kapitlet tar for seg simulering av estimatorene beskrevet i kapittel 4.4. I tillegg til ulike projeksjoner fremlegges de statistiske egenskapene til de estimerte tilstandene i tabeller. I appendiks A.6 fremlegges konstanter som benyttes i simuleringen av estimatorene.

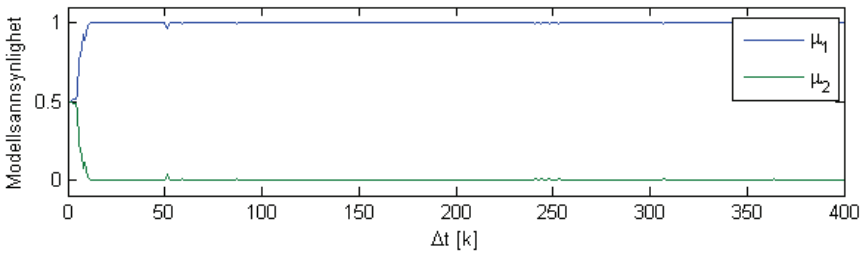
Rettlinjet målbevegelse

I figur 52 vises projeksjonen av estimatorsimuleringene for målbane 1. Her fremgår det EKF og IMM filtrenes estimater er forholdsvis like. Begge algoritmene har estimering som forbedrer målfølgingen betraktelig, men IMM filteret har litt mindre avvik.



Figur 52: Estimering av målbane 1

Grunnen til at forskjellen mellom EKF og IMM filterets estimater er små kan forklares ved figur 53 der modellsannsynligheten i IMM filteret er illustrert. Fraværet av akselerasjon medfører at IMM filteret kun estimerer med hastighetsmodellen. EKF filteret består kun av en hastighetsmodell og dermed fungerer estimatorene svært likt.



Figur 53: Modellsannsynlighet ved estimering av målbane 1

IMM filteret ser fra tabell 4 ut til å tilnærme signalet med lavere absoluttfeil og standardfeil. Dette kan skyldes at hastighetsmodellen i IMM filteret estimerer flere tilstander enn EKF filteret. Standardfeilen er minst i høyde siden ingen manøver finner sted i elevasjon

	Standardfeil			Absoluttfeil
	σ_x	σ_y	σ_z	RMS(ξ)
EKF	7,57	11,91	4,26	14,70
IMM filter	7,41	11,37	4,09	14,16

Tabell 4: Standardfeil og absoluttfeil ved estimering av målbane 1

Den fargede støyen i form av bias, som er forårsaket av genererte observasjonsdata, reduseres ikke merkbart med estimatene fra EKF eller IMM filteret. I [12, Haugen, F.] beskrives en metode for hvordan estimatene kan utbedres. Ved å augmentere systemmodellene i filtrene med tilstander som representerer et konstant avvik i dimensjonene x, y og z kan den fargede støyen reduseres. Tilstandsvektorene og systemmatrisen ble dermed utvidet slik som i ligning 5-10 og 5-11 for EKF og tilsvarende for IMM filteret.

$$\bar{x}_k = \begin{bmatrix} \hat{x}_k \\ d_k \end{bmatrix} \tag{5-10}$$

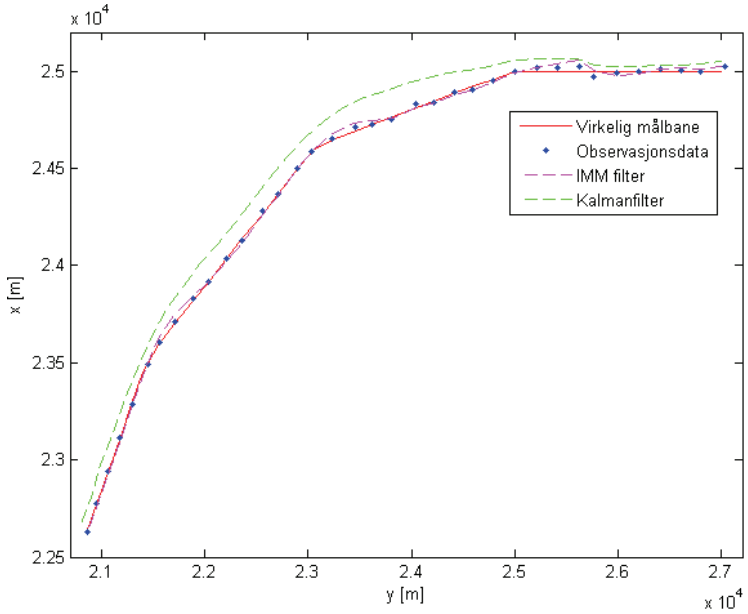
$$d_k = [d_k^1 \quad d_k^2 \quad d_k^3]^T \tag{5-11}$$

$$\Phi_k = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 & K_1 & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & K_2 & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & K_3 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (5\text{-}12)$$

Augmenteringen gav ingen merkbar effekt på simuleringene. Dette kan skyldes at støymodellen er for unøyaktig eller at prosesstøyens kovariansmatrise må innstilles bedre for støykomponentene i tilstandsvektoren.

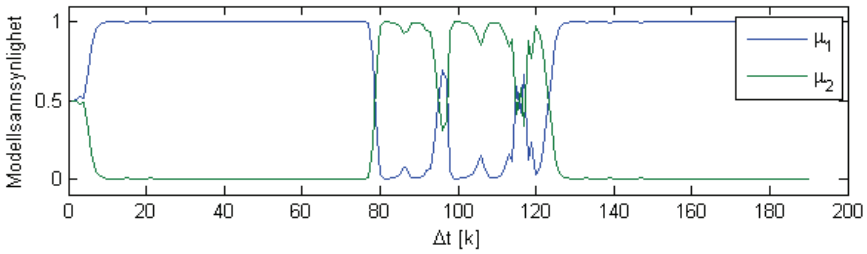
Svakt manøvrerende mål

Figur 54 viser manøveren i målbane 2 i x-y-planet. Her er det tydelig at avviket mellom virkelig og estimert målbane er størst med EKF. IMM filteret ser ut til å følge målet med større nøyaktighet under manøveren grunnet den dynamiske vektleggingen av estimatet fra hastighets- og akselerasjonsmodellene.



Figur 54: Estimering av målbane 2 i x-y-planet

IMM filteret vektlegger i figur 55 akselerasjonsmodellen i oppdatering av estimater med en gang manøveren inntreffer og går tilbake til hastighetsmodellen når målet igjen går i en rettlinjet bevegelse. Manøveren i målbane 2 er svak og dermed forekommer de små ujevnhetene i modellsannsynligheten.



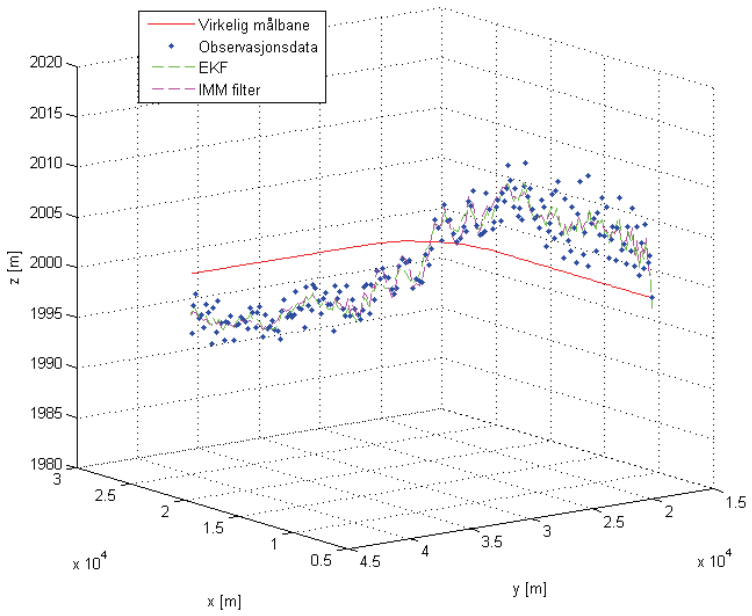
Figur 55: Modellsannsynlighet ved estimering av målbane 2

I tabell 5 er resultatene av absoluttfeilen og standardfeilen fremlagt for EKF og IMM filteret. Her er det tydelig at ytelsesforskjellen er stor. Antagelsen om at IMM filteret gir bedre estimater ved manøver er altså i overensstemmelse med disse resultatene.

	Standardfeil			Absoluttfeil
	σ_x	σ_y	σ_z	RMS(ξ)
EKF	32,79	32,10	5,09	46,17
IMM filter	13,61	15,50	5,01	21,20

Tabell 5: Standardfeil og absoluttfeil ved estimering av målbane 2

Siden estimatene i figur 56 går fra å være over til å være under den virkelige målbanen økes det totale avviket og nøyaktigheten i estimatene reduseres.

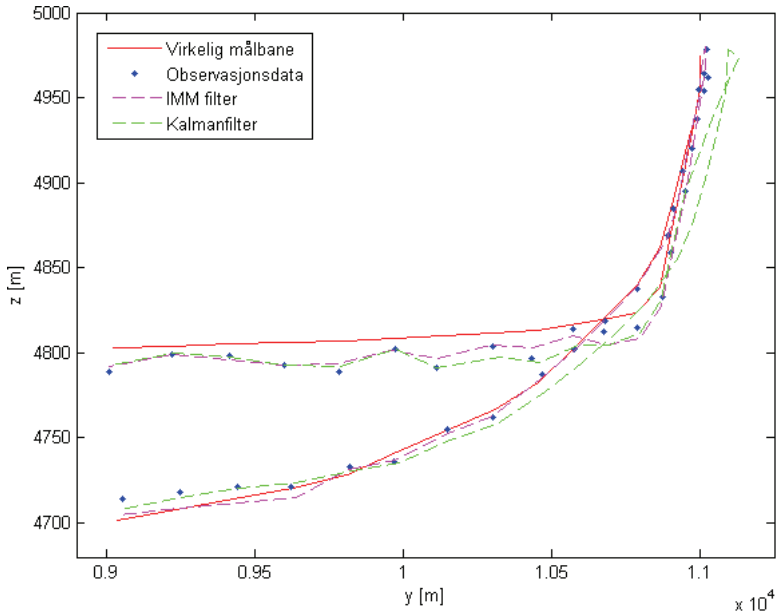


Figur 56: Estimering av målbane 2

Hardt manøvrerende mål

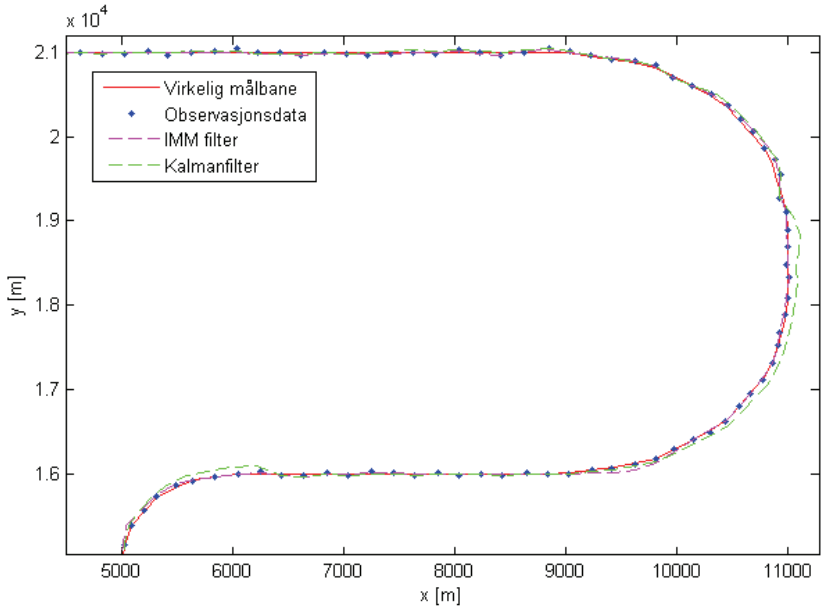
Målbane 4 har like manøvre som målbane 3 i x-y-planet slik at estimatorenes egenskaper blir de samme i denne dimensjonen. Estimatorene simuleres dermed kun med målbane 4 da denne også består av harde manøvre i høyde.

I figur 57 vises et utsnitt av målbane 4 i y-z-planet. Her fremgår det at avviket i estimatet ved EKF øker med manøver i høyde. Dette samsvarer med oppbygningen av EKF der manøverdeteksjonsalgoritmen påfører filteret økt prosesstøy ved kraftig nok manøver. Dette gir ingen jevn overgang slik som ved IMM filteret.



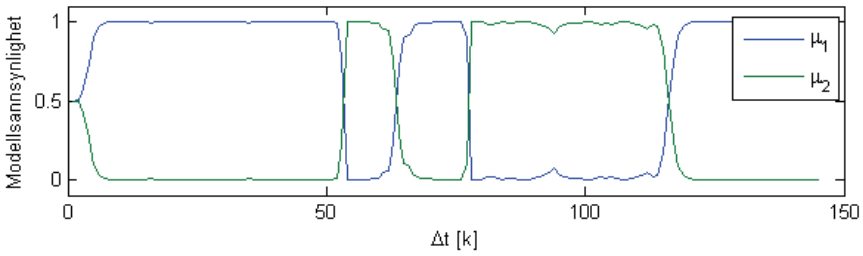
Figur 57: Estimering av målbane 4 i y-z-planet

Manøvrene har samme utslag i x-y-planet illustrert i figur 58. I begge manøvrene gir EKF oversving mens IMM filteret følger målet med større nøyaktighet.



Figur 58: Estimering av målbane 4 i x-y-planet

I figur 59 vises modellsannsynligheten for IMM filterets modeller gjennom simuleringen av målbane 4. De to manøvrene i figur 58 kommer til syne i figur 59 som kraftige vektninger av akselerasjonsmodellen. Dette indikerer at IMM filteret fungerer slik det skal og foretar nødvendige modelltransisjoner.

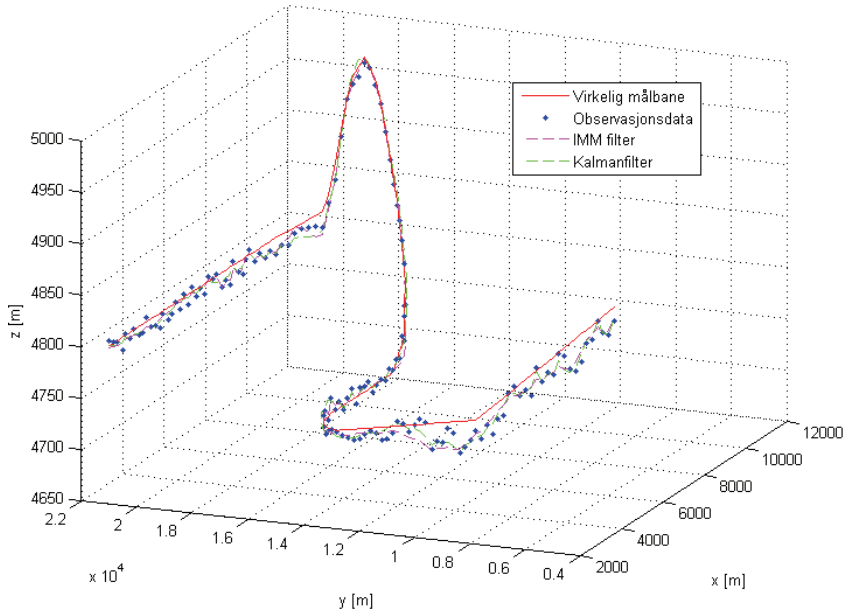


Figur 59: Modellsannsynlighet ved estimering av målbane 4

Standardfeilen for EKF og IMM filteret i tabell 6 bekrefter at EKF har større avvik, og da spesielt i manøverpartier. Dette er også i overensstemmelse med absoluttfeilen hvor forskjellen mellom estimatorene er stor. Standardfeilen er her vesentlig større enn i de andre målbanene i høyde grunnet manøvrene i elevasjon.

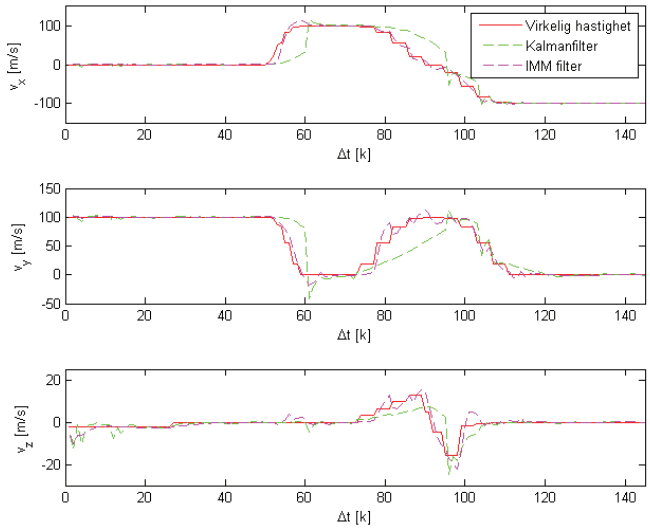
	Standardfeil			Absoluttfeil
	σ_x	σ_y	σ_z	RMS(ξ)
EKF	25,33	29,38	10,68	40,22
IMM filter	9,16	18,01	9,88	22,42

Tabell 6: Standardfeil og absoluttfeil ved estimering av målbane 4



Figur 60: Estimering av målbane 4

De estimerte tilstandene av hastigheten i målbane 4 med EKF og IMM filteret er illustrert i figur 60. Her tydeliggjøres det ytterligere at IMM filteret tilnærmer seg den virkelige målhastigheten bedre enn EKF. Figurer av de estimerte hastighetene i målbane 2 til 3 tas ikke med her da de gir tilsvarende resultater, men kan finnes i appendiks A.7.



Figur 61: Hastighetsestimering ved målbane 4

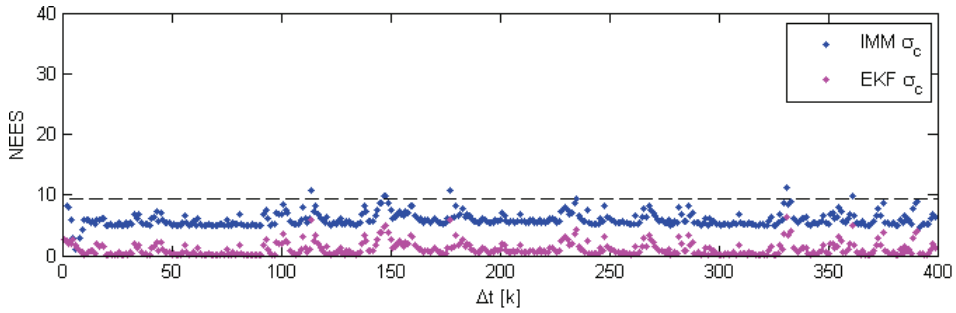
Singularitetsproblem

Ved simulering av estimatorer med variabel observasjonsstøy oppstod det først problemer med inverteringen i beregning av KF-forsterkningen i IMM filterets akselerasjonsmodell. Dette skyldes at nevnerleddets resulterende matrise blir singulær når målestøymatrisen, R_k, og kovariansmatrisen, P_k er dårlig dimensjonert. Dette ble håndtert ved å sette inn kode som sjekker kovariansmatrisen for singulariteter. Dersom singulariteter oppstår adderes da en liten verdi, ε, til diagonalelementene av P_k.

Etter at EKF og IMM filteret ble innstilt med større diagonalkomponenter i prosessmatrisen, Q_k, forsvant singularitetsproblemet. Det kan allikevel være greit å være klar over det og hvordan det lar seg løse dersom systemet skal videreutvikles.

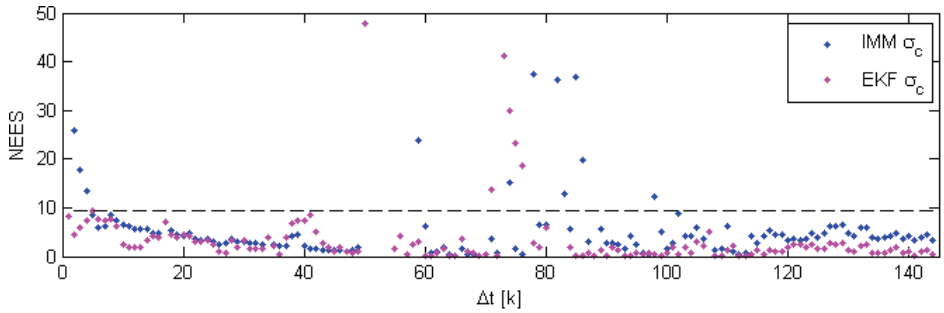
Konsistenstest

Ved å beregne NEES av estimator kan det avgjøres om den er konsistent eller ikke. I figur 62 vises NEES for EKF og IMM filteret med konstant observasjonsstøy gjennom målbane 1. Som beskrevet i kapittel 5.3.1 må resultatet av NEES beregningene ligge i intervallet gitt av, $\epsilon(k) \in [0, 9{,}35]$. Av figuren fremgår det at EKF ikke overstiger grensen, men at IMM filteret overstiger grensen 5 steder. Dette er akseptabelt når antall tidsskritt er gitt ved $\Delta t = 400$.



62: NEES ved konstant observasjonsstøy i målbane 1

I figur 63 er tilsvarende NEES beregningene av målbane 4 vist. Her overstiger EKF og IMM filteret grensen henholdsvis 16 og 34 ganger. IMM filteret er da i grenseland for hva som kan aksepteres ved 145 tidsskritt. Estimatoren sies da å være for *optimistisk*.



63: NEES ved konstant observasjonsstøy i målbane 4

Innstilling av et filter som opererer i et manøvrerende miljø kan være krevende fordi støy som dette ikke kan betraktes som hvit. Ved stille inn filteret nærmere konsistens vil absoluttfeilen som regel økes. Absoluttfeilen er nemlig unormalisert mens den statistiske beregningen er normalisert. Målet blir altså å innstille filteret uten å forårsake en økning i absoluttfeilen.

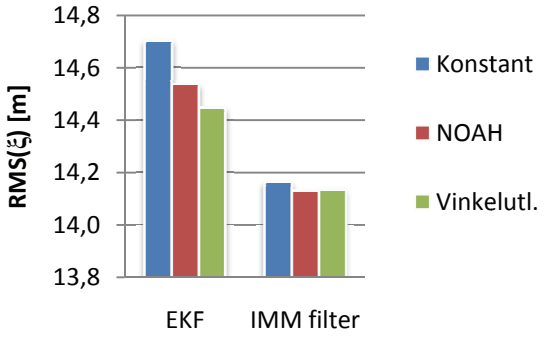
IMM filteret ble dermed forsøkt innstilt med ulik prosesstøymatrise, Q_k, og forskjellig transisjonsmatrise, p_k, men dette gav store endringer i absoluttfeilen. Dermed var det ikke mulig å redusere antall ganger grensene i $\epsilon(k)$ overstiges.

Det tas ikke med figurer av NEES for målbane 2 og 3 da resultatene sammenfaller med NEES for målbane 4.

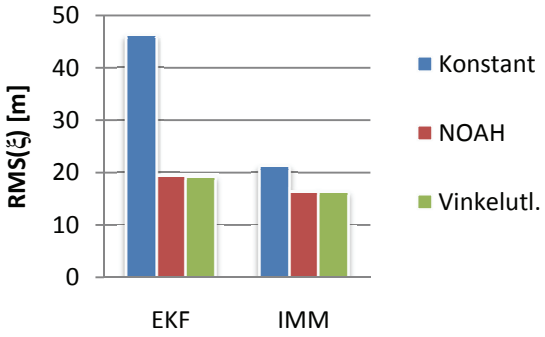
5.3.3 Estimerte måldata ved variabel observasjonsstøy

Under simulering av estimeringsalgoritmene i forrige kapittel ble observasjonsstøyen holdt konstant. I denne delen vil målestøyen genereres ved de variable støyestimeringsmetodene beskrevet i kapittel 4.3.

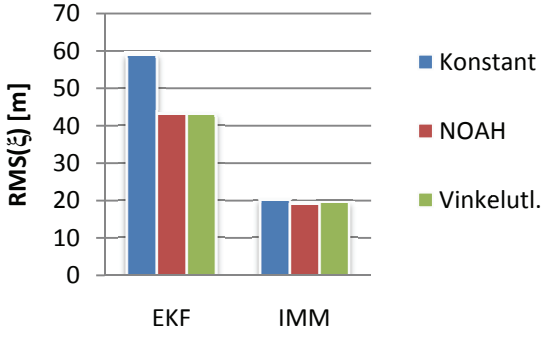
I figur 64 til 67 er absoluttfeilen for EKF og IMM filter med ulik variabel observasjonsstøy illustrert med stolpediagram. Simuleringene ble gjentatt 100 ganger med ulike sekvenser hvit støy før et gjennomsnitt av absoluttfeilen ble beregnet. Dette for å øke nøyaktigheten i resultatene. Dersom det er av interesse å se estimerte avvik delt i x-, y- og z-retning er figurer med stolpediagram av middelkvadratfeilen vist i appendiks A.8.



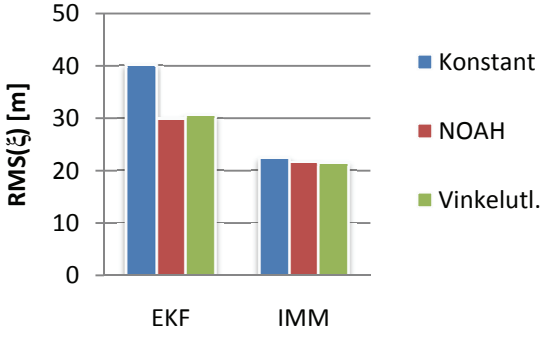
Figur 64: Absoluttfeil for målbane 1



Figur 65: Absoluttfeil for målbane 2



Figur 66: Absoluttfeil for målbane 3



Figur 67: Absoluttfeil for målbane 4

Her er det tydelig at kjennskap til variabel målestøy gir bedre estimering, men virkningsgraden varierer fra målbane til målbane. IMM filteret yter i alle tilfellene bedre enn EKF, men også her varierer virkningsgraden.

I målbane 1 foretas ingen manøver og dermed blir estimert avviksforskjell mellom EKF og IMM filteret lite. Allikevel genereres estimater med mindre avvik med variabel observasjonsstøy for begge metodene. Observasjonsstøy generert med NOAH's og vinkelstøybasert metode gir også svært like estimater.

Av figurene 65 til 67 fremgår det at avviksforskjellen mellom EKF og IMM filteret øker når simuleringene kjøres i manøvrerende miljø. Den variable målestøyen gir også ytterligere forbedret effekt på estimatene.

Grunnen til dette er at den konstante tilnærmingen av målestøyen kun er basert på variansen av den hvite støyen som tillegges den virkelige målbanen og ikke tar hensyn til effekten algoritmene for oppdatering av observasjonsdata har på støyen.

Ved store manøvre vil de genererte observasjonsdataene være sterkt påvirket av algoritmenes nøyaktighet. Dette gjelder også rettlinjede målbevegelser men siden akselerasjon ikke forekommer, genereres riktigere observasjonsdata. Variabel observasjonsstøy vil dermed ha større effekt ved estimering av manøvrerende målbevegelser.

Fra figurene 42 til 46 og 47 til 50 i kapittel 5.2 ble det fastslått at NOAH's metode og vinkelstøybasert metode gav relativt lik tilnærming av observasjonsstøyen og dette bekreftes av absoluttfeilen i figurene 64 til 67. I tabell 7 er den prosentvise forbedringen ved å gå fra konstant til variabel observasjonsstøy presentert både for NOAH's og vinkelstøybasert metode.

	Målbane 1		Målbane 2		Målbane 3		Målbane 4	
	NOAH	Vinkelst.	NOAH	Vinkelst.	NOAH	Vinkelst.	NOAH	Vinkelst.
EKF %	1,15	1,77	138,83	140,02	36,71	36,80	34,51	31,50
IMM %	0,23	0,19	30,19	30,43	5,15	3,25	3,73	4,40

Tabell 7: Estimeringsavvikets prosentvise forbedring fra konstant til variabel observasjonsstøy

Her er fremgår det at den prosentvise forbedringen er større med EKF enn IMM filter. Dette skyldes at IMM filteret i utgangspunktet gir mindre avvik slik at den variable observasjonsstøyen har mindre effekt.

Figurer av estimert hastighet tas ikke med for estimering med variabel støy fordi figur 61 har tilnærmet like estimater og dermed er tilstrekkelig beskrivende.

Konsistenstest

I appendiks A.9 er det fremlagt figurer av NEES for EKF og IMM filteret med variabel observasjonsstøy. I utgangspunktet viste variabel observasjonsstøy seg å forverre resultatene av konsistenstesten. En mindre økning av forsterkningen i beregningen av variabel observasjonsstøy gav forbedrede resultater uten endringer i absoluttfeilen. Simuleringen gav da tilnærmet like resultater som med konstant observasjonsstøy. IMM filteret viste seg å være dårligere tilpasset i manøvrerende miljø og er dermed optimistisk også med variabel observasjonsstøy.

5.3.4 Vurdering av estimeringsalgoritmer

EKF og IMM filtre er velkjente estimatorer for bruk innen målfølging med manøvrerende miljø. Selv om begge benytter KF i sine modeller har de forskjellige måter å takle manøvre på og dette innebærer at de genererer ulike estimater. Allikevel har de det til felles at blant annet målestøyens kovariansmatrise i modellenes KF brukes som tuningsparameter for regulatorene. Det er dermed interessant å simulere kombinasjonen av estimatorer og ulik generering av observasjonsstøy. Resultatet av simuleringene vil her bli kommentert.

EKF og IMM filter

Simuleringen av EKF og IMM filteret med den rettlinjede målbevegelsen i målbane 1 gav gode resultater. IMM filteret hadde 11,89 % mindre absoluttfeil enn EKF. Grunnen kan være at hastighetsmodellen som ble brukt i IMM filteret er utvidet med tre tilstander og dermed er en bedre tilnærming av den virkelige systemmodellen.

Et av kravene for at KF skal fungere godt er at innkommende signaler er påvirket av hvit støy. Algoritmene for oppdatering av observasjonsdata etterlater seg allikevel farget støy på signalet som bidrar til at estimatene ikke blir optimale. EKF og IMM filteret ble dermed utvidet med tre tilstander for å forsøke å integrere vekk støyen. Dette gav ingen merkbar forbedring, noe som kan skyldes at de augmenterte tilstandene ikke ble realisert med riktig nok modell.

De harde manøvrene i målbane 4 setter estimatorene på en prøve. Som forventet yter IMM ut ifra absoluttfeilen bedre da den beregner modellsannsynligheten mellom en hastighetsmodell og en akselerasjonsmodell, mens EKF kun øker observasjonsstøyen. Dette kan blant annet ses av projeksjonene som er fremstilt av svake og harde manøvre gjennom dette kapitlet. EKF får lettere oversving i manøveren slik at absoluttfeilen øker, og dermed utgjør absoluttfeilen med IMM filter i disse målbanene bare halvparten av EKF.

I målbaner med harde manøvre oppstod singularitetsproblemer som ble løst ved å legge korrigeringsverdier til diagonalen av kovariansmatrisen, P_k. Problemet forsvant etter innstilling av matrisen, Q_k, men det er allikevel greit å være klar over hvordan det kan løses dersom problemet gjenoppstår.

Konsistenstesten viste at manøvrerende miljø kan være svært krevende både for EKF og IMM filteret. Ved rettlinjet bevegelse er det tydelig at begge estimatorene kan betraktes som *optimale*. I sterkt manøvrerende miljø kan allikevel IMM filteret bli *optimistisk* og dette er en indikasjon på at det er dårligere tilpasset systemet eller at det er mer følsomt for manøvre.

Konstant og variabel observasjonsstøy

Etter at EKF og IMM filteret er simulert med konstant og variabel observasjonsstøy fremgår det at sistnevnte gir vesentlig mindre estimeringsavvik. Dette gjelder både for rettlinjede og manøvrerende målbaner. NOAH's metode og vinkelstøybasert metode gir for øvrig svært like resultater. Grunnen til dette er at observasjonsstøy generert med begge metodene gir gjennomsnittlig lik tilnærming til virkelig målestøy selv om karakteristikken er noe forskjellig.

I kapittel 5.2 ble det argumentert for at vinkelstøybasert metode skulle gi bedre estimater ved målbane 2 enn NOAH's metode siden observasjonsstøyen lå nærmere virkelig målestøy. Dette viste seg ikke å stemme og kan skyldes at prosesstøyen reduserte effekten av målestøyen.

Både NOAH's metode og vinkelstøybasert metode gir større avviksforskjeller enn konstant observasjonsstøy ved manøvrerende målbevegelser. Dette skyldes at det i utgangspunktet er større avvik i manøvrerende miljø slik at metodene får større effekt.

Etter simuleringen av EKF og IMM filtrene var det tydelig at IMM gav mindre avvik og større nøyaktighet. Med variabel observasjonsstøy hadde IMM filteret dermed prosentvis lavere forbedring enn EKF. Dette fordi IMM filteret i utgangspunktet gir bedre resultater.

Variabel observasjonsstøy gav svært like resultater som konstant observasjonsstøy ved konsistenstesting. Filtrene var dermed stort sett optimale, mens IMM filteret viste seg å være optimistisk ved harde manøvre.

6 Diskusjon

Her følger et tilbakeblikk på det som er gjort i oppgaven med diskusjon rundt utførelse og resultater. Problemstillinger som har oppstått under gjennomføringen tas opp og forslag til utbedring kommenteres.

Observasjonsdata

Flere prinsipper for oppdatering av observasjonsdata ble i kapittel 3.2 presentert og av disse ble det valgt å bruke metoder basert på ACM og SL. Altså monopulsbasert og vinkelbasert metode. Førstnevnte gav større avvik enn sistnevnte i form av bias grunnet dårlig dimensjonering til SNR data. Da EKF og IMM filteret i utgangspunktet ikke er egnet for denne typen støy ble metoden valgt bort.

Vinkelbasert metode gav nøyaktigere observasjonsdata, men hadde også mindre bias i enkelte manøverpartier. Årsaken til avviket var at posisjonsoppdateringen ikke tok hensyn til endring i avstand, R. Bias ble dermed redusert ved å legge inn variabel R i forsterkningen av posisjonsoppdateringen.

I en videreføring av denne oppgaven kunne det være interessant å implementere avstanden som en del av selve posisjonsoppdateringen eller å se på helt nye metoder. Dette for redusere støy i form av bias da det er avgjørende for estimatorenes ytelse.

Observasjonsstøy

Tre metoder ble i kapittel 4.3 lagt frem for å tilnærme observasjonsstøy. NOAH's og vinkelbasert metode hadde ulike avvikskarakteristikker men allikevel relativt lik tilnærming til virkelig målestøy i gjennomsnitt. Bartons metode gav vesentlig større avvik fra virkelig målestøy enn NOAH's og vinkelstøybasert metode og ble dermed valgt bort.

Årsaken til avviket er at Bartons metode ikke tar tilstrekkelig hensyn til forholdet mellom SNR i målepunktene og heller ikke er utledet for vinkelbasert metode i oppdatering av observasjonsdata.

Estimator

For å glatte observasjonsdataene ble to estimatorer som representerer ulike former for målfølging presentert. EKF filteret ble realisert med en manøverdeteksjonsalgoritme og IMM filteret ble realisert med en hastighets- og akselerasjonsmodell for å takle manøvre.

IMM filteret gav mindre absoluttfeil og kan dermed sies å gi best total ytelse. Ut i fra standardfeilen viste det seg at posisjonsnøyaktigheten var nokså jevnt fordelt både for EKF og IMM filteret. Størrelsesforskjellen i absoluttfeil og standardfeil ved IMM filter og EKF kan knyttes opp mot hvordan de håndterer manøvre. Manøverdeteksjonsalgoritmen i EKF er basert på å øke usikkerheten i målingen mens IMM filteret baserer sine estimater på en egen akselerasjonsmodell. Dette medfører at EKF blir mindre nøyaktig.

Variabel observasjonsstøy basert på SNR i målepunktene reduserte absoluttfeil for samtlige målbaner både med EKF og IMM filter. Spesielt var dette nyttig ved manøvrerende miljø der reduksjonen i absoluttfeil var størst.

EKF og IMM filteret viste seg å være konsistente ved rettlinjede målbevegelser. Ved harde manøvre var EKF også konsistent, mens IMM filteret viste seg å være optimistisk. Dette kan tyde på at akselerasjonsmodellen i filteret er dårlig tilpasset og dermed må utbedres. Det kunne også være interessant å innføre en tredje modell, IMM3, i IMM filteret med annen kovariansmatrise for prosesstøy i en videreføring av denne oppgaven for å oppnå konsistens.

Singularitetsproblemene som oppstod ved harde manøvre ble løst ved å øke diagonalelementene i prosesstøyens kovariansmatrise, Q_k, i IMM filterets akselerasjonsmodell. Siden KF-forsterkningen er sterkt avhengig av sammensetning av prosesstøyens og målestøyens kovariansmatriser ble også den variable støyen generert med større forsterkning. Dette kan være en grunn til at akselerasjonsmodellen blir optimistisk i konsistenstesten.

Generelt for KF modeller er det ved praktisk implementering at mange av filterets egenskaper kommer frem. Hvis vinkeldeteksjons- og manøverdeteksjonsalgoritmer legges til kan det bli vanskelig å forutsi filterets nøyaktighet og ytelse. Det er dermed ofte en krevende oppgave å stille inn filteret med riktig matriser for prosesstøy og målestøy, og med riktige forsterkninger og grenseverdier.

Videreføring av oppgave

Både EKF og IMM filteret bygger på lineære modeller av KF og dermed tilnærmes systemene ved å linearisere rundt nye arbeidspunkt. Siden de lineariserte matrisene kan gi en unøyaktig tilnærming kunne det være interessant å gå videre med ulineær Lyapunov analyse på differensialligningene til systemet. Altså systemligningene og Riccati ligningene. Ved å benytte en slik fremgangsmåte ville forsterkningen, K, til oppdateringen av aposteriori estimatet være den eneste dynamiske oppdateringen i systemet.

Et problem med systemmodellen i denne oppgaven er at det foretas en konvertering fra kartesiske til polare koordinater i målelikningen. Dette medfører at avvik i en dimensjon vil forplante seg i alle dimensjoner ved koordinatkonvertering. En måte å omgå dette problemet er å basere hele systemmodellen på målingens koordinatsystem, polare koordinater. Da vil forplantning av avvik i koordinatkonvertering unngås. I [13, Fossen, T. I. & Strand, J. P.] er et ulineært dynamisk posisjoneringssystem realisert med polare koordinater. Denne artikkelen kan brukes som utgangspunkt dersom det er ønskelig å realisere en slik modell i en videreføring av denne oppgaven.

Bibliografi

[1] Strand T. E. (2001). *Data Fusjon I Distribuert Luftvern System*. Masteroppgave. Institutt for Teknisk kybernetikk. Trondheim: NTNU.

[2] Thales Raytheon Systems. (Dato utilgjengelig). *AN/MPQ 64 Sentinel*. Hentet Februar 15, 2008 fra http://www.thalesraytheon.com/us-anmpq64.htm.

[3] Skolnik, M. I (2001). *Introduction to Radar Systems*. International Edition: McGraw-Hill.

[4] Stimson, G. W. (1998). *Introduction to Airbourne Radar*. USA: ScyTech Publishing.

[5] Fagerlund, S. (2007). *Beskrivelse av oppgaven Radar Tracking.* Kongsberg: Kongsberg Defence and Aerospace AS.

[6] Mahafza, B. R. & Elsherbeni, A. Z. (2001). *MATLAB Simulations for Radar Systems Design.* USA: Chapman & Hall

[7] Barton, D. K. (1988). *Modern radar system analysis*. USA: Artech House Inc.

[8] Ground System Group. (1987). *NOAH Tracking filter error analysis.* California: Hughes Aircraft Company.

[9] Brown, R. G. & Hwang P. Y. (2001). *Introduction to Random Signals and Applied Kalman Filtering.* USA: John Wiley & Sons, Inc.

[10] Bar-Shalom, Y., Li, X. R., & Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation.* USA: John Wiley & Sons, Inc.

[11] Newman, P. & Leonard, J. (2002). *A Matrix Oriented Note on Joint, Marginal, and Conditional Multivariate Gaussian Distributions*, Massachusetts Institute of Technology.

[12] Haugen, F. (2001). *Regulering av dynamiske systemer*. Trondheim: Tapir Akademisk forlag.

[13] Fossen, T. I. & Strand, J. P. (2000). *Nonlinear passive weather optimal positioning control (WOPC) systems for ships and rigs: experimental results*. Trondheim: Elsevier Science Ltd.

Appendiks

A. Utledninger, beregninger og resultater

A.1 Utledning av Bartons metode

Barton har utledet følgende modell for generering av termisk målestøy.

$$\sigma_\theta = \frac{\theta_3}{k_m\sqrt{2 \cdot SNR \cdot n}} \quad [rad]$$

Der σ_θ er standardavviket som beskriver presisjonen i posisjonsoppdateringen. Antall målte pulser er gitt ved n og variabelen k_m beskriver gradienten til differansestrålen i antennekonfigurasjonen.

$$k_m = -\frac{d(\Delta/\Sigma)}{d(\theta/\theta_3)}\bigg|_{\theta=\theta_0}$$

Δ og Σ er de samme som i ligning 4-3 til 4-5 og θ_3 er båndbredden ved -3 dB for funksjonen $\Sigma(\theta)$. k_m kan tilnærmes til 1.6.

$$\sigma_\theta = \frac{\theta_3}{1.6\sqrt{2 \cdot SNR \cdot n}}$$

Dette uttrykket kan benyttes for å finne σ_{Az} og σ_{El} ved å dele inn i strålesektorer for asimut og elevasjon. n settes lik 1 siden det kun måles på én puls.

$$\sigma_{Az} = \frac{\theta_{Az,3}}{1.6\sqrt{2 \cdot SNR_{Az}}} \quad [rad]$$

$$\sigma_{El} = \frac{\theta_{El,3}}{1.6\sqrt{2 \cdot SNR_{El}}} \quad [rad]$$

A.2 Utledning av NOAH's metode

For NOAH er det utledet følgende modell for generering av termisk målestøy.

Vinkelavviket er gitt ved:

$$\beta = \frac{F \cdot \theta}{5.55} ln\left(\frac{V_2}{V_1}\right) + \varepsilon$$

Da er den tidsderiverte av vinkelavviket gitt ved:

$$\dot{\beta} = \frac{F \cdot \theta}{5.55}\left[\frac{\dot{V}_2}{V_2} - \frac{\dot{V}_1}{V_1}\right]$$

Ved å anta et gjennomsnitt lik 0 kan vinkelavvikets forventningsverdi kvadreres slik:

$$E\left[\dot{\beta}^2\right] = \frac{F \cdot \theta}{5.55}\left[E\left(\frac{\dot{V}_2}{V_2}\right)^2 - E\left(\frac{\dot{V}_1}{V_1}\right)^2\right]$$

I dette uttrykket inngår:

$$\frac{E\left(\dot{V}_1\right)^2}{{V_1}^2} = \frac{{\sigma_1}^2}{{V_1}^2} = \left(\frac{N}{S_1}\right)_{MAX} e^{5.55\left(\frac{\Delta + 2\beta}{2\theta}\right)^2}$$

Hvor SNR kan tilnærmes ved:

$$\left(\frac{S_1}{N}\right)_{MAX} = 2\left(\frac{S_1}{N}\right)_{AVE} = 2\left(\frac{\bar{S}}{\bar{N}}\right)$$

Med $\Delta = \theta / F$ kan ligningen skrives om:

$$\frac{{\sigma_1}^2}{{V_1}^2} = \frac{1}{2\bar{S}/\bar{N}} e^{5.55\left(\frac{1}{2F}+\frac{\beta}{\theta}\right)^2}$$

I likhet med:

$$\frac{{\sigma_2}^2}{{V_2}^2} = \frac{1}{2\bar{S}/\bar{N}} e^{5.55\left(\frac{1}{2F}-\frac{\beta}{\theta}\right)^2}$$

Ved å sette inn for disse ligningene fås:

$$E\left[\dot{\beta}^2\right] = \frac{F \cdot \theta}{5.55}\left[\frac{1}{2\bar{S}/\bar{N}} e^{5.55\left(\frac{1}{2F}-\frac{\beta}{\theta}\right)^2} - \frac{1}{2\bar{S}/\bar{N}} e^{5.55\left(\frac{1}{2F}+\frac{\beta}{\theta}\right)^2}\right]$$

Vinkelavvikets standardavvik kan dermed finnes ved å ta kvadratroten til denne ligningen:

$$\sigma_\beta = E\left[\dot{\beta}\right] = \frac{F \cdot \theta}{5.55\sqrt{2\bar{S}/\bar{N}}}\left[e^{5.55\left(\frac{1}{2F}-\frac{\beta}{\theta}\right)^2} - e^{5.55\left(\frac{1}{2F}+\frac{\beta}{\theta}\right)^2}\right]^{\frac{1}{2}}$$

Ved å tilpasse dette SNR data og riktig forsterkning fås følgende ligninger:

$$\sigma_{Az} = E\left[\dot{\beta}_{Az}\right] = K_{\sigma_{Az}} \cdot \frac{F \cdot AzBw}{5.55\sqrt{2 \cdot SNR_{Az}}}\left[e^{5.55\left(\frac{1}{2F}-\frac{\beta_{Az}}{AzBw}\right)^2} - e^{5.55\left(\frac{1}{2F}+\frac{\beta_{Az}}{AzBw}\right)^2}\right]^{\frac{1}{2}}$$

$$\sigma_{El} = E\left[\dot{\beta}_{El}\right] = K_{\sigma_{El}} \cdot \frac{F \cdot ElBw}{5.55\sqrt{2 \cdot SNR_{El}}}\left[e^{5.55\left(\frac{1}{2F}-\frac{\beta_{El}}{ElBw}\right)^2} - e^{5.55\left(\frac{1}{2F}+\frac{\beta_{El}}{ElBw}\right)^2}\right]^{\frac{1}{2}}$$

Der:

$$\beta_{Az} = \frac{F \cdot AzBw}{5.55} ln\left(\frac{V_1}{V_4}\right) \approx \frac{F \cdot AzBw}{5.55} ln\left(\frac{SNR_1}{SNR_4}\right)$$

$$\beta_{El} = \frac{F \cdot ElBw}{5.55} ln\left(\frac{SNR_2}{SNR_3}\right)$$

A.3 Utledning av vinkelstøybasert metode

Utledning av observasjonsstøyen som standardavvik gjøres ved først å beregne den tidsderiverte av vinkeloppdateringen, ΔAz og ΔEl. Vinkeloppdateringen er fra vinkelbasert metode for oppdatering av observasjonsdata.

$$Az_{k+1} = Az_k + \underbrace{K_{Az} \cdot \left[SNR_{db,4} - SNR_{db,1}\right]}_{=Az_\varepsilon}$$

$$El_{k+1} = El_k + \underbrace{\frac{K_{El} \cdot \left[SNR_{db,2} - SNR_{db,3}\right] + \left[Az_{k+1} - Az_k\right] \cdot \cos\gamma}{\sin\gamma}}_{El_\varepsilon}$$

$$\frac{d(Az_\varepsilon)}{dt} = \frac{\partial(Az_\varepsilon)}{\partial SNR_1} \cdot \frac{\partial SNR_1}{\partial t} + \frac{\partial(Az_\varepsilon)}{\partial SNR_4} \cdot \frac{\partial SNR_4}{\partial t} = K_{Az} \cdot (\dot{SNR}_4 - \dot{SNR}_1)$$

$$\frac{d(El_\varepsilon)}{dt} = \frac{K_{El} \cdot (\dot{SNR}_2 - \dot{SNR}_3)}{\sin\gamma}$$

Videre finnes standardavviket ved å ta forventningsverdien opphøyd i andre.

$$\sigma_{\Delta Az}{}^2 = E\left[\dot{Az}_\varepsilon{}^2\right] = E\left[K_{Az}{}^2 \cdot (\dot{SNR}_4 - \dot{SNR}_1)^2\right]$$
$$= E\left[K_{Az}{}^2 \cdot (\dot{SNR}_4{}^2 + \dot{SNR}_1{}^2 - 2 \cdot \dot{SNR}_4 \dot{SNR}_1)\right]$$
$$= K_{Az}{}^2 \cdot E\left[\dot{SNR}_4{}^2\right] + E\left[\dot{SNR}_1{}^2\right]$$

$$\sigma_{\Delta El}{}^2 = E\left[\dot{El}_\varepsilon{}^2\right] = \frac{K_{El}{}^2 \cdot E\left[\dot{SNR}_2{}^2\right] + E\left[\dot{SNR}_3{}^2\right]}{\sin\gamma}$$

$E\left[\dot{SNR}^2\right]$ kan settes på formen nedenfor:

$$\sigma_n{}^2 = E\left[\dot{SNR}_n{}^2\right] = \frac{1}{SNR_{n,Maks}} = \frac{1}{2 \cdot (SNR_{n,Gj.snitt})} = \frac{1}{2 \cdot SNR_n}, \quad n = 1, \ldots, 4$$

Observasjonsstøyen er da gitt ved:

$$\sigma_{Az} = \left| \frac{K_{Az}}{\sqrt{2}} \cdot \sqrt{\frac{1}{SNR_4} - \frac{1}{SNR_1}} \right|$$

$$\sigma_{El} = \left| \frac{K_{El}}{\sqrt{2}} \cdot \sqrt{\frac{SNR_3 - SNR_2}{SNR_3 \cdot SNR_2 \cdot \sin\gamma}} \right|$$

A.4 Uavhengighet av avstand, R

Monopulsbasert metode

Ved å sette ligning 4-1 inn i 4-3 til 4-5 fås følgende ligningssett:

$$\delta_{Az} = \frac{\mathrm{SNR_{målt,4}} - \mathrm{SNR_{målt,1}} + (\mathrm{SNR_{målt,3}} - \mathrm{SNR_{målt,2}}) \cdot \sin(\gamma)}{\mathrm{SNR_{målt,1}} + \mathrm{SNR_{målt,4}} + \left(\mathrm{SNR_{målt2}} + \mathrm{SNR_{målt,3}}\right) \cdot \sin(\gamma)} + \frac{2 \cdot (\mathrm{SNR_{målt,3}} - \mathrm{SNR_{målt,2}}) \cdot \cos(\gamma)}{\mathrm{SNR_{målt,1}} + \mathrm{SNR_{målt,4}} + \left(\mathrm{SNR_{målt2}} + \mathrm{SNR_{målt,3}}\right) \cdot \sin(\gamma)} = \frac{\mathrm{L_4}^{-1} - \mathrm{L_1}^{-1} + \left(\mathrm{L_3}^{-1} - \mathrm{L_2}^{-1}\right) \cdot \sin(\gamma) + 2 \cdot (\mathrm{L_3}^{-1} - \mathrm{L_2}^{-1}) \cdot \cos(\gamma)}{\mathrm{L_1}^{-1} + \mathrm{L_4}^{-1} + \left(\mathrm{L_2}^{-1} + \mathrm{L_3}^{-1}\right) \cdot \sin(\gamma)}$$

$$\delta_{El} = \frac{\left(\mathrm{SNR_{målt2}} - \mathrm{SNR_{målt,3}}\right) \cdot (\sin(\gamma) - 2 \cdot \cos(\gamma)) + \mathrm{SNR_{målt,4}} - \mathrm{SNR_{målt,1}}}{\mathrm{SNR_{målt,1}} + \mathrm{SNR_{målt,4}} + \left(\mathrm{SNR_{målt2}} + \mathrm{SNR_{målt,3}}\right) \cdot \sin(\gamma)} = \frac{\left(\mathrm{L_2}^{-1} - \mathrm{L_3}^{-1}\right) \cdot \left(\sin(\gamma) - 2 \cdot \cos(\gamma)\right) + \mathrm{L_4}^{-1} - \mathrm{L_1}^{-1}}{\mathrm{L_1}^{-1} + \mathrm{L_4}^{-1} + \left(\mathrm{L_2}^{-1} + \mathrm{L_3}^{-1}\right) \cdot \sin(\gamma)}$$

Siden RCS holdes konstant og avstanden, R, er lik i alle de fire målepunktene i denne oppgaven kan disse settes tilnærmet like.

$$R_1 \approx R_2, R_3, R_4, \quad RCS_1 \approx RCS_2, RCS_3, RCS_4$$

$$\begin{bmatrix} Az_\varepsilon \\ El_\varepsilon \end{bmatrix} = R_{\frac{\pi}{4}} \begin{bmatrix} Az_\delta \\ El_\delta \end{bmatrix}$$

$$Az_\delta = \delta_{Az} \cdot {}^{\Delta\theta}/_{2}$$

$$El_\delta = \delta_{El} \cdot {}^{\Delta\theta}/_{2}$$

$$Az_{k+1} = Az_k + K_{Az} \cdot Az_\varepsilon$$

$$El_{k+1} = El_k + K_{El} \cdot El_\varepsilon$$

Avstanden, R, RCS og konstanten K_{SNR} lar seg dermed stryke og vinkeloppdateringen for monopulsbasert metode er da uavhengig av disse variablene.

Vinkelbasert metode

Ved å sette ligning 4-1 inn i 4-23 fås følgende ligningssett:

$$Az_{k+1} = Az_k + K_{Az} \cdot \left[10 \cdot log_{10}\left(\frac{K_{SNR} \cdot RCS_4}{{R_4}^4 \cdot L_4}\right) - 10 \cdot log_{10}\left(\frac{K_{SNR} \cdot RCS_1}{{R_1}^4 \cdot L_1}\right)\right]$$

$$= Az_k + K_{Az} \cdot \left[10 \cdot log_{10}\left(\frac{\left(\frac{K_{SNR} \cdot RCS_4}{{R_4}^4 \cdot L_4}\right)}{\frac{K_{SNR} \cdot RCS_1}{{R_1}^4 \cdot L_1}}\right)\right], \quad R_4 \approx R_1, \quad RCS_1 \approx RCS_4$$

$$= Az_k + K_{Az} \cdot \left[10 \cdot log_{10}\left(\frac{L_1}{L_4}\right)\right]$$

$$= Az_k + 10 \cdot K_{Az} \cdot \left[log_{10}\left(10^{1.2\left[\left(\frac{\Delta Az_1}{AzBw}\right)^2 + \left(\frac{\Delta El_1}{ElBw}\right)^2\right]}\right) - log_{10}\left(10^{1.2\left[\left(\frac{\Delta Az_4}{AzBw}\right)^2 + \left(\frac{\Delta El_4}{ElBw}\right)^2\right]}\right)\right]$$

$$= Az_k + K_{Az} \cdot \left[12 \cdot \left(\left[\left(\frac{\Delta Az_1}{AzBw}\right)^2 + \left(\frac{\Delta El_1}{ElBw}\right)^2\right] - \left[\left(\frac{\Delta Az_4}{AzBw}\right)^2 + \left(\frac{\Delta El_4}{ElBw}\right)^2\right]\right)\right]$$

$$= Az_k + 12 \cdot K_{Az} \cdot \left[\frac{1}{AzBw^2}\left({\Delta Az_1}^2 - {\Delta Az_4}^2\right) + \frac{1}{ElBw^2}\left({\Delta El_1}^2 - {\Delta El_4}^2\right)\right]$$

Her er $\Delta El_1 = \Delta El_4$ fordi feilen i elevasjonsvinkel er lik både for målepunkt 1 og 4.

$$Az_{k+1} = Az_k + 12 \cdot K_{Az} \cdot \left[\frac{1}{AzBw^2}\left({\Delta Az_1}^2 - {\Delta Az_4}^2\right)\right]$$

Av dette fremgår det at avstanden, R, RCS, K_{SNR} og ΔEl lar seg stryke og vinkeloppdateringen i asimut for vinkelbasert metode er dermed uavhengig av disse variablene.

Ved å sette ligning 4-1 inn i 4-24 fås følgende ligningssett:

$$El_{k+1} = El_k + \frac{K_{El} \cdot \left[10 \cdot log_{10}\left(\frac{K_{SNR} \cdot RCS_2}{{R_2}^4 \cdot L_2}\right) - 10 \cdot log_{10}\left(\frac{K_{SNR} \cdot RCS_3}{{R_3}^4 \cdot L_3}\right)\right]}{\sin\gamma} + \frac{[Az_{k+1} - Az_k] \cdot \cos\gamma}{\sin\gamma}$$

$$= El_k + \frac{K_{El} \cdot \left[10 \cdot log_{10}\left(\frac{\left(\frac{K_{SNR} \cdot RCS_2}{{R_2}^4 \cdot L_2}\right)}{\frac{K_{SNR} \cdot RCS_3}{{R_3}^4 \cdot L_3}}\right)\right]}{\sin\gamma} + \frac{[Az_{k+1} - Az_k] \cdot \cos\gamma}{\sin\gamma}, \quad R_2 \approx R_3, \quad RCS_2 \approx RCS_3$$

$$= El_k + \frac{K_{El} \cdot \left[10 \cdot log_{10}\left(\frac{L_3}{L_2}\right)\right]}{\sin\gamma} + \frac{[Az_{k+1} - Az_k] \cdot \cos\gamma}{\sin\gamma}$$

$$= El_k + \frac{10 \cdot K_{El} \cdot \left[log_{10}\left(10^{1.2\left[\left(\frac{\Delta Az_3}{AzBw}\right)^2 + \left(\frac{\Delta El_3}{ElBw}\right)^2\right]}\right) - log_{10}\left(10^{1.2\left[\left(\frac{\Delta Az_2}{AzBw}\right)^2 + \left(\frac{\Delta El_2}{ElBw}\right)^2\right]}\right)\right]}{\sin\gamma} + \frac{[Az_{k+1} - Az_k] \cdot \cos\gamma}{\sin\gamma}$$

$$= El_k + \frac{K_{El} \cdot \left[12 \cdot \left(\left[\left(\frac{\Delta Az_3}{AzBw}\right)^2 + \left(\frac{\Delta El_3}{ElBw}\right)^2\right] - \left[\left(\frac{\Delta Az_2}{AzBw}\right)^2 + \left(\frac{\Delta El_2}{ElBw}\right)^2\right]\right)\right]}{\sin\gamma} + \frac{[Az_{k+1} - Az_k] \cdot \cos\gamma}{\sin\gamma}$$

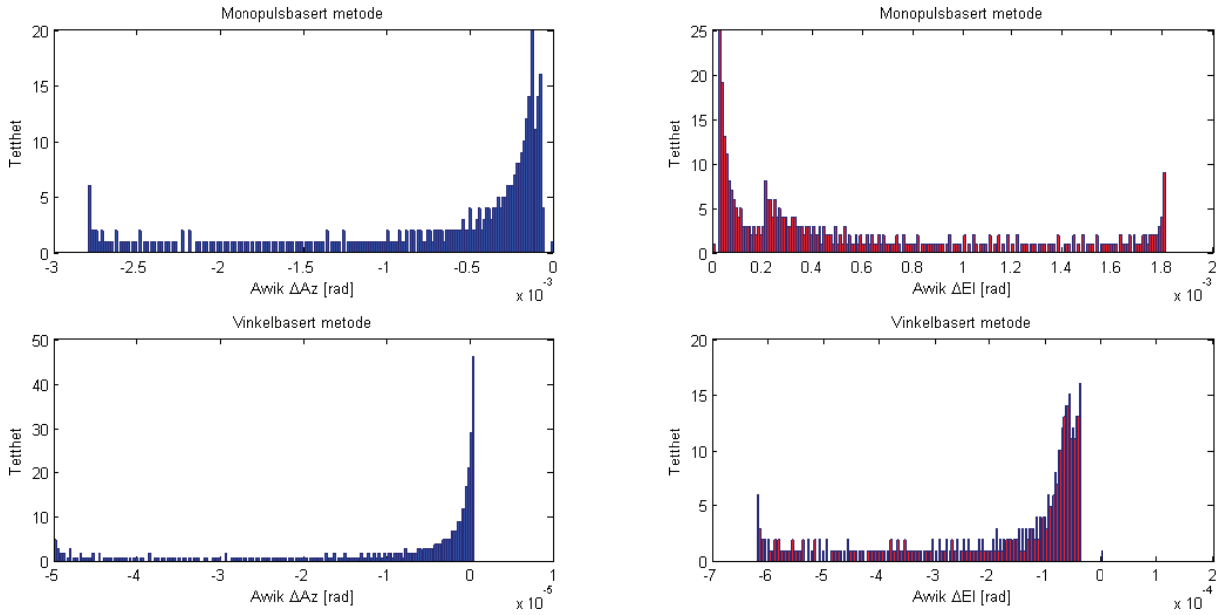
$$= El_k + \frac{12 \cdot K_{El} \cdot \left[\frac{1}{AzBw^2}\left(\Delta {Az_3}^2 - \Delta {Az_2}^2\right) + \frac{1}{ElBw^2}\left(\Delta {El_3}^2 - \Delta {El_2}^2\right)\right]}{\sin\gamma} + \frac{[Az_{k+1} - Az_k] \cdot \cos\gamma}{\sin\gamma}$$

Av dette fremgår det at avstanden, R, RCS og konstanten K_{SNR} også lar stryke og vinkeloppdateringen i elevasjon for vinkelbasert metode er dermed også uavhengig av disse variablene.

A.5 Observasjonsdata fra målbane 1 til 3

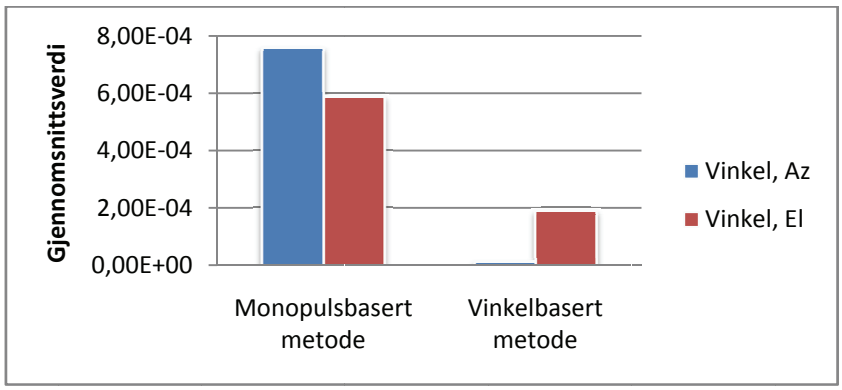
Målbane 1

Histogrammet i figur A-1 viser tettheten til avviket i oppdaterte observasjonsdata for monopulsbasert og vinkelbasert metode med målbane 1.



Figur A-1: Histogram av oppdaterte observasjonsdata for målbane 1

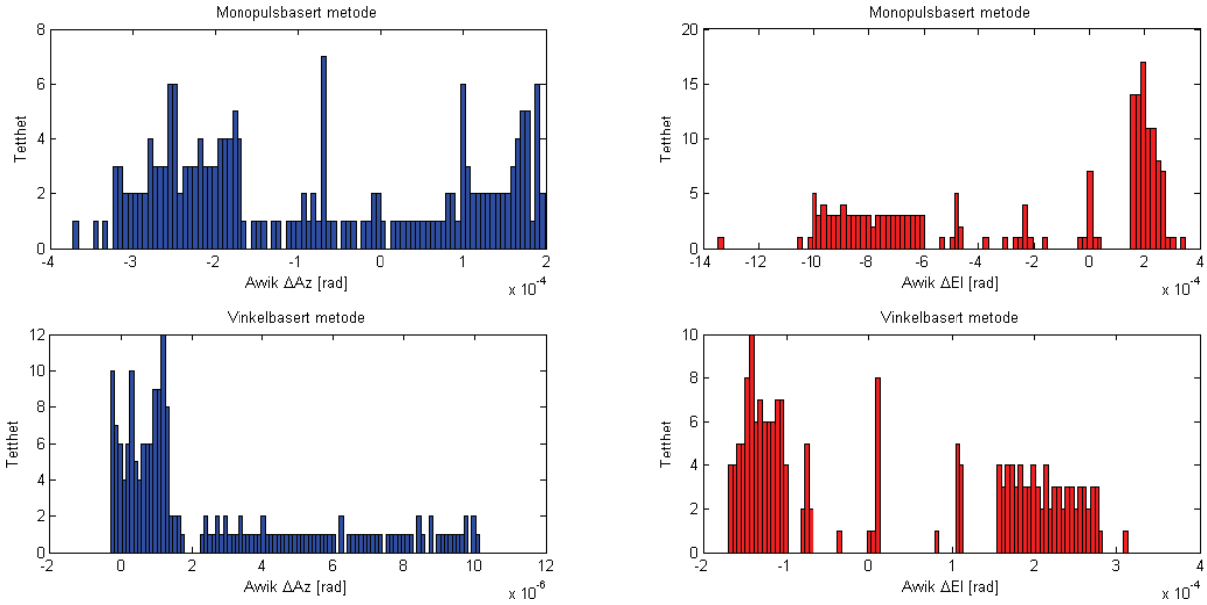
Figur A-2 viser gjennomsnittlige vinkelavvik i målbane 1 med monopulsbasert og vinkelbasert metode for oppdaterte observasjonsdata



Figur A-2: Gjennomsnittlig vinkelavvik i målbane 1

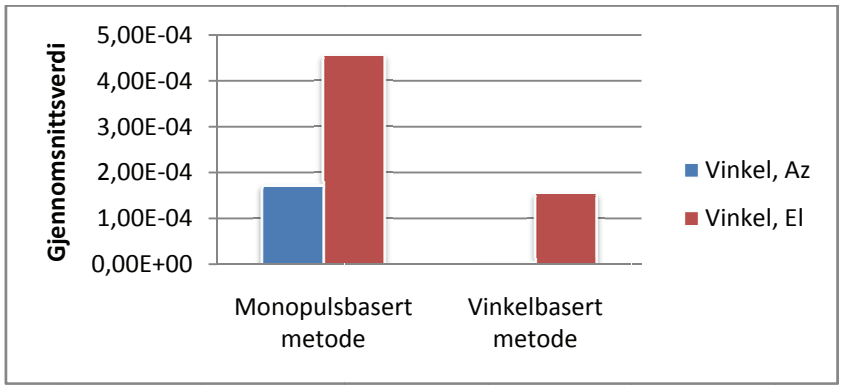
Målbane 2

Histogrammet i figur A-3 viser tettheten til avviket i oppdaterte observasjonsdata for monopulsbasert og vinkelbasert metode med målbane 2.



Figur A-3: Histogram av oppdaterte observasjonsdata for målbane 2

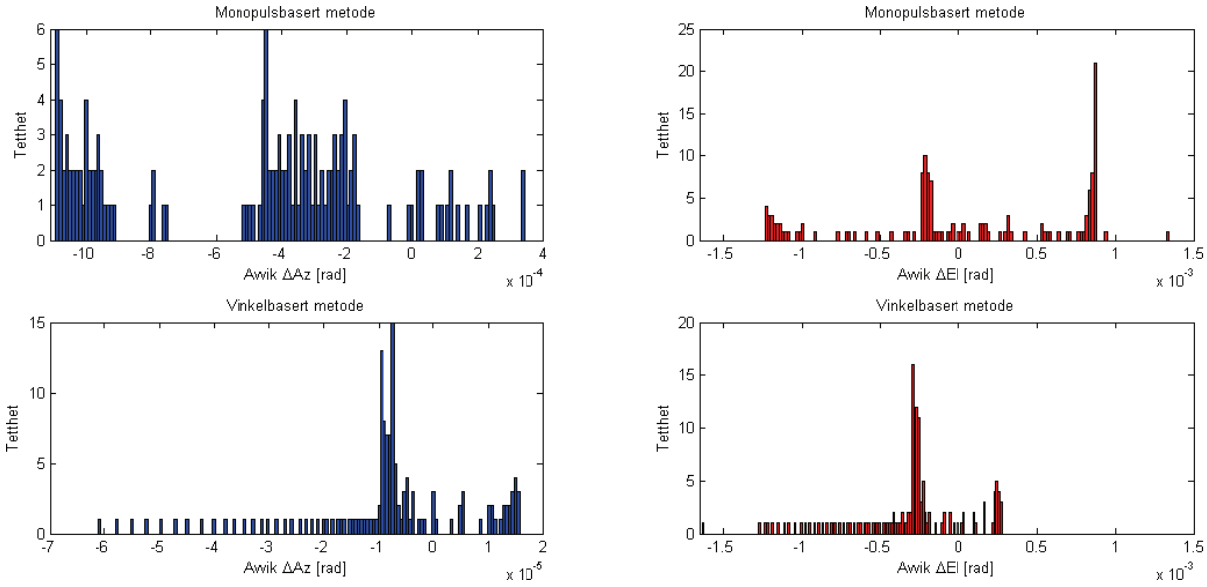
Figur A-4 viser gjennomsnittlige vinkelavvik i målbane 2 med monopulsbasert og vinkelbasert metode for oppdaterte observasjonsdata



Figur A-4: Gjennomsnittlig vinkelavvik i målbane 2

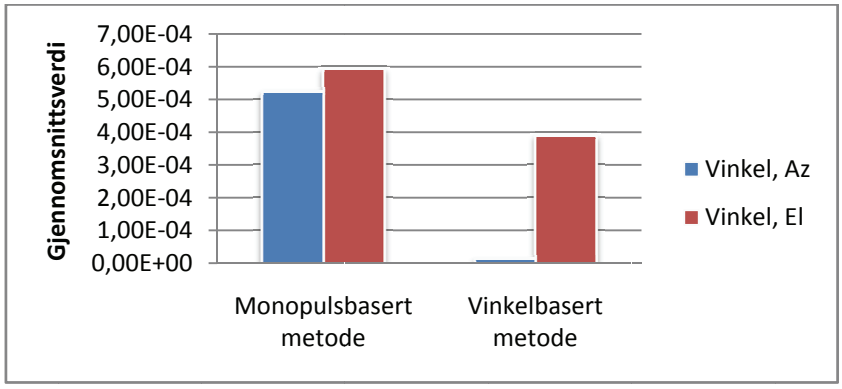
Målbane 3

Histogrammet i figur A-5 viser tettheten til avviket i oppdaterte observasjonsdata for monopulsbasert og vinkelbasert metode med målbane 3.



Figur A-5: Histogram av oppdaterte observasjonsdata for målbane 3

Figur A-6 gjennomsnittlige vinkelavvik i målbane 3 med monopulsbasert og vinkelbasert metode for oppdaterte observasjonsdata



Figur A-6: Gjennomsnittlig vinkelavvik i målbane 3

A.6 Konstanter ved oppstart av EKF og IMM filteret

Estimeringsalgoritmene som blir presentert i dette kapitlet vil simuleres med følgende konstanter.

Prosesstøyens kovariansmatrise blir tilnærmet ved simulering.

$$Q_{KF} = 0{,}95 \cdot diag([0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1])$$

IMM filterets modeller vil ha forskjellig vekting av prosesstøyens kovariansmatriser i hastighetsmodellen og akselerasjonsmodellen.

$$Q_{IMM}^{1} = 0{,}95 \cdot diag([0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0])$$

$$Q_{IMM}^{2} = 20 \cdot diag([0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1])$$

Siden det i denne oppgaven ikke blir tatt hensyn til pådragsfaktorer vil pådragsvektoren både i KF og IMM filterets modeller settes lik 0.

$$u_k^{IMM} = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T$$

$$u_k^{KF} = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T$$

Som tidligere nevnt holdes målestøyens kovarians konstant i de første estimatorsimuleringene og blir implementert variabelt med algoritmene for observasjonsstøy senere. Siden EKF og IMM filteret bruker samme inngangsdata vil konstantene være like og gitt etter den variansen det forventes at målestøyen vil ha.

$$\sigma_{KF} = \sigma_{IMM}^{1} = \sigma_{IMM}^{2} = diag([25 \quad 0.001 \quad 0.001])$$

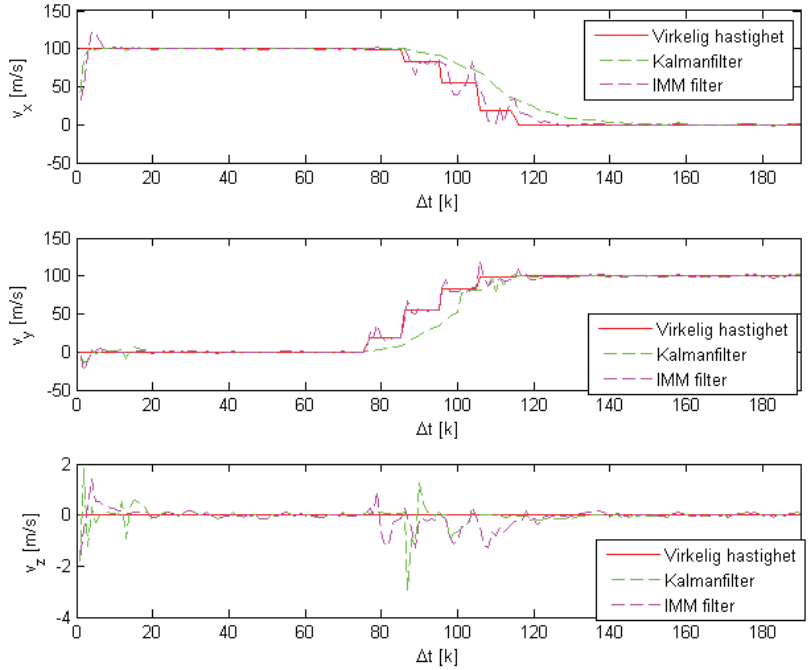
Selv om målbanene i forkant her er kjent vil de i virkeligheten ikke være det. For å gjøre systemets reaksjon så virkelighetsnært som mulig settes dermed modellsannsynligheten til modellene i IMM filteret lik med 50 % sannsynlighet.

$$\mu_0^1 = \mu_0^2 = 0{,}5$$

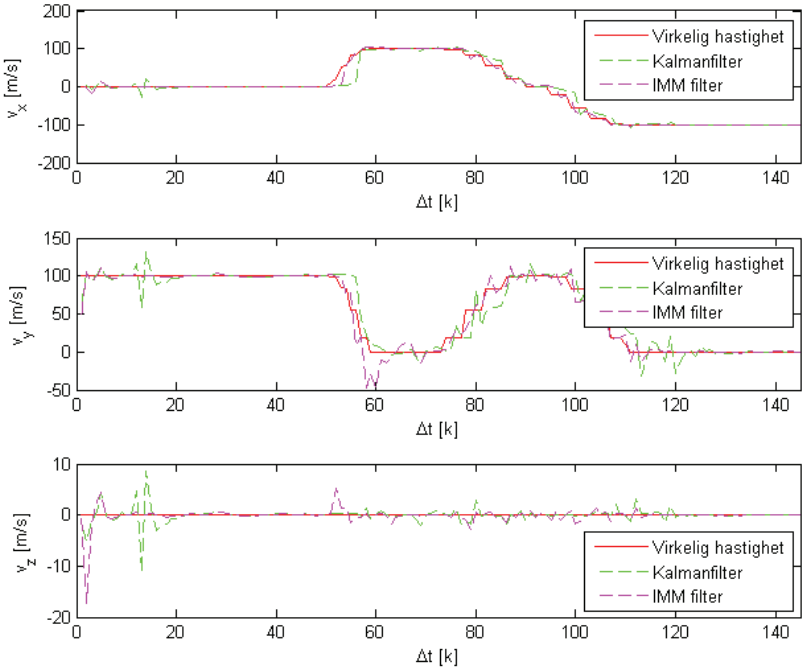
Markovkjedens transisjonsmatrise vektes hardt slik at det må større manøvre til før modelltransisjonen går fra hastighetsmodellen til akselerasjonsmodellen eller motsatt.

$$p_k = \begin{bmatrix} 0.999 & 0.001 \\ 0.001 & 0.999 \end{bmatrix}$$

A.7 Estimat av hastighet for målbane 2 og 3

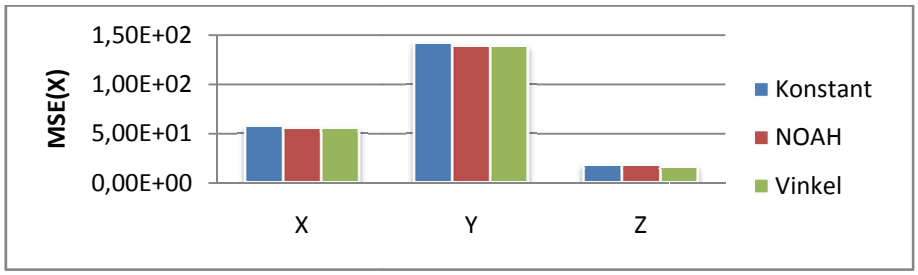


A-7: Hastighetsestimering ved målbane 4

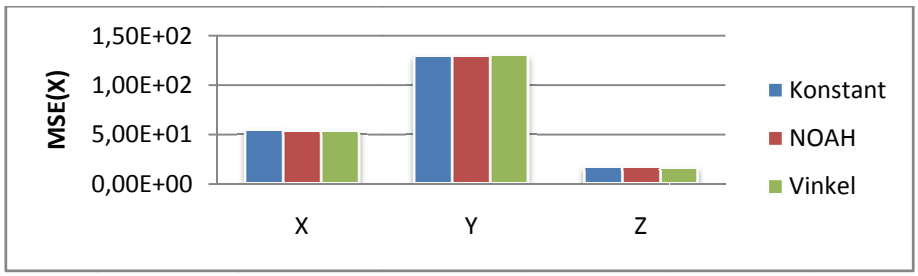


A-8: Hastighetsestimering ved målbane 3

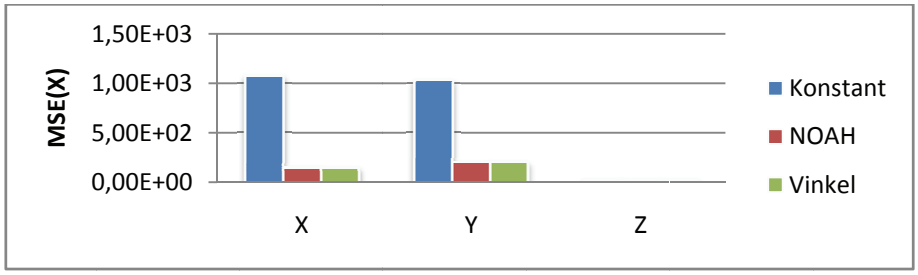
A.8 Middelkvadratsfeil ved variabel observasjonsstøy



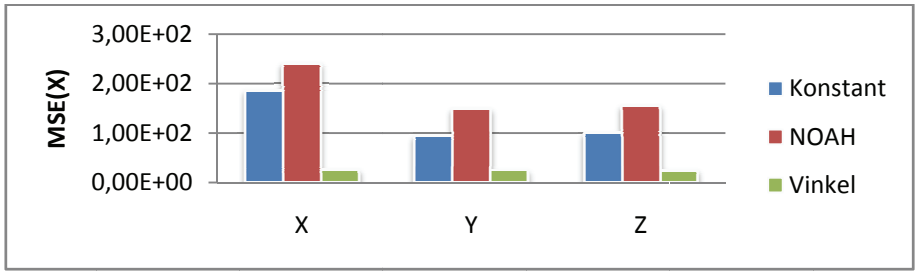
Figur A-9: Middelkvadratsfeil med EKF i målbane 1



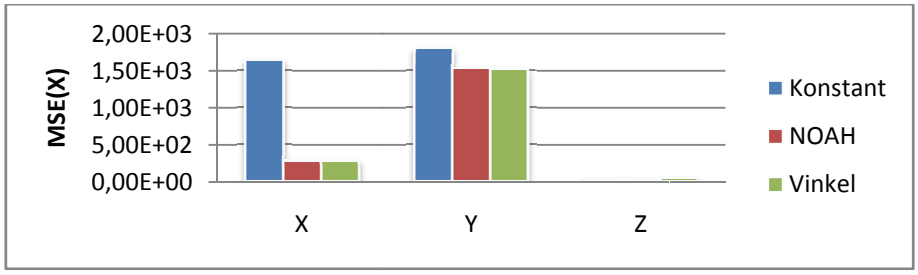
Figur A-10: Middelkvadratsfeil med IMM filter i målbane 1



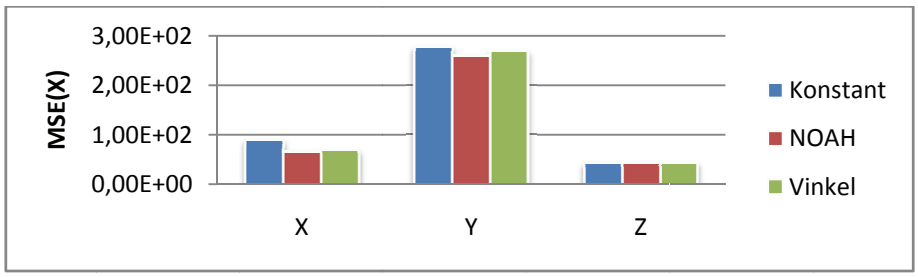
Figur A-11: Middelkvadratsfeil med EKF i målbane 2



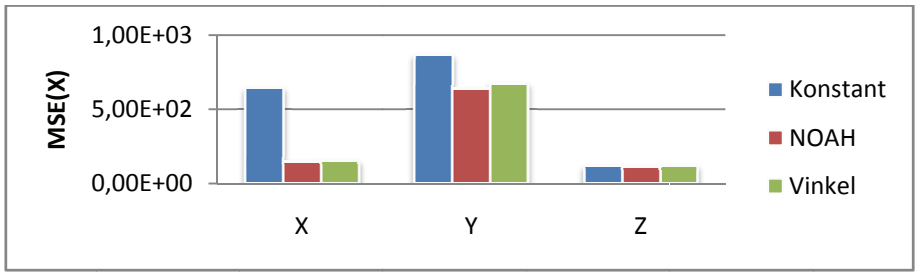
Figur A-12: Middelkvadratsfeil med IMM filter i målbane 2



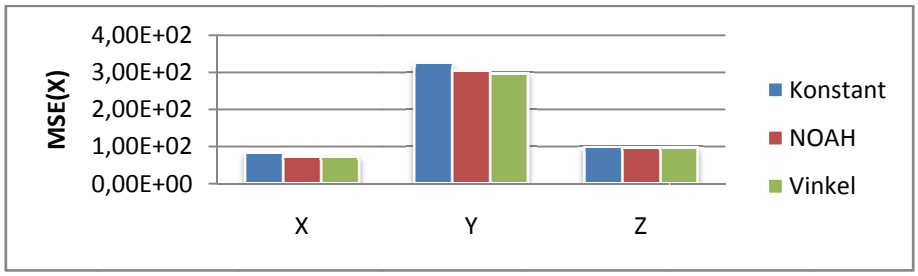
Figur A-13: Middelkvadratsfeil med EKF i målbane 3



Figur A-14: Middelkvadratsfeil med IMM filter i målbane 3

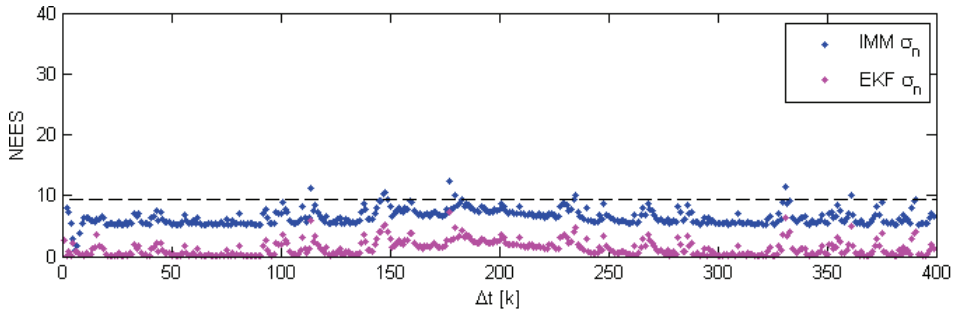


Figur A-15: Middelkvadratsfeil med EKF i målbane 4

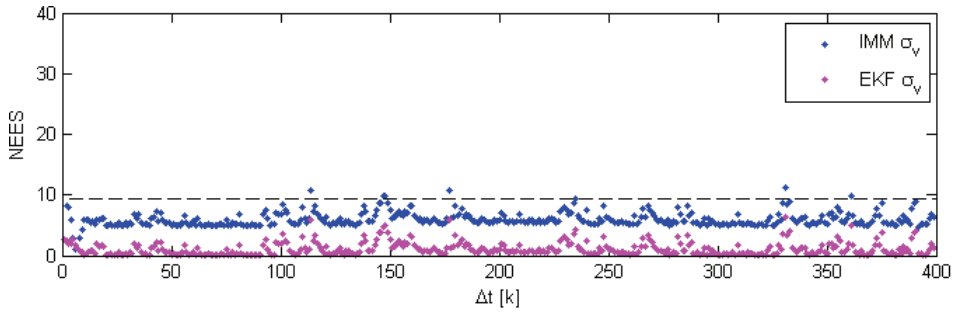


Figur A-16: Middelkvadratsfeil med IMM filter i målbane 4

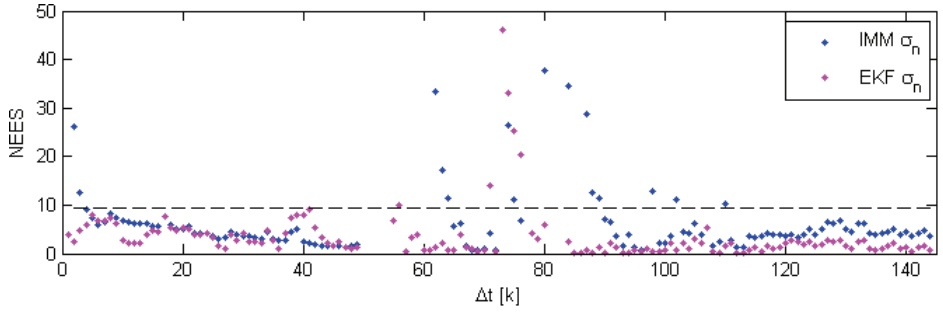
A.9 Konsistenstest ved variabel observasjonsstøy



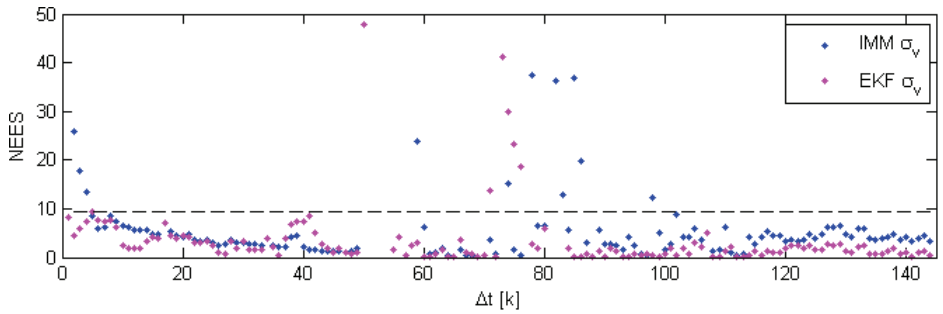
A-17: NEES ved observasjonsstøy fra NOAH's metode i målbane 1



A-18: NEES ved observasjonsstøy fra vinkelstøybasert metode i målbane 1



A-19: NEES ved observasjonsstøy fra NOAH's metode i målbane 4



A-20: NEES ved observasjonsstøy fra vinkelstøybasert metode i målbane 4

B. Programkode

B.1 Hovedfil

main.m

```
function [ output_args ] = main( input_args )

% Jan S. Karlsen
% 26.05.2008
%
% ============== Main file ==============
%
% Description:
% This file constitutes a framework for the simulation of all
% the methods presented in the master thesis
%
% Dependencies:
% - ttraject.m
% - snr.m
% - kalman_init.m
% - imm_init.m
% - observation_data_m.m
% - observation_data_v.m
% - observation_noise_b.m
% - observation_noise_n.m
% - observation_noise_v.m
% - imm.m
% - kalman.m
%

clc;
clear;

% -------------- Target rajectory --------------

x_true = ttraject();
x_true = x_true';
N = length(x_true);

x_tr = x_true(1,:);
y_tr = x_true(2,:);
z_tr = x_true(3,:);

for i=1:N

    R_true(i) = sqrt(x_tr(i)^2 + y_tr(i)^2 + z_tr(i)^2);
    Az_true(i) = atan2(y_tr(i),x_tr(i));
    El_true(i) = atan2(-z_tr(i),sqrt(x_tr(i)^2 + y_tr(i)^2));

end

% True target trajectory in polar coordinates
p_true = [ R_true; Az_true; El_true ];

% Random white noise
R_rand = randn(1,N) * 25;
Az_rand = randn(1,N) * 0.001;
El_rand = randn(1,N) * 0.001;
p_rand = [R_rand; Az_rand; El_rand];

meas = p_true + p_rand;  % Measured position in polar coordinates
```

```
obs_data = p_true(1,:) + p_rand(1,:); % Range for observation data

% Initial target position
pos_update(1) = p_true(2,1) + 1e-6;
pos_update(2) = p_true(3,1) + 1e-6;

% Constant observation noise
sigma_c = [ 25 0.001 0.001 ];


% -------------- Kalman filter initialization --------------

[xhat_rec, P_k_rec] = kalman_init( meas, sigma_c );

% Initial state and covariance for Kalman-filter
x1hat_c = x1hat;
x1hat_n = x1hat;
x1hat_v = x1hat;
x2hat_c = x1hat;
x2hat_n = x1hat;
x2hat_v = x1hat;
P1_k_c = P1_k;
P1_k_n = P1_k;
P1_k_v = P1_k;
P2_k_c = P1_k;
P2_k_n = P1_k;
P2_k_v = P1_k;


% -------------- IMM filter initialization --------------

[x1hat, P1_k] = imm_init( meas, sigma_c );

% Initial model propability
mu = [ 0.5 0.5 ];
mu_c = mu;
mu_n = mu;
mu_v = mu;

% Initial state and covariance for IMM filter
xhat_rec_c = xhat_rec;
xhat_rec_n = xhat_rec;
xhat_rec_v = xhat_rec;
P_k_rec_c = P_k_rec;
P_k_rec_n = P_k_rec;
P_k_rec_v = P_k_rec;

res_prev_c = zeros(3,10);
res_prev_n = res_prev_c;
res_prev_v = res_prev_c;

for i = 1:N

    % -------------- SNR data --------------
```

```
    SNR = snr( p_true(:,i), pos_update );
    SNR_p(:,i) = SNR;


    % -------------- Observation data --------------

    % Monoulse based method
%     [ pos_update, pos_prev ] = observation_data_m( SNR, pos_update, meas(:,i) );

    % Angle based method
    [ pos_update, pos_prev ] = observation_data_v( SNR, pos_update, meas(:,i) );
    obs_data(2:3,i) = pos_update;


    % -------------- Observation noise --------------

    % Barton's method
    [ sigma_R_b, sigma_Az_b, sigma_El_b ] = observation_noise_b( SNR );
    sigma_b = [ sigma_R_b sigma_Az_b sigma_El_b ];

    % NOAH's method
    [ sigma_R_n, sigma_Az_n, sigma_El_n ] = observation_noise_n( SNR );
    sigma_n = [ sigma_R_n sigma_Az_n sigma_El_n ];

    % Angle noise based method
    [ sigma_R_v, sigma_Az_v, sigma_El_v ] = observation_noise_v( SNR );
    sigma_v = [ sigma_R_v sigma_Az_v sigma_El_v ];


    % -------------- Extended Kalman filter --------------

    % EKF with constant observation noise
    [ xhat_rec_c, P_k_rec_c, res_prev_c, res_p, K_k ]...
        = kalman( sigma_c_p, xhat_rec_c, obs_data(:,i), P_k_rec_c, res_prev_c );
    xhat_kalman_c(:,i) = xhat_rec_c;

    % EKF with observation noise from NOAH's method
    [ xhat_rec_n, P_k_rec_n, res_prev_n, res_p, K_k ]...
        = kalman( sigma_n_p, xhat_rec_n, obs_data(:,i), P_k_rec_n, res_prev_n );
    xhat_kalman_n(:,i) = xhat_rec_n;

    % EKF with observation noise from angle noise based method
    [ xhat_rec_v, P_k_rec_v, res_prev_v, res_p, K_k ]...
        = kalman( sigma_v_p, xhat_rec_v, obs_data(:,i), P_k_rec_v, res_prev_v );
    xhat_kalman_v(:,i) = xhat_rec_v;


    % -------------- IMM filter --------------

    % IMM filter with constant observation noise
    [ xhat_c, P_k_c, mu_c, x1hat_c, x2hat_c, P1_k_c, P2_k_c ] =...
        imm(mu_c, x1hat_c, x2hat_c, P1_k_c, P2_k_c, obs_data(:,i), sigma_c_p);
    mu1_c = mu_c(1);
    mu2_c = mu_c(2);

    xhat_c = x1hat_c * mu1_c + x2hat_c * mu2_c;
```

```
    P_k_c = mu1_c * (P1_k_c + (x1hat_c - xhat_c) * (x1hat_c - xhat_c)') +...
        mu2_c * (P2_k_c + (x2hat_c - xhat_c) * (x2hat_c - xhat_c)');
    xhat_imm_c(:,i) = xhat_c;

    % IMM filter with observation noise from NOAH's method
    [ xhat_n, P_k_n, mu_n, x1hat_n, x2hat_n, P1_k_n, P2_k_n ] =...
        imm(mu_n, x1hat_n, x2hat_n, P1_k_n, P2_k_n, obs_data(:,i), sigma_n_p);
    mu1_n = mu_n(1);
    mu2_n = mu_n(2);

    xhat_n = x1hat_n * mu1_n + x2hat_n * mu2_n;
    P_k_n = mu1_n * (P1_k_n + (x1hat_n - xhat_n) * (x1hat_n - xhat_n)') +...
        mu2_n * (P2_k_n + (x2hat_n - xhat_n) * (x2hat_n - xhat_n)');
    xhat_imm_n(:,i) = xhat_n;

    % EKF with observation noise from angle noise based method
    [ xhat_v, P_k_v, mu_v, x1hat_v, x2hat_v, P1_k_v, P2_k_v ] =...
        imm(mu_v, x1hat_v, x2hat_v, P1_k_v, P2_k_v, obs_data(:,i), sigma_v_p);
    mu1_v = mu_v(1);
    mu2_v = mu_v(2);

    xhat_v = x1hat_v * mu1_v + x2hat_v * mu2_v;
    P_k_v = mu1_v * (P1_k_v + (x1hat_v - xhat_v) * (x1hat_v - xhat_v)') +...
        mu2_v * (P2_k_v + (x2hat_v - xhat_v) * (x2hat_v - xhat_v)');
    xhat_imm_v(:,i) = xhat_v;

end
```

B.2 SNR data generator

snr.m

```
function SNR = snr( p_true, pos_update )

% Jan S. Karlsen
% 26.05.2008
%
% ============== SNR calculation ==============
%
% Description:
% This file generates SNR data at updated position
%

K = 2.29e+018;
AzBw = 2 * pi / 180;
ElBw = 1.65 * pi / 180;
RCS = 10;
SNR = [0 0 0 0];

theta_dif = 1.85 * pi / 180;
T_d = 0.0015;
W = pi;
gamma = acos(W*T_d/theta_dif);

TError_angle = [ p_true(2) - pos_update(1),...
    p_true(3) - pos_update(2) ];

% Calculation of error angle
delta_Az(1) = (theta_dif/2) + TError_angle(1);
delta_El(1) = TError_angle(2);
delta_Az(2) = (theta_dif/2) * cos(gamma) + TError_angle(1);
delta_El(2) = TError_angle(2) - (theta_dif/2) * sin(gamma);
delta_Az(3) = TError_angle(1) - (theta_dif/2) * cos(gamma);
delta_El(3) = (theta_dif/2) * sin(gamma) + TError_angle(2);
delta_Az(4) = TError_angle(1) - (theta_dif/2);
delta_El(4) = TError_angle(2);

R = p_true(1) / cos(theta_dif/2);

% Calculation of SNR
for j=1:4

    L = 10^(1.2 * ((delta_Az(j) / AzBw)^2 + (delta_El(j) / ElBw)^2));
    SNR(j) = K * RCS / (R^4 * L);

end
```

B.3 Målbaner

ttraject.m

```
function [pos_traject, vel_traject] = ttraject( input_args )

% Jan S. Karlsen
% 26.05.2008
%
% ============== Target trajectory ==============
%
% Description:
% This file imports target trajectories from precreated tsa-files.
%
% Dependencies:
% - traj_1.tsa
% - traj_2.tsa
% - traj_3.tsa
% - traj_4.tsa
%

% -------------- Open/Close tsa-files --------------

fid = fopen('traj_1.tsa','r');
[ array, elements ] = fscanf(fid,'%g'); % Collect trajectory data from file
fclose(fid);


% -------------- Initialize data vectors --------------

N = length(array)/10;
x_coords = zeros(N,1);
y_coords = zeros(N,1);
z_coords = zeros(N,1);
x_velocity = zeros(N,1);
y_velocity = zeros(N,1);
z_velocity = zeros(N,1);
x_accelerate = zeros(N,1);
y_accelerate = zeros(N,1);
z_accelerate = zeros(N,1);
time = zeros(N,1);


% -------------- Allocate trajectory data to vectors --------------

A = reshape(array',10,N);
Array = A';

for k=1:N
    x_coords(k,1) = Array(k,2);
    y_coords(k,1) = Array(k,3);
    z_coords(k,1) = Array(k,4);
    x_velocity(k,1) = Array(k,5);
    y_velocity(k,1) = Array(k,6);
    z_velocity(k,1) = Array(k,7);
    x_accelerate(k,1) = Array(k,8);
    y_accelerate(k,1) = Array(k,9);
    z_accelerate(k,1) = Array(k,10);
    time(k,1) = Array(k,1);
end
```

```
pos_traject = [ x_coords y_coords z_coords...
    x_velocity y_velocity z_velocity...
    x_accelerate y_accelerate z_accelerate];
```

B.4 Monopulsbasert og vinkelbasert metode

Observasjonsdata_m.m

Observasjonsdata_v.m

```
function [pos_update, pos_prev ] = observation_data_m(SNR, pos_update, meas)

% Jan S. Karlsen
% 26.05.2008
%
% ============== Observation data ==============
%
% Description:
% This file updates observation data based on SNR
% measurements in the previous position
%

% -------------- Monopulse based method --------------

pos_prev = pos_update;
theta_er = [0 0 0 0];
theta_dif = 1.85 * pi / 180;
AzBw = 2 * pi / 180;
ElBw = 1.65 * pi / 180;
T_d = 0.0015;
W = pi;
gamma = acos(W*T_d/theta_dif);

% Adjustment of SNR elevation measurement angle
SNR1 = SNR(1) + (SNR(2)) * cos(gamma);
SNR4 = SNR(4) + (SNR(3)) * cos(gamma);
SNR2 = SNR(2) * sin(gamma);
SNR3 = SNR(3) * sin(gamma);

% Ratio calculations
Sum_SNR = SNR1 + SNR4 + SNR2 + SNR3;
Diff_SNR_El = SNR2 + SNR4 - (SNR1 + SNR3);
Diff_SNR_Az = SNR3 + SNR4 - (SNR1 + SNR2);

Error_ratio = [Diff_SNR_Az / Sum_SNR;  Diff_SNR_El / Sum_SNR];
Error_v = (theta_dif / 2) * Error_ratio;

% 45 degrees adjustment of SNR measurement angle
R_theta = [cos(pi / 4) sin(pi / 4); -sin(pi / 4) cos(pi / 4)];
theta_er = R_theta * Error_v;

% Gain for position update
K_Az = 1.1 + 10^(Error_ratio(1)-1);
K_El = 1.5 + 10^(Error_ratio(2)-1);

% Position update
Az_update = (pos_update(1)) + K_Az * theta_er(1);
El_update = (pos_update(2)) + K_El * theta_er(2);

pos_update = [Az_update El_update];
```

```
function [pos_update, pos_prev ] = observation_data_v(SNR, pos_update, meas)

% Jan S. Karlsen
% 26.05.2008
%
% ============== Observation data ==============
%
% Description:
% This file updates observation data based on SNR
% measurements in the previous position
%

% -------------- Angle based method --------------

pos_prev = pos_update;
theta_er = [0 0 0 0];
theta_dif = 1.85 * pi / 180;
AzBw = 2 * pi / 180;
ElBw = 1.65 * pi / 180;
T_d = 0.0015;
W = pi;
gamma = acos(W*T_d/theta_dif);


% Gain for position update
K_Az = 0.001572 * (1 + 2e-3 * (-log(meas(1)/4e4)));
K_El = 0.001116 * (1 + 1e-1 * (-log(meas(1)/4e4)));

% Calculation of logarithmic SNR data
SNR_db = 10*log10(SNR);

% Direct calculation of position update
Az_update = pos_update(1) + K_Az * (SNR_db(4) - SNR_db(1));
El_update = pos_update(2) + (K_El * (SNR_db(2) - SNR_db(3))...
    + (Az_update - pos_update(1)) * cos(gamma)) / sin(gamma);

pos_update = [Az_update El_update];
```

B.5 Bartons, NOAH's og vinkelstøybasert metode

Observasjonsstoy_b.m

Observasjonsstoy_n.m

Observasjonsstoy_v.m

```
function [ sigma_R, sigma_Az, sigma_El ] = observation_noise_b( SNR )

% Jan S. Karlsen
% 26.05.2008
%
% ============== Observation noise ==============
%
% Description:
% This file updates observation data based on SNR
% measurements in the previous position
%

% -------------- Barton's method --------------

AzBw = 2*2*pi/360;
ElBw = 1.65*2*pi/360;
theta_dif = 1.85 * pi / 180;
T_d = 0.0015;
W = pi;
gamma = acos(W*T_d/theta_dif);

% Calculation of theta at -3 dB
theta_Az = AzBw + theta_dif;
theta_El = ElBw + theta_dif;

% Gain for noise update
K_Az_v = 3e-3;
K_El_v = 1.3e-1;

% Updated observation noise calculations
sigma_R = 25;
sigma_Az = K_Az_v * theta_Az / (1.6 * sqrt((SNR(1) + SNR(4))));
sigma_El = K_El_v * theta_El / (1.6 * sqrt((SNR(2) + SNR(3))));
```

```
function [ sigma_R, sigma_Az, sigma_El ] = observation_noise_n( SNR )

% Jan S. Karlsen
% 26.05.2008
%
% ============== Observation noise ==============
%
% Description:
% This file updates observation data based on SNR
% measurements in the previous position
%

% -------------- NOAH's method --------------

AzBw = 2*2*pi/360;
ElBw = 1.65*2*pi/360;
theta_dif = 1.85 * pi / 180;
T_d = 0.0015;
W = pi;
gamma = acos(W*T_d/theta_dif);

% Calculation of stack factor
F_Az = AzBw / theta_dif;
F_El = ElBw / theta_dif;

beta(1) = (F_Az * AzBw / 2.25) * log(SNR(4)/SNR(1));
beta(2) = (F_Az * AzBw / 2.25) * log(SNR(2)/SNR(3));

alphaAz(1) = 2.25*(inv(2*F_Az) + (beta(1)/AzBw))^2;
alphaAz(2) = 2.25*(inv(2*F_Az) - (beta(1)/AzBw))^2;
alphaEl(1) = 2.25*(inv(2*F_El) + (beta(2)/ElBw))^2;
alphaEl(2) = 2.25*(inv(2*F_El) - (beta(2)/ElBw))^2;

% Gain for noise update
K_Az_v = 2.7e-3;
K_El_v = 1.3e-1;

% Updated observation noise calculations
sigma_R = 25;
sigma_Az = K_Az_v * (F_Az * AzBw) * sqrt(exp(alphaAz(1)) + exp(alphaAz(2)))...
    / (2.25 * sqrt(2 * (SNR(1)+SNR(4))/2));
sigma_El = K_El_v * (F_El * ElBw) * sqrt(exp(alphaEl(1)) + exp(alphaEl(2)))...
    / (2.25 * sqrt(2 * (SNR(2)+SNR(3))/2));
```

```
function [ sigma_R, sigma_Az, sigma_El ] = observation_noise_v( SNR )

% Jan S. Karlsen
% 26.05.2008
%
% ============== Observation noise ==============
%
% Description:
% This file updates observation data based on SNR
% measurements in the previous position
%

% -------------- Angle noise based method --------------

theta_dif = 1.85 * pi / 180;
T_d = 0.0015;
W = pi;
gamma = acos(W*T_d/theta_dif);

% Gain for noise update
K_Az_v = 7.86e-5;
K_El_v = 2.232e-3;

% Updated observation noise calculations
sigma_R = 25;
sigma_Az = (K_Az_v / sqrt(2)) * abs(sqrt(inv(SNR(4))...
    - inv(SNR(1))));
sigma_El = (K_El_v / sqrt(2)) * abs(sqrt((SNR(3)...
    - SNR(2)) / (SNR(2) * SNR(3) * sin(gamma))));
```

B.6 Initialisering av utvidet Kalman-filter

kalman_init.m

```
function [xhat_init, P_k_init] = kalman_init(meas, sigma)

% Jan S. Karlsen
% 26.05.2008
%
% ============== Kalman filter initialization ==============
%
% Description:
% This file calculates the initial conditions for the Kalman
% filter.
%

% -------------- Calculation of initial xhat  --------------

deltat = 2; % Timestep (Sample time)

R_init1 = meas(1,1);
Az_init1 = meas(2,1);
El_init1 = meas(3,1);

R_init2 = meas(1,2);
Az_init2 = meas(2,2);
El_init2 = meas(3,2);

x_init1 = R_init1 * cos(Az_init1) * cos(El_init1);
y_init1 = R_init1 * sin(Az_init1) * cos(El_init1);
z_init1 = -R_init1 * sin(El_init1);

x_init2 = R_init2 * cos(Az_init2) * cos(El_init2);
y_init2 = R_init2 * sin(Az_init2) * cos(El_init2);
z_init2 = -R_init2 * sin(El_init2);

v_x_init = (x_init2 - x_init1) / deltat;
v_y_init = (y_init2 - y_init1) / deltat;
v_z_init = (z_init2 - z_init1) / deltat;

xhat_init = [ x_init1 y_init1 z_init1 v_x_init v_y_init v_z_init]';


% -------------- Calculation of initial P_k  --------------

sigma_R = sigma(1);
sigma_Az = sigma(2);
sigma_El = sigma(3);

sigma_x = sqrt((x_init1 / R_init1)^2 * sigma_R^2 ...
    + (y_init1 * sigma_Az)^2 ...
    + ((y_init1 * z_init1)^2/(x_init1^2 + z_init1^2))*sigma_El^2);

sigma_y = sqrt((y_init1 / R_init1)^2 * sigma_R^2 ...
    + (x_init1 * sigma_Az)^2 ...
    + ((y_init1 * z_init1)^2/(x_init1^2 + z_init1^2))*sigma_El^2);

sigma_z = sqrt((z_init1 / R_init1)^2 * sigma_R^2 ...
        + (x_init1^2 + y_init1^2)*sigma_El^2);
```

```
w_init = diag([sigma_x^2 sigma_y^2 sigma_z^2]);

P_k_init = [w_init w_init/deltat; w_init/deltat 2*w_init/deltat];
```

B.7 Initialisering av multiple samvirkende modeller filter

imm_init.m

```
function [xhat_init, P_k_init] = imm_init(meas, sigma)

% Jan S. Karlsen
% 26.05.2008
%
% ============== IMM filter initialization ==============
%
% Description:
% This file calculates the initial conditions for the IMM
% filter.
%

% -------------- Calculation of initial xhat  --------------

deltat = 2; % Timestep (Sample time)

R_init1 = meas(1,1);
Az_init1 = meas(2,1);
El_init1 = meas(3,1);

R_init2 = meas(1,2);
Az_init2 = meas(2,2);
El_init2 = meas(3,2);

x_init1 = R_init1 * cos(Az_init1) * cos(El_init1);
y_init1 = R_init1 * sin(Az_init1) * cos(El_init1);
z_init1 = -R_init1 * sin(El_init1);

x_init2 = R_init2 * cos(Az_init2) * cos(El_init2);
y_init2 = R_init2 * sin(Az_init2) * cos(El_init2);
z_init2 = -R_init2 * sin(El_init2);

v_x_init = (x_init2 - x_init1) / deltat;
v_y_init = (y_init2 - y_init1) / deltat;
v_z_init = (z_init2 - z_init1) / deltat;

xhat_init = [ x_init1 y_init1 z_init1...
              v_x_init v_y_init v_z_init...
              0 0 0]';


% -------------- Calculation of initial P_k  --------------

sigma_R = sigma(1);
sigma_Az = sigma(2);
sigma_El = sigma(3);

sigma_x = sqrt((x_init1 / R_init1)^2 * sigma_R^2 ...
    + (y_init1 * sigma_Az)^2 ...
    + ((y_init1 * z_init1)^2/(x_init1^2 + z_init1^2))*sigma_El^2);

sigma_y = sqrt((y_init1 / R_init1)^2 * sigma_R^2 ...
    + (x_init1 * sigma_Az)^2 ...
    + ((y_init1 * z_init1)^2/(x_init1^2 + z_init1^2))*sigma_El^2);

sigma_z = sqrt((z_init1 / R_init1)^2 * sigma_R^2 ...
```

```
        + (x_init1^2 + y_init1^2)*sigma_El^2);

w_init = diag([sigma_x^2 sigma_y^2 sigma_z^2]);

P_k_init = [w_init w_init/deltat zeros(3,3);...
    w_init/deltat 2*w_init/deltat zeros(3,3);...
    zeros(3,3) zeros(3,3) zeros(3,3)];
```

B.8 Utvidet Kalman-filter

kalman.m

```
function [ xhat, P_k, res_new, res_p, K_k ] = kalman(sigma, xhat, meas, P_k, res_prev)

% Jan S. Karlsen
% 26.05.2008
%
% ============== Extended Kalman Filter ==============
%
% Description:
% This file constitutes a recursive filter for state
% estimation in stochastic systems
%
% State-space model
% x(k+1) = PHI * x(k) + DELTA * u(k) + GAMMA * w(k)
% z(k) = H(x(k)) * x(k) + G(x(k)) * v(k)
%

deltat = 2; % Timestep (Sample time)
PHI = [ 1 0 0 deltat 0 0;...
    0 1 0 0 deltat 0;...
    0 0 1 0 0 deltat;...
    0 0 0 1 0 0;...
    0 0 0 0 1 0;...
    0 0 0 0 0 1 ]; % System matrix
G = eye(3);
Q = 0.95 * diag([ 0 0 0 1 1 1 ]);
res = [ 0 0 0 ];

% -------------- Noise matrices  --------------

sigma_R = sigma(1);
sigma_Az = sigma(2);
sigma_El = sigma(3);

v_k = diag([ sigma_R^2 sigma_Az^2 sigma_El^2 ]); % Measurement noise

xhat = PHI * xhat; % Aposteriori estimate

% -------------- Measurement calculation --------------

x_i = xhat(1);
y_i = xhat(2);
z_i = xhat(3);

R_i = sqrt(x_i^2 + y_i^2 + z_i^2);
Az_i = atan2(y_i,x_i);
El_i = atan2(-z_i,sqrt(x_i^2 + y_i^2));
zhat = [R_i Az_i El_i];

% -------------- Linearization --------------

dR_dx = (x_i) / sqrt(x_i^2 + y_i^2 + z_i^2);
dR_dy = (y_i) / sqrt(x_i^2 + y_i^2 + z_i^2);
dR_dz = (z_i) / sqrt(x_i^2 + y_i^2 + z_i^2);
dAz_dx = -y_i / (x_i^2 + y_i^2);
dAz_dy = x_i / (x_i^2 + y_i^2);
dAz_dz = 0;
```

```
dEl_dx = (z_i * x_i) / (sqrt(x_i^2 + y_i^2)*(x_i^2 + y_i^2 + z_i^2));
dEl_dy = (z_i * y_i) / (sqrt(x_i^2 + y_i^2)*(x_i^2 + y_i^2 + z_i^2));
dEl_dz = -sqrt(x_i^2 + y_i^2) / (x_i^2 + y_i^2 + z_i^2);


H_k =[ dR_dx dR_dy dR_dz 0 0 0; ...
    dAz_dx dAz_dy dAz_dz 0 0 0; ...
    dEl_dx dEl_dy dEl_dz 0 0 0]; % Linearized measuerement matrix

% -------------- Angle detection --------------

res(1) = meas(1) - zhat(1);

for i = 2:3
    if (meas(i)>0 && zhat(i)<0)
        res(i) = meas(i) + zhat(i);
    elseif (meas(i)<0 && zhat(i)>0)
        res(i) = meas(i) + zhat(i);
    else
        res(i) = meas(i) - zhat(i); % Residual (Estimaion error)
    end
end

res_p=res;

% -------------- Maneuver detection --------------

man_thres = [10 0.00001 0.00001];

for k = 1:9
    res_new(:,k) = res_prev(:,k+1);
end

res_new(:,10) = res;

res_rate = (res_new- res_prev)/deltat;

for i = 1:3
    res_mean = abs(mean(res_rate(i,:)));
    if(res_mean > man_thres(i))
        Q = Q + 10000*diag([0 0 0 1 1 1]);
        P_k = P_k + 400*diag([1 1 1 0 0 0 ]);
    else
        Q = diag([0 0 0 1 1 1]);
    end
end


% -------------- Kalman filter equations --------------

P_k = PHI * P_k * PHI' + Q; % Error covariance of next timestep

R = G * v_k * G; % Measurement covariance noise matrix

% S_k = H_k * P_k * H_k' + R_k; % Estimate deviation covariance update
```

```
K_k = P_k * H_k' * inv(H_k * P_k * H_k' + R); % Kalman gain

xhat = xhat + K_k * res'; % Estimate update

P_k = (eye(6) - K_k * H_k) * P_k * (eye(6) - K_k * H_k)'...
    + K_k * R * K_k'; % Estimate covariance update
```

B.9 Multiple samvirkende modeller filter

imm.m

imm1.m

imm2.m

```
function [ xhat, P_k, mu, x1hat, x2hat, P1_k, P2_k ] =...
    imm(mu, x1hat, x2hat, P1_k, P2_k, meas, sigma)

% Jan S. Karlsen
% 26.05.2008
%
% ============== IMM Filter ==============
%
% Description:
% This file constitutes a filter for state estimation
% in stochastic systems
%
% Dependencies:
% - imm1.m
% - imm2.m
%

p_k = [0.999 0.001; 0.001 0.999]; %  Markov chain transition matrix
mu1 = mu(1);
mu2 = mu(2);
chat1 = p_k(1,1) * mu1 + p_k(2,1) * mu2;
chat2 = p_k(2,1) * mu1 + p_k(2,2) * mu2;

% Combination propability
mu11 = (1 / chat1) * p_k(1,1)*mu1;
mu12 = (1 / chat2) * p_k(1,2)*mu1;
mu21 = (1 / chat1) * p_k(2,1)*mu2;
mu22 = (1 / chat2) * p_k(2,2)*mu2;

% Initial state state and covariance estimate
x0hat_1 = x1hat * mu11 + x2hat * mu21;
x0hat_2 = x1hat * mu12 + x2hat * mu22;
P0_1 = mu11 * (P1_k + (x1hat - x0hat_1) * (x1hat - x0hat_1)') +...
    mu21 * (P2_k + (x2hat - x0hat_1) * (x2hat - x0hat_1)');
P0_2 = mu12 * (P1_k + (x1hat - x0hat_2) * (x1hat - x0hat_2)') +...
    mu22 * (P2_k + (x2hat - x0hat_2) * (x2hat - x0hat_2)');

% Model state and covarince estimate and model likelihood
[ x1hat, P1_k, Lambda1 ] = imm1( sigma,...
    x0hat_1, meas, P0_1); % Model 1
[ x2hat, P2_k, Lambda2 ] = imm2( sigma,...
    x0hat_2, meas, P0_2); % Model 2 (manouvering)

% Normalized constant
c = Lambda1 * chat1 + Lambda2 * chat2;

% Model propability update
mu1 = (1 / c) * Lambda1 * chat1;
mu2 = (1 / c) * Lambda2 * chat2;
mu = [mu1 mu2];

% Filter state and covariance output
xhat = x1hat * mu1 + x2hat * mu2;
P_k = mu1 * (P1_k + (x1hat - xhat) * (x1hat - xhat)') +...
    mu2 * (P2_k + (x2hat - xhat) * (x2hat - xhat)');
```

```
function [ xhat, P_k, Lambda ] = imm1(sigma, xhat, meas, P_k)

% Jan S. Karlsen
% 26.05.2008
%
% ============== IMM model 1 ==============
%
% Description:
% This file constitutes the velocity model for the
% IMM filter
%
% State-space model
% x(k+1) = PHI * x(k) + DELTA * u(k) + GAMMA * w(k)
% z(k) = H(x(k)) * x(k) + G(x(k)) * v(k)
%

deltat = 2; % Timestep (Sample time)
PHI = [ 1 0 0 deltat 0 0 0 0 0;...
    0 1 0 0 deltat 0 0 0 0;...
    0 0 1 0 0 deltat 0 0 0;...
    0 0 0 1 0 0 0 0 0;...
    0 0 0 0 1 0 0 0 0;...
    0 0 0 0 0 1 0 0 0;...
    0 0 0 0 0 0 1 0 0;...
    0 0 0 0 0 0 0 1 0;...
    0 0 0 0 0 0 0 0 1]; % System matrix

G = eye(3);
Q = 0.95 * diag([ 0 0 0 1 1 1 0 0 0 ]);
res = [ 0 0 0 ];


% -------------- Noise matrices  --------------

sigma_R = sigma(1);
sigma_Az = sigma(2);
sigma_El = sigma(3);

v_k = diag([ sigma_R^2 sigma_Az^2 sigma_El^2 ]); % Measurement noise
xhat = PHI * xhat; % Aposteriori estimate


% -------------- Measurement calculation --------------

x_i = xhat(1);
y_i = xhat(2);
z_i = xhat(3);

R_i = sqrt(x_i^2 + y_i^2 + z_i^2);
Az_i = atan2(y_i,x_i);
El_i = atan2(-z_i,sqrt(x_i^2 + y_i^2));
zhat = [R_i Az_i El_i];


% -------------- Linearization --------------
```

```
dR_dx = (x_i) / sqrt(x_i^2 + y_i^2 + z_i^2);
dR_dy = (y_i) / sqrt(x_i^2 + y_i^2 + z_i^2);
dR_dz = (z_i) / sqrt(x_i^2 + y_i^2 + z_i^2);
dAz_dx = -y_i / (x_i^2 + y_i^2);
dAz_dy = x_i / (x_i^2 + y_i^2);
dAz_dz = 0;
dEl_dx = (z_i * x_i) / (sqrt(x_i^2 + y_i^2)*(x_i^2 + y_i^2 + z_i^2));
dEl_dy = (z_i * y_i) / (sqrt(x_i^2 + y_i^2)*(x_i^2 + y_i^2 + z_i^2));
dEl_dz = -sqrt(x_i^2 + y_i^2) / (x_i^2 + y_i^2 + z_i^2);

H_k =[ dR_dx dR_dy dR_dz 0 0 0 0 0 0; ...
    dAz_dx dAz_dy dAz_dz 0 0 0 0 0 0; ...
    dEl_dx dEl_dy dEl_dz 0 0 0 0 0 0]; % Linearized measuerement matrix


% -------------- Angle detection --------------

res(1) = meas(1) - zhat(1);
for i = 2:3
    if (meas(i)>0 && zhat(i)<0)
        res(i) = meas(i) + zhat(i);
    elseif (meas(i)<0 && zhat(i)>0)
        res(i) = meas(i) + zhat(i);
    else
        res(i) = meas(i) - zhat(i); % Residual (Estimaion error)
    end
end


% -------------- Kalman filter equations --------------

P_k = PHI * P_k * PHI' + Q; % Error covariance of next timestep

R = G * v_k * G; % Measurement covariance noise matrix

S_k = H_k * P_k * H_k' + R; % Estimate deviation covariance update

K_k = P_k * H_k' * inv( H_k * P_k * H_k' + R ); % Kalman gain

xhat = xhat + K_k * res'; % Estimate update

P_k = (eye(9) - K_k * H_k) * P_k * (eye(9) - K_k * H_k)'...
    + K_k * R * K_k'; % Estimate covariance update


% -------------- Gaussian PDF --------------
res = res';
Lambda = exp(-0.5*(res'*inv(S_k)*res))/sqrt(det(2*pi*S_k));
```

```
function [ xhat, P_k, Lambda] = imm2(sigma, xhat, meas, P_k)

% Jan S. Karlsen
% 26.05.2008
%
% ============== IMM model 2 ==============
%
% Description:
% This file constitutes the acceleration model for the
% IMM filter
%
% State-space model
% x(k+1) = PHI * x(k) + DELTA * u(k) + GAMMA * w(k)
% z(k) = H(x(k)) * x(k) + G(x(k)) * v(k)
%

deltat = 2; % Timestep (Sample time)
PHI = [ 1 0 0 deltat 0 0 deltat^2/2 0 0;...
    0 1 0 0 deltat 0 0 deltat^2/2 0;...
    0 0 1 0 0 deltat 0 0 deltat^2/2;...
    0 0 0 1 0 0 deltat 0 0;...
    0 0 0 0 1 0 0 deltat 0;...
    0 0 0 0 0 1 0 0 deltat;...
    0 0 0 0 0 0 1 0 0;...
    0 0 0 0 0 0 0 1 0;...
    0 0 0 0 0 0 0 0 1 ]; % System matrix

G = eye(3);
Q = 20 * diag([ 0 0 0 0 0 0 1 1 1 ]);
res = [ 0 0 0 ];


% -------------- Noise matrices  --------------

sigma_R = sigma(1);
sigma_Az = sigma(2);
sigma_El = sigma(3);

v_k = diag([ sigma_R^2 sigma_Az^2 sigma_El^2 ]); % Measurement noise
xhat = PHI * xhat; % Aposteriori estimate


% -------------- Measurement calculation --------------

x_i = xhat(1);
y_i = xhat(2);
z_i = xhat(3);

R_i = sqrt(x_i^2 + y_i^2 + z_i^2);
Az_i = atan2(y_i,x_i);
El_i = atan2(-z_i,sqrt(x_i^2 + y_i^2));
zhat = [R_i Az_i El_i];


% -------------- Linearization --------------
```

```
dR_dx = (x_i) / sqrt(x_i^2 + y_i^2 + z_i^2);
dR_dy = (y_i) / sqrt(x_i^2 + y_i^2 + z_i^2);
dR_dz = (z_i) / sqrt(x_i^2 + y_i^2 + z_i^2);
dAz_dx = -y_i / (x_i^2 + y_i^2);
dAz_dy = x_i / (x_i^2 + y_i^2);
dAz_dz = 0;
dEl_dx = (z_i * x_i) / (sqrt(x_i^2 + y_i^2)*(x_i^2 + y_i^2 + z_i^2));
dEl_dy = (z_i * y_i) / (sqrt(x_i^2 + y_i^2)*(x_i^2 + y_i^2 + z_i^2));
dEl_dz = -sqrt(x_i^2 + y_i^2) / (x_i^2 + y_i^2 + z_i^2);


H_k =[ dR_dx dR_dy dR_dz 0 0 0 0 0 0; ...
    dAz_dx dAz_dy dAz_dz 0 0 0 0 0 0; ...
    dEl_dx dEl_dy dEl_dz 0 0 0 0 0 0]; % Linearized measuerement matrix


% -------------- Angle detection --------------

res(1) = meas(1) - zhat(1);
for i = 2:3
    if (meas(i)>0 && zhat(i)<0)
        res(i) = meas(i) + zhat(i);
    elseif (meas(i)<0 && zhat(i)>0)
        res(i) = meas(i) + zhat(i);
    else
        res(i) = meas(i) - zhat(i); % Residual (Estimaion error)
    end
end


% -------------- Kalman filter equations --------------

P_k = PHI * P_k * PHI' + Q; % Error covariance of next timestep

R = G * v_k * G; % Measurement covariance noise matrix

S_k = H_k * P_k * H_k' + R; % Estimate deviation covariance update

K_k = P_k * H_k' * inv( S_k ); % Kalman gain

xhat = xhat + K_k * res'; % Estimate update

P_k = (eye(9) - K_k * H_k) * P_k * (eye(9) - K_k * H_k)'...
    + K_k * R * K_k'; % Estimate covariance update


% -------------- Gaussian PDF --------------
res = res';
Lambda = exp(-0.5*(res'*inv(S_k)*res))/sqrt(det(2*pi*S_k));
```

C. Initialisering og avslutning av målfølging

tracking_initialization.m

tracking_termination.m

```
% Jan S. Karlsen
% 26.05.2008
%
% ============== Target initialization ==============
%
% Description:
% This file initializes target tracking
%
% Input:
% - Number of previous targets missed
%

% -------------- Tracking initialization --------------

target_data = []; % Contains updated position and missed tracks
target_nr = 0;

%------------- Search routine -------------

for i = 1:16 % Sectors in elevation
    for j = 1:256 % Sectors in azimuth

        tracking_distance = 0;

%------------- Target detection -------------

        if( SNR_detect > Eps_SNR ) % SNR detection thresold

%------------- Distance check between targets -------------

            nr_of_targets = length(:,1);

            for k = 1:nr_of_targets
                for l = 1:2
                    if ( target_data(k,l) - new_target_pos(l) < Eps_distance )
                        % Distance thresold is 150 [m]
                        track_distance = 1; % Distance between targets are too short
                    end
                end
            end
        end

        if ( track_distance == 0 )

 %------------- Backscanning -------------

            target_pos = [ i j ];
            [ SNR1 SNR2 SNR3 SNR4 ] = backscan( target_pos ); % Acquisition of SNR data

%------------- Position update -------------

            new_target_pos = observation_data( SNR1, SNR2, SNR3, SNR4 );

%------------- Generate target data -------------

            target_data(target_nr,:) = [ new_target_pos(1) new_target_pos(2)...
```

```
                tracks_missed];

        end
    end
end
```

```
% Jan S. Karlsen
% 26.05.2008
%
% ==============  Tracking termination ==============
%
% Description:
% This file terminates target tracking
%
% Input:
% - Number of previous targets missed
% - Previous target_data
%

nr_of_targets = length(target_data(:,1));

%------------- Missed track check -------------

for i = 1:nr_of_targets

    if (target_data(i,3) > 3) % Number of missed tracking attempts
        target_data(i) = [0 0 0]; % Delete track
    end

end
```