UNIVERSITETET I OSLO
Fysisk institutt

Analyse av Treghets-navigasjons Systemer

Masteroppgave

(30 studiepoeng)

Petter Moagi Lefoka

21. Mai 2013

Forord

Denne oppgaven er en del av min mastergrad elektronikk og datateknologi ved fysisk institutt på universitetet i Oslo. Jeg vil gjerne takke veilederen min, Oddvar Hallingstad for all støtte og innspill til denne oppgaven, og jeg vil takke moren min for å ha lest korrektur.

Petter Moagi Lefoka

Kjeller 21. Mai 2013

Sammendrag

I denne oppgaven er en analyse av treghetsnavigasjons systemer. Målet har hvert å finne minimums kvalitet akselerometer og gyrometer sensorer må opprettholde for treghetsnavigasjons systemer til forskjellige bruksområder.

Dette har blitt gjort ved å lage en mattematiskmodell av systemet som inneholder først èn banegenerator og et støymodellerings program der det blir generert målinger fra en kjent bane så genereres støy som inneholder hvitstøy og fargetstøy som adderes på målingene. Deretter et TNS system som gjør navigasjonssimuleringer på de stokastiske målingene, med et Kalmanfilter som minimerer driften gitt av støyen og integreringsfeil. Tilslutt er det et kovariansanalyseprogram for og se hva som må bli gjort av endringer i det teoretiske systemet for at det skal kunne opprettholde minimumsdriftkravene.

Det har blitt simulert på to forskjell forskjellige senarioer, der det første er et senario der målet er og kunne måle konturene til et lite objekt med en mobiltelefon, der bevegelses område er lite og det blir brukt nullhastighets måleoppdateringer for å korrigere for driften. I det andre senarioet er det ett modell fly der bevegelses området er mye større og måleoppdateringer blir gjort ved hjelp av GPS, i dette senarioet må sensorene være gode nokk til at driften i TNS'et blir mindre en 1 meter når det går minst 60 sekunder uten måleoppdateringer.

INNHOLD

Notasjon

Banegenerator:

R_n^b	Transformasjonsmatrise fra n til b
$\underline{p}^n$	Sann posisjonsvektor
$\underline{v}^n$	Sann hastighetsvektor
$\underline{a}^n$	Sann akselerasjonsvektor
$\underline{f}^n$	akselerasjonsvektor sett fra n
$\underline{\omega}_b^{nb}$	Vinkelhastighetsvektor sett fra n

Kalman:

$\underline{x}$	Sann tilstandsvektor
$\delta\underline{x}$	feilvektor
$\delta\overline{\underline{x}}$	Prediktert feilvektor
$\delta\widehat{\underline{x}}$	Estimert feilvektor
$\underline{z}$	Målevektor
H	Målematrise

Kontinuerlig:

$\underline{x}(t)$	Tilstandsvektor
$\underline{u}(t)$	Pådragsvektor
$\underline{v}(t)$	Hvitstøyvektor
$F(t)$	Prosessmatrise
$L(t)$	Pådragsmatrise
$G(t)$	Støymatrise
$P(t)$	Kovariansmatrise
$\tilde{Q}$	Spektraltetthetsmatrise

Diskret:

$\underline{x}_k$	Tilstandsvektor
$\underline{u}_k$	Pådragsvektor
$\underline{v}_k$	Hvitstøyvektor
Φ_k	Prosessmatrise
Λ_k	Pådragsmatrise
Γ_k	Støymatrise
P_k	Kovariansmatrise
Q	Spektraltetthetsmatrise

NAV Prosessmodell:

$\underline{x}$	Sann tilstandsvektor
$\underline{\tilde{x}}$	Estimert tilstandsvektor
$\underline{\tilde{p}}^n$	Estimert posisjonvektor sett fra n
$\underline{\tilde{v}}^n$	Estimert hastighetvektor sett fra n
$\tilde{R}_b^n$	Estimert transformasjonsmatrise fra b til n
$\underline{f}^b$	Akselerasjon sett fra b
$\underline{\omega}_b^{bb}$	Vinkelhastighet sett fra b
$\underline{\gamma}$	Støyvektor gamma
$\underline{\beta}$	Støyvektor beta
$\underline{v}$	Hvitstøyvektor

Del A: Innledning

Denne oppgave er basert på flere tidligere masteroppgaver der temaet har hvert treghetsnavigasjons systemer. Der det blant annet tidligere har blitt lagd en Android app. Som kan logge data fra akselerometer og gyrometer sensorer, og det blitt lagd et program som estimerer gyroparametre fra stokastiske målinger. Det har også blitt gjort en simuleringsoppgave der det ble generert deterministiske måledata til sensormodeller for en enkel bane og et Kalmanfilter, men Kalmanfilteret er enda ikke testet på stokastisk måledata.

Målet med denne oppgaven er og hovedsakelig gjøres en analyse av treghetsnavigasjons systemer der det må undersøkes hvilken effekt forskjellige typer støy har på et treghetsnavigasjons system og det skal undersøkes hvilke spesifikasjoner de forskjellige sensorene må ha for at TNS'et skal fungere optimalt. Da må det bygges videre på den sistnevnte oppgaven, der det må lages et støymodelleringsprogram for få testet Kalmanfilteret på stokastiske målinger. Det må også lages et kovariansanalyseprogram som kan isolere effekten de forskjellige støykildene har på et TNS.

Videre skal det også lages en mattematiskmodell for et TNS beregnet modellfly med en banegenerator som genererer stokastiske målinger, der sensorene er en klasse bedre enn mobiltelefon sensorer. Og det skal gjøres en kovariansanalyse på dette systemet for og se hvilken effekt de samme støykildene har på et system med høyere nøyaktighetskrav.

Del B: Matematiskgrunnlag

Treghetsnavigasjon er navigasjon der man beregner posisjon, hastighet og orientering i rommet, ved å integrere opp akselerasjon og vinkelhastighets målinger over tid. Denne metoden trenger i teorien ikke noen eksterne referanse kilder til å beregne posisjonen, og er derfor mye brukt i romfart, luftfart og maritime sammenhenger. I praksis så er driften i et TNS en funksjon av tiden, og over tid vil derfor være nødvendig med en ekstern korrigerings kilde. I denne oppgaven skal det modelleres et TNS basert på MEMS (MicroElectroMechanical Systems) sensorer, også kjent som lavkost sensorer.

B.1 Navigasjonsrammer

B.1.1 Jordramme {e}

Jordrammen er viser posisjonen i forhold til jordkloden, den er ikke i bruk i oppgaven men er som regel en ECEF (Earth centered, earth fixed) ramme, med origo i jordens massesenter, og akser som sammenfaller med IRP (International reference pole) og IRM (International reference merdian).

B.1.2 Geografiskramme, {n}

Den geografiske rammen, er en lokal representasjon av område det navigeres i. Den geografiske rammen kan være en geografiskrepresentasjon av område legemet oppholder seg i der aksene til rammen sammenfaller med nord, øst og opp. Eller den kan være et enkelt kartesisk koordinatsystem, der origo ligger i et kjent punkt ofte p_0.

B.1.3 Legemeramme, {b}.

Legemerammen har origo i massesenteret til legemet som det navigeres fra, aksene sammenfaller med roll, pitch og yaw aksene til legemet.

B.1.4 Akselerometerramme og Gyrometerramme, {a} og {g}.

Akselerometer og gyrometer rammene ligger i hvert sitt kartesiske koordinatsystem, de har sitt origo i massesenteret til henholdsvis akselerometeret og gyrometeret. Det er her antatt at de sammenfaller med legemerammen.

B.2 Vektorer

Normen til en vektor $\underline{x}$ er definert ved $||\underline{x}|| = \sqrt{x_1^2 + x_2^2 + x_3^2}$

Enhetsvektoren er en retningsvektor med lengde en og er definert ved $\underline{b} = \frac{\underline{x}}{||\underline{x}||}$

B.3 Skjevsymmetriskmatrise

En skjevsymmetriskmatrise A er en nxn matrise der den transponerte er lik den negative av

matrisen, slik at $A^T = -A$. Den skjevsymmetriskmatrisen av en vektor $\underline{x}$,er gitt av

$$S(\underline{x}) = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \tag{B- 1}$$

Når den skjevsymmetriskematrisen er definert slik kan kryssproduktet av to vektorer skrives slik:

$$\vec{a}x\vec{b} <=> S(\underline{a}) * \underline{b} \tag{B- 2}$$

B.4 Koordinattransformasjonsmatrise (KTM)

Koordinattransformasjonsmatrisen transformerer koordinater mellom koordinatsystemer. I denne oppgaven blir det mellom geografiskramme {n} og legemerammen {b}. Matrisen R_b^n beskriver en transformasjon fra legemerammen {b} til geografiskramme {n}. Hvis vektoren $\underline{\theta}$ er bygd opp av eulervinklene $\begin{bmatrix} \theta & \varphi & \psi \end{bmatrix}^T$ som står for roll, pitch og yaw. Når eulervinklene beskriver orienteringen til {b} i forhold til {n}, kan koordinattransformasjonsmatrisen defineres som $R_b^n = R_{321}(\underline{\theta})$. Der rotasjonsmatrisen $R_{321}(\underline{\theta})$ beskriver en rotasjon rundt z-y-x aksene.

$R_{321}(\underline{\theta})$ er difinert som:

$$R_{321}(\underline{\theta}) = R_3(\psi) * R_2(\varphi) * R_1(\theta) \tag{B- 3}$$

Der R_3 en rotasjon rundt z-aksen, R_2 en rotasjon rundt y-aksen og R_1 en rotasjon rundt x-aksen er gitt av

$$R_3(\theta_3) = \begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 \\ \sin(\theta_3) & \cos(\theta_3) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{B- 4}$$

$$R_2(\theta_2) = \begin{bmatrix} \cos(\theta_2) & 0 & \sin(\theta_2) \\ 0 & 1 & 0 \\ -\sin(\theta_2) & 0 & \cos(\theta_2) \end{bmatrix} \tag{B- 5}$$

$$R_1(\theta_1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_1) & -\sin(\theta_1) \\ 0 & \sin(\theta_1) & \cos(\theta_1) \end{bmatrix} \tag{B- 6}$$

Det gjør at $R_{321}(\underline{\theta})$ kan skrives slik, når $c_x = \cos(x)$ og $s_x = \sin(x)$.

$$R_{321}(\underline{\theta}) = \begin{bmatrix} c_{\theta_3}c_{\theta_2} & c_{\theta_3}s_{\theta_2}s_{\theta_1} - s_{\theta_3}c_{\theta_1} & c_{\theta_3}s_{\theta_2}c_{\theta_1} + s_{\theta_2}s_{\theta_1} \\ s_{\theta_3}c_{\theta_2} & s_{\theta_3}s_{\theta_2}s_{\theta_1} + c_{\theta_3}c_{\theta_1} & s_{\theta_3}s_{\theta_2}c_{\theta_1} - c_{\theta_3}s_{\theta_1} \\ -s_{\theta_2} & c_{\theta_2}s_{\theta_1} & c_{\theta_2}c_{\theta_1} \end{bmatrix} \tag{B- 7}$$

$$R_n^b = (R_b^n)^{-1} = R_b^{n\ T} \tag{B- 8}$$

B.5 Støy

B.5.1 Hvitstøy

Hvitstøy er definert som et signal med null som forventningsverdi og en uniform sannsynlighetstetthetsfordeling med uendelig båndbredde. Et hvitstøy signal med en båndbredde som dekker hele frekvensspekteret er vanskelig å modellere, og det blir derfor ofte brukt gausiskhvitstøy som en tilnærming.

Browns bevegelse er integrert hvitstøy som fører til en tilfeldig vandring fra forventingsverdien også kalt virrevandring (random walk). Denne typen støy er hvit støy gjeldende på $\dot{\omega}$ og $\dot{f}$ nivå gitt av

$$\dot{x} = w \tag{B- 9}$$
$$der\ w \backsim N(0, \sigma^2) \tag{B- 10}$$

B.5.2 Fargetstøy

Fargetstøy forekommer når hvitstøy blir filteret. Den fargetstøy prosessen som beskrives i oppgaven er en første ordens Gaus-Markov prosess. En Gaus-Markov prosess er beskrevet av

$$\dot{x} = \frac{1}{T}x + w \tag{B- 11}$$
$$der\ w \backsim N(0, \sigma^2) \tag{B- 12}$$

Der $\frac{1}{T}$ er korelasjons tiden, og w er et hvitstøy element. En prosess er en første ordens Markov prosess hvis

$$p[x(t_k)|x(t_{k-1}), ..., x(t_1)] = p[x(t_k)|x(t_{k-1})] \tag{B- 13}$$

Dvs. hvis sannsynlighetstetthetsfunksjonen kun er avhengig av verdiene fra forrige tidsskritt. Hvis da hvitstøyelementet, w, har en gausisk sannsynlighetstetthetsfunksjon, kan prosessen kalles en Gauss-Markov prosess.

B.6 Diskretisering

Hvis man har et kontinuerlig-diskret system på formen

$$\dot{\underline{x}}(t) = F\underline{x}(t) + L\underline{u}(t) + G\underline{v}(t) \tag{B- 14}$$
$$\dot{P}(t) = FP(t) + P(t)F^T + G\widetilde{Q}G^T \tag{B- 15}$$
$$\underline{z}_k = H\underline{x}_k + \underline{w}_k \tag{B- 16}$$

Skrives diskretiseringslikningene slik

$$\Phi(t_{k+1,}t_k) = e^{F(t_{k+1}-t_k)} = e^{F\Delta t} \tag{B- 17}$$
$$\Lambda\underline{u}_k = \int_{t_k}^{t_{k+1}} \Phi(t_{k+1,}\tau)L\underline{u}(\tau)d\tau \tag{B- 18}$$
$$\Gamma Q\Gamma^T = S = \int_{t_k}^{t_{k+1}} \Phi(t_{k+1,}\tau)G\tilde{Q}G^T\Phi^T(t_{k+1,}\tau)d\tau \tag{B- 19}$$

Hvis man antar at $\underline{u}(t_k) = \underline{u}_k$ kan det lages et nytt system på formen $\frac{d}{dt}\widetilde{\underline{x}} = \widetilde{F}\widetilde{\underline{x}}$

$$\begin{bmatrix} \dot{\underline{x}} \\ \dot{\underline{u}} \end{bmatrix} = \begin{bmatrix} F & L \\ \underline{0} & \underline{0} \end{bmatrix} \begin{bmatrix} \underline{x} \\ \underline{u} \end{bmatrix} \ der\ \widetilde{F} = \begin{bmatrix} F & L \\ \underline{0} & \underline{0} \end{bmatrix}$$
$$\widetilde{\Phi} = e^{\widetilde{F}\Delta t} = \begin{bmatrix} \widetilde{\Phi}_{11} & \widetilde{\Phi}_{12} \\ \underline{0} & I \end{bmatrix} \tag{B- 20}$$

Slik at nå er

$$\begin{bmatrix} \underline{x}_{k+1} \\ \underline{u}_{k+1} \end{bmatrix} = \begin{bmatrix} \widetilde{\Phi}_{11} & \widetilde{\Phi}_{12} \\ \underline{0} & I \end{bmatrix} \begin{bmatrix} \underline{x}_k \\ \underline{u}_k \end{bmatrix} \tag{B- 21}$$

Nå kan systemet skrives slik $\underline{x}_{k+1} = \Phi\underline{x}_k + \Gamma\underline{u}_k = \widetilde{\Phi}_{11}\underline{x}_k + \widetilde{\Phi}_{12}\underline{u}_k$ man har da funnet Φ og Γ

matrisene.

For å finne lambda kan det konstrueers en $\tilde{\tilde{F}}$ matrise på formen

$$\tilde{\tilde{F}} = \begin{bmatrix} F & G\tilde{Q}G^T \\ \underline{0} & -F^T \end{bmatrix} \quad \text{(B- 22)}$$

$$e^{\tilde{\tilde{F}}\Delta t} = \begin{bmatrix} \tilde{\tilde{\Phi}}_{11} & \tilde{\tilde{\Phi}}_{12} \\ \underline{0} & \tilde{\tilde{\Phi}}_{22} \end{bmatrix} \quad \text{(B- 23)}$$

Slik at man kan finne $\Gamma Q \Gamma^T$ slik $\Gamma Q \Gamma^T = \tilde{\tilde{\Phi}}_{12}\tilde{\tilde{\Phi}}_{22}^{-1}$. Ut ifra $\Gamma Q \Gamma^T$ kan Γ bli funnet ved hjelp av UD faktorisering. Der $S = UDU^T = \Gamma Q \Gamma^T$

Hvis $Q = I$ kan stykket skrives om til $\Gamma * I * \Gamma^T = UD^{\frac{1}{2}} * I * (UD^{\frac{1}{2}})^T$ ved hjelp av Cholesky faktorisering, slik at $\Gamma = UD^{\frac{1}{2}}$.

Der $S = \Gamma\Gamma^T = UD^{\frac{1}{2}} * (UD^{\frac{1}{2}})^T$

$$\Gamma\Gamma^T = \begin{bmatrix} \Gamma_{11} & 0 & 0 \\ \Gamma_{21} & \Gamma_{22} & 0 \\ \Gamma_{31} & \Gamma_{32} & \Gamma_{33} \end{bmatrix} \begin{bmatrix} \Gamma_{11} & \Gamma_{12} & \Gamma_{13} \\ 0 & \Gamma_{22} & \Gamma_{23} \\ 0 & 0 & \Gamma_{33} \end{bmatrix} \quad \text{(B- 24)}$$

$$\Gamma_{ii} = \sqrt{A_{ii} - \sum_{k=1}^{i-1} L_{i,k}^2} \quad \text{(B- 25)}$$

$$\Gamma_{ji} = \frac{1}{\Gamma_{jj}}(A_{ji} - \sum_{k=1}^{i-1} L_{j,k}L_{i,k}) \quad \text{(B- 26)}$$

Dermed har man funnet Γ, og får det diskrete systemet

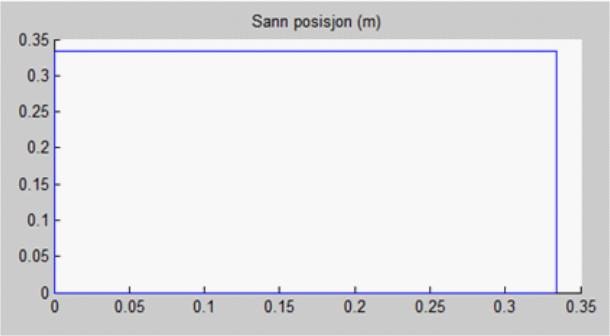
$$\underline{x}_{k+1} = \Phi\underline{x}_k + \Lambda\underline{u}_k + \Gamma\underline{v}_k \quad \text{(B- 27)}$$

$$P_{k+1} = \Phi P_k \Phi^T + \Gamma Q \Gamma^T \; der \; Q = I \; og \; P_0 = \Delta t * P(t_0) \quad \text{(B- 28)}$$

$$\underline{z}_k = H\underline{x}_k + \underline{w}_k \quad \text{(B- 29)}$$

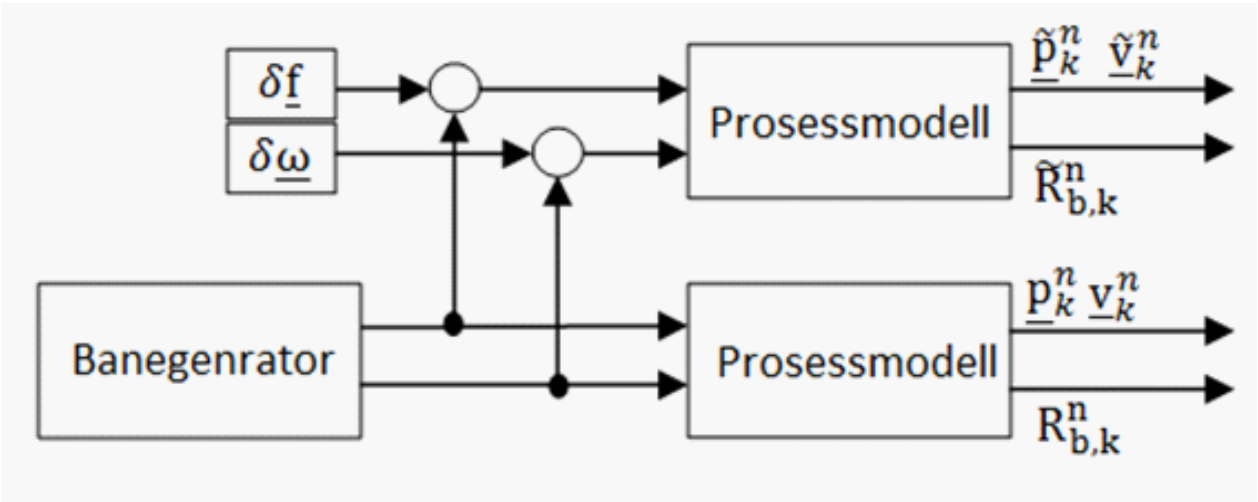
Del C:
Banegenerator

Banegeneratoren gir ut realistiske målinger for en forhåndsbestemt bane, de målingene brukes til og teste nøyaktigheten til navigasjons systemet. I første omgang modelleres banen til et kvadrat, der legemet (mobiltelefonen) som er i bevegelse stopper i to sekunder i hvert hjørne. Der banen sett rett oven i fra ser slik ut som i figuren under.



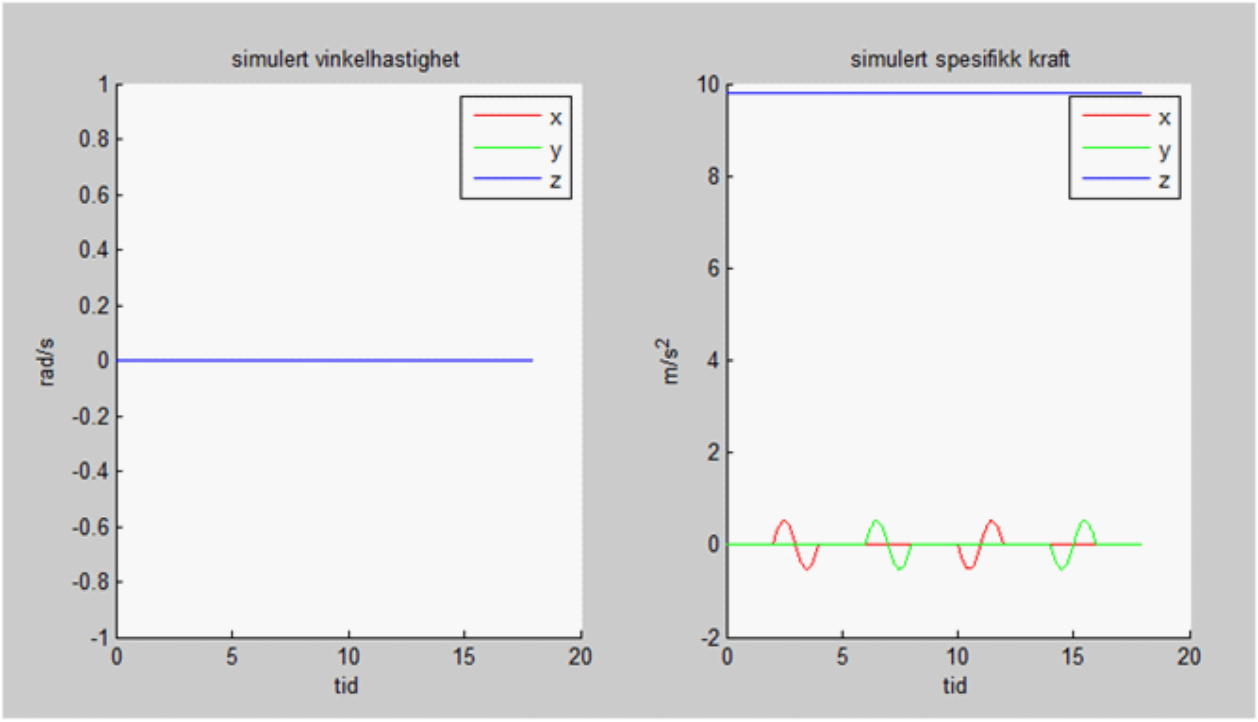
Banen generert av banegeneratoren

Figuren, prosessmodell banegenerator, viser blokkskjemat til banegeneratoren. Iden der er at det først blir generert støyfrie målinger som etter og ha kjørt igjennom treghetsnavigasjonslikningene gir sannposisjon, sannhastighet og sannorientering. Deretter blir støyen modellert og lagt til målingen slik at man har realistisk støyfylte målinger, som brukes til å teste treghetsnavigasjonssystemet.



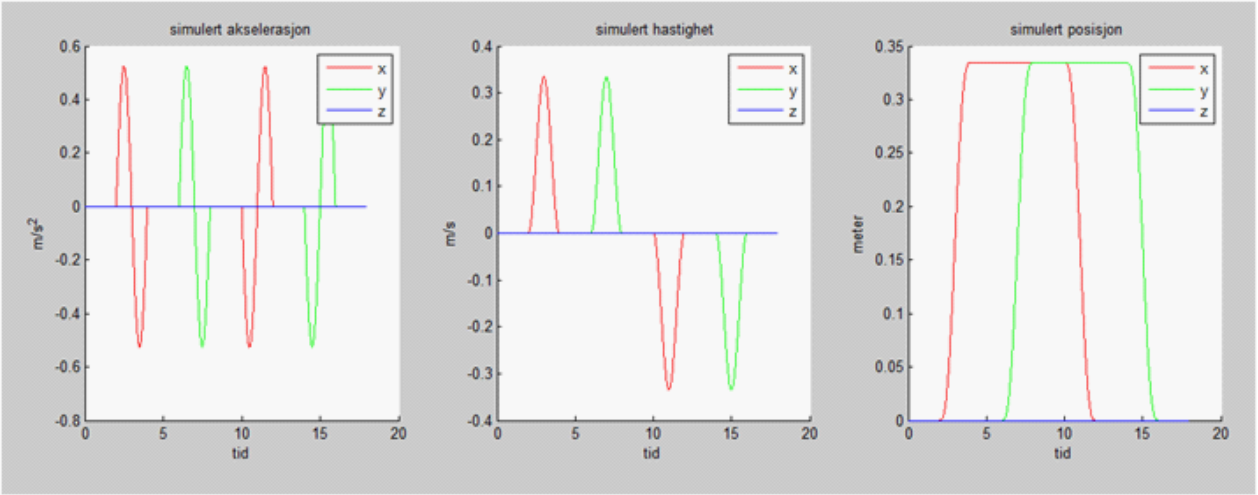
Prosessmodell, Banegenerator

For simuleringen av mobiltelefon MEMS-sensorer så ser målingene som blir generert fra banegeneratoren slik ut, figuren under. Der ses det at legemet roteres ikke. Langs z-aksen dvs. ned, måles det en konstant akselerasjon på 9.81 m/s^2. Og det kan ses at legemet står stille fra 0-2 sek, 4-6sek, 8-10 sek og 12-14 sek. (Legemet står også stille fra 16 til 18 sekunder, det er noe som ble lagt til senere og jeg kommer tilbake til det.) Der det kan bli gjort en nullhastighets måleoppdatering av Kalmanfilteret.



Simulert målt-akselerasjon og vinkelhastighet

Banegeneratoren gir også ut simulert akselerasjon, simulert hastighet og simulert posisjon, for og kunne sammenligne avviket fra beregnet hastighet og beregnet posisjon fra navigasjonssystemet med den sanne tilstanden. Den simulerte akselerasjonen, hastigheten og posisjonen ser slik ut, illustrert i figuren ove.



Sann akselerasjon, hastighet og posisjon.

Den simulerte hastigheten og posisjonen som figuren over illustrerer er det optimale resultatet man har som mål og sitte igjen med når alt er sagt og gjort og systemet er ferdig. Resultatet videre kommer til og bli vist i avvik fra denne banen.

C.1 Støymodellering

C.1.1 Mattematiskgrunnlag

Støyligningen som må modelleres er $\delta\underline{f}$ og $\delta\underline{\omega}$ de er gitt av

$$\delta\underline{f} = \underline{\gamma}^a + \underline{\beta}^a + \underline{\alpha}^a + \underline{v}^f \quad \text{(C- 30)}$$
$$\delta\underline{\omega} = \underline{\gamma}^g + \underline{\beta}^g + \underline{\alpha}^g + \underline{v}^\omega \quad \text{(C- 31)}$$

De inneholder et fargetstøy element, et integrert hvitstøy element og et hvitstøy element. De inneholder også en tilfeldig konstant, men det ses bort ifra siden det leddet kan inkluderes i det integrerte hvitstøy elementet. Slik at likningene som modelleres blir

$$\delta\underline{f} = \underline{\gamma}^a + \underline{\beta}^a + \underline{v}^f \quad \text{(C- 32)}$$
$$\delta\underline{\omega} = \underline{\gamma}^g + \underline{\beta}^g + \underline{v}^\omega \quad \text{(C- 33)}$$

Der selve gamma, beta og de tilhørende hvitstøyelementene er gitt av følgende likninger, som er like for både akselerometre og gyrometre.

$$\dot{\underline{\gamma}}(t) = \frac{1}{T}\underline{\gamma}(t) + \underline{v}_\gamma(t) \quad \text{(C- 34)}$$

der $\underline{\gamma}(t_0) = \underline{0}$ og $\underline{v}_\gamma(t) \tilde{} N(0, \underline{\sigma}_\gamma^2)$

$$\dot{\underline{\beta}}(t) = \underline{v}_\beta(t) \quad \text{(C- 35)}$$

der $\underline{v}_\beta(t) \tilde{} N(0, \underline{\sigma}_\beta^2)$ og $\underline{v}^x(t) \tilde{} N(0, \underline{\sigma}_x^2)$

C.1.2 Modellering

Størrelsen til variansen i hvitstøyelementene er proposjonal med driften over tid i systemet, i første omgang er det ønsket å ha en sammenlagt drift på cirka 1 deg/s og 1 mG/s.

Ved å bruke disse likningene kan det lages to prosessmodeller som beskriver støyen. Modellene er på formen:

$$\dot{\underline{x}}(t) = F\underline{x}(t) + G\underline{v}(t) \quad \text{(C- 36)}$$
$$\dot{P}(t) = FP(t) + P(t)F^T + G\tilde{Q}G^T \quad \text{(C- 37)}$$

Akselerometerstøy

For akselerometerstøyen kan likningene skrives slik.

$$\dot{\underline{\beta}}^a(t) = \underline{v}_\beta^a(t) \quad \text{(C- 38)}$$
$$\dot{\underline{\gamma}}^a(t) = \frac{1}{T^a}\underline{\gamma}^a(t) + \underline{v}_\gamma^a(t) \quad \text{(C- 39)}$$
$$\underline{v}^f(t) \tilde{} N(0, \underline{\sigma}_f^2) \quad \text{(C- 40)}$$

der $\underline{v}_\beta^a(t) \tilde{} N(0, \underline{\sigma}_\beta^2)$, $\underline{\gamma}^a(t_0) = \underline{0}$ og $\underline{v}_\gamma^a(t) \tilde{} N(0, \underline{\sigma}_{\gamma^a}^2)$

Der variansen til hvitstøyelementene settes til

$$\underline{\sigma}^2_{\beta^a} = (1mG/s)^2 = (0.00981m/s^3)^2 \quad \text{(C- 41)}$$
$$\underline{\sigma}^2_{\gamma^a} = (1mG/s)^2 = (0.00981m/s^3)^2 \quad \text{(C- 42)}$$
$$\underline{\sigma}^2_f = \underline{\tilde{q}}_f = (1mG/s)^2 = (0.00981m/s^3)^2 \quad \text{(C- 43)}$$

og korrelasjonstiden blir satt til $T^a = 2$

Hvis tilstandsvektoren og støyvektoren blir definert slik, $\underline{x}^a = \begin{bmatrix} \underline{\beta}^a & \underline{\gamma}^a \end{bmatrix}^T$ og $\underline{v}^a = \begin{bmatrix} \underline{v}^a_\beta & \underline{v}^a_\gamma \end{bmatrix}^T$.
Kan systemmodellen skrives slik,

$$\dot{\underline{x}}^a(t) = F^a \underline{x}^a(t) + G^a \underline{v}^a(t) \; der \; \underline{x}^a(t_0) \; er \; kjent. \quad \text{(C- 44)}$$
$$\dot{P}^a(t) = F^a P^a(t) + P^a(t) F^{aT} + G^a \tilde{Q}^a G^{aT} \; der \; P^a(t_0) \; og \; \tilde{Q}^a \; er \; kjent \quad \text{(C- 45)}$$

så blir F^a og G^a lik

$$F^a = \begin{bmatrix} \underline{0} & \underline{0} \\ \underline{0} & \frac{\underline{I}}{T^a} \end{bmatrix} \quad \text{(C- 46)}$$
$$G^a = \begin{bmatrix} I & \underline{0} \\ \underline{0} & I \end{bmatrix} \quad \text{(C- 47)}$$

Systemmodellen sier at initialverdiene $\underline{x}^a(t_0)$ *og* $P^a(t_0)$, og spektraltetthetmatrisen $\tilde{Q}^a$ er kjent. Initialverdien til tilstandsvektoren settes til null, $\underline{x}^a(t_0) = \underline{0}$. For og finne spektraltetthet og initiel kovorians må man ses på hvordan kovariansen utvikler seg. Utviklingen til kovariansen finner man ved og sette opp likningene for utviklingen til varians for beta og gamma elementene, det gir:

$$Beta \; : \; \dot{p}^a_\beta(t) = \widetilde{q}^a_\beta \quad \text{(C- 48)}$$
$$Gamma \; : \; \dot{p}^a_\gamma(t) = -\frac{2}{T^a} p^a_\gamma(t) + \widetilde{q}^a_\gamma \quad \text{(C- 49)}$$

Beta: Variansen til betaelementene øker linært siden den deriverte er lik en konstant, som er spektraltettheten til gitt varibel. Hvis den likningen integreres opp ser man at den initiele kovariansen er lik 0. $\dot{p}^a_\beta(t) = \widetilde{q}^a_\beta => p^a_\beta(t) = \widetilde{q}^a_\beta * t$ Som gir at initiel varians $p^a_\beta(0)$ må være lik 0 og spektraltettheten må være lik variansen ved 1 sekund.

$$p^a_\beta(0) = 0 \quad \text{(C- 50)}$$
$$\widetilde{q}^a_\beta = p^a_\beta(1) \quad \text{(C- 51)}$$

Gamma: Kovariansen til gamma går mot en stabil tilstand når tiden går mot uendelig slik at
$\dot{p}^a_\gamma(\infty) = \underline{0} = -\frac{2}{T^a} p^a_\gamma(\infty) + \widetilde{q}^a_\gamma =>$
$\frac{2}{T^a} p^a_\gamma(\infty) = \widetilde{q}^a_\gamma$
Der vi setter $p^a_\gamma(\infty) = p^a_\gamma(0)$ slik at

$$p^a_\gamma(0) = \underline{\sigma}^2_{\gamma^a} \quad \text{(C- 52)}$$
$$\underline{\widetilde{q}}^a_\gamma = \frac{2}{T^a} p^a_\gamma(0) \quad \text{(C- 53)}$$

Spektraltettheten kan da skrives slik

$$\tilde{Q}^a = \begin{bmatrix} I * \underline{\widetilde{q}}^a_\beta & \underline{0} \\ \underline{0} & I * \underline{\widetilde{q}}^a_\gamma \end{bmatrix} = \begin{bmatrix} P^a_\beta(1) & \underline{0} \\ \underline{0} & \frac{2}{T^a} P^a_\gamma(0) \end{bmatrix}$$

$$\tilde{Q}^a = \begin{bmatrix} I * \underline{\sigma}^2_{\gamma^a} & \underline{0} \\ \underline{0} & I * \frac{2}{T^a} \underline{\sigma}^2_{\gamma^a} \end{bmatrix} \quad \text{(C- 54)}$$

og initiel kovarians kan skrives

$$P^a(t_0) = \begin{bmatrix} \underline{0} & \underline{0} \\ \underline{0} & I * \underline{\sigma}^2_{\gamma^a} \end{bmatrix} \quad \text{(C- 55)}$$

C.1.3 Diskretisering

Diskretiseringen av denne prosessen gir

$$\Phi^a = e^{F^a \Delta t} \quad \text{(C- 56)}$$

Spektraltettheten kan diskretiseres slik

$$Q^a_\gamma = \frac{I * \underline{\widetilde{q}}^a_\gamma T^a}{2}(1 - e^{-\frac{2\Delta t}{T^a}}) \quad \text{(C- 57)}$$

$$Q^a_\beta = \Delta t * I * \underline{\widetilde{q}}^a_\beta \quad \text{(C- 58)}$$

$$\underline{q}_f = \Delta t * \underline{\widetilde{q}}_f = \Delta t * \underline{\sigma}^2_f \quad \text{(C- 59)}$$

$$Q^a = \begin{bmatrix} Q^a_\beta & \underline{0} \\ \underline{0} & Q^a_\gamma \end{bmatrix} \quad \text{(C- 60)}$$

De diskrete hvitstøyelementene bli da beskrevet slik

$\underline{v}^a_k \tilde{} N(0, Q^a)$

$\underline{v}^f_k \tilde{} N(0, \underline{q}_f)$

Når spektratetthetmatrisen blir diskretisert slik blir støymatrisen Γ lik G slik at $\Gamma^a = G^a$

$$\underline{x}^a_{k+1} = \Phi^a \underline{x}^a_k + \Gamma^a \underline{v}^a(t_k) \quad \text{(C- 61)}$$

$$P^a_{k+1} = \Phi^a P^a_k \Phi^{aT} + \Gamma^a Q^a \Gamma^{aT} \; der \; P^a_k = \Delta t * P^a(t_0) \quad \text{(C- 62)}$$

Der gamma og beta er gitt ved

$$\underline{\beta}^a_{k+1} = \underline{\beta}^a_k + \underline{v}^a_{\beta,k} \; og \; \underline{v}^a_{\beta,k} \tilde{} N(0, \underline{q}^a_\beta) \quad \text{(C- 63)}$$

$$\underline{\gamma}^a_{k+1} = e^{-\frac{T^a}{2}\Delta t} * \underline{\gamma}^a_k + \underline{v}^a_{\gamma,k} \; og \; \underline{v}^a_{\beta,k} \tilde{} N(0, \underline{q}^a_\gamma) \quad \text{(C- 64)}$$

Deretter adderes alle elementene samme til

$$\delta \underline{f}_k = \underline{\gamma}^a_k + \underline{\beta}^a_k + \underline{v}^f_k \quad \text{(C- 65)}$$

Gyrometerstøy

For gyrometerstøyen gjelder samme framgangsmåte. Men ved og bruke diskretiserings metoden beskrevet i innledingen (denne metoden blir brukt for alle diskretiseringer), blir koden mindre komplisert.

$\dot{\underline{\beta}}^g(t) = \underline{v}^g_\beta(t)$ der $\underline{v}^g_\beta(t) \tilde{} N(0, \underline{\sigma}^2_\beta)$

$\dot{\underline{\gamma}}^g(t) = \frac{1}{T^g}\underline{\gamma}^g(t) + \underline{v}_\gamma^g(t)$ der $\underline{\gamma}^g(t_0) = \underline{0}$ og $\underline{v}_\gamma^g(t) \tilde{} N(0, \underline{\sigma}_{\gamma^g}^2)$

$\underline{v}^\omega(t) \tilde{} N(0, \underline{\sigma}_\omega^2)$

Der korrelasjons tiden T^g er lik $T^g = 1$. Og variansen til hvitsøyelementene blir her satt til

$$\underline{\sigma}_{\beta^g}^2 = P_\beta^g(1) = (1\,\mathrm{deg}/s)^2 = (0.017453 rad/s)^2 \quad \text{(C- 66)}$$

$$\underline{\sigma}_{\gamma^g}^2 = P_\gamma^g(0) = (1\,\mathrm{deg}/s)^2 = (0.017453 rad/s)^2 \quad \text{(C- 67)}$$

$$\underline{\sigma}_\omega^2 = \underline{\widetilde{q}}_f = (1\,\mathrm{deg}/s)^2 = (0.017453 rad/s)^2 \quad \text{(C- 68)}$$

$$P^g(t_0) = \begin{bmatrix} \underline{0} & \underline{0} \\ \underline{0} & \underline{\sigma}_{\gamma^g}^2 \end{bmatrix}$$

$$\widetilde{Q}^g = \begin{bmatrix} \underline{\widetilde{q}}_\beta^g & \underline{0} \\ \underline{0} & \underline{\widetilde{q}}_\gamma^g \end{bmatrix} = \begin{bmatrix} P_\beta^g(1) & \underline{0} \\ \underline{0} & \frac{2}{T^g} P_\gamma^g(0) \end{bmatrix}$$

Hvis tilstandsvektoren og hvistøyvektoren skrives $\underline{x}^g = \begin{bmatrix} \underline{\beta}^g & \underline{\gamma}^g \end{bmatrix}^T$ og $\underline{v}^g = \begin{bmatrix} \underline{v}_\beta^g & \underline{v}_\gamma^g \end{bmatrix}^T$.

blir prosessmatrisen F^g og støymatrisen G^g lik

$$F^g = \begin{bmatrix} \underline{0} & \underline{0} \\ \underline{0} & \frac{\underline{I}}{T^g} \end{bmatrix} \quad \text{(C- 69)}$$

$$G^g = \begin{bmatrix} I & \underline{0} \\ \underline{0} & I \end{bmatrix} \quad \text{(C- 70)}$$

Systemmodellen kan da settes opp slik

$$\dot{\underline{x}}^g(t) = F^g \underline{x}^a(t) + G^g \underline{v}^a(t) \quad \text{(C- 71)}$$

$$\dot{P}^g(t) = F^g P^g(t) + P^g(t) F^{gT} + G^g \tilde{Q}^g G^{gT} \quad \text{(C- 72)}$$

Diskretisert

$$\underline{x}_{k+1}^g = \Phi^g \underline{x}_k^g + \Gamma^g \underline{v}^g(t_k) \quad \text{(C- 73)}$$

$$P_{k+1}^g = \Phi^g P_k^g \Phi^{gT} + \Gamma^g Q^g \Gamma^{gT} \; der \; Q^g = I \; og \; P_k^g = P^g(t_0) \quad \text{(C- 74)}$$

diskretiseringen av $\underline{v}^\omega(t)$ blir her gjort på samme måte som forrige gang som var og multiplisere variansen med samplingstiden Δt, slik at

$$\underline{q}_\omega = \Delta t * \underline{\widetilde{q}}_\omega = \Delta t * \underline{\sigma}_\omega^2 \quad \text{(C- 75)}$$

$$\underline{v}_k^\omega \tilde{} N(0, \underline{q}_\omega) \quad \text{(C- 76)}$$

Deretter adderes alle elementene samme til

$$\delta\underline{\omega}_k = \underline{\gamma}_k^g + \underline{\beta}_k^g + \underline{v}_k^\omega \quad \text{(C- 77)}$$

C.1.4 Pesudokode

$I = eye(3);$

$N = 9 * 200;$

$T^a = 2;$

$T^g = 1;$

$\Delta t = \frac{1}{100};$

$\sigma_a^2 = (0.00981 \frac{m/s^2}{s})^2$

$\sigma_g^2 = (0.017453 \frac{rad}{s})^2$

$P_\gamma^a(0) = P_\beta^a(1) = diag\left(\left[\begin{array}{ccc} \sigma_a^2 & \sigma_a^2 & \sigma_a^2 \end{array}\right]\right);$

$P_\gamma^g(0) = P_\beta^g(1) = diag\left(\left[\begin{array}{ccc} \sigma_g^2 & \sigma_g^2 & \sigma_g^2 \end{array}\right]\right);$

$\widetilde{Q}^g = \left[\begin{array}{cc} P_\beta^g(1) & \underline{0} \\ \underline{0} & \frac{2}{T^g} P_\gamma^g(0) \end{array}\right];$

$\widetilde{Q}^a = \left[\begin{array}{cc} P_\beta^a(1) & \underline{0} \\ \underline{0} & \frac{2}{T^a} P_\gamma^a(0) \end{array}\right];$

$P_\beta^g(0) = P_\beta^a(0) = \underline{0};$

$F^g = \left[\begin{array}{cc} \underline{0} & \underline{0} \\ \underline{0} & \frac{I}{T^g} \end{array}\right];$

$F^a = \left[\begin{array}{cc} \underline{0} & \underline{0} \\ \underline{0} & \frac{I}{T^a} \end{array}\right];$

$G = G^g = G^a = I;$

$\underline{x}^g(0) = \left[\begin{array}{cc} \underline{\beta}^g(0) & \underline{\gamma}^g(0) \end{array}\right]^T = \left[\begin{array}{cc} \underline{0} & chol(P_\gamma^a(0), lower) * \mathrm{rand}(3,1) \end{array}\right]^T;$

$\underline{x}^a(0) = \left[\begin{array}{cc} \underline{\beta}^a(0) & \underline{\gamma}^a(0) \end{array}\right]^T = \left[\begin{array}{cc} \underline{0} & chol(P_\gamma^a(0), lower) * \mathrm{rand}(3,1) \end{array}\right]^T;$

$\left[\begin{array}{cc} \Phi^g & \Gamma^g \end{array}\right] = diskretisering(F^g, G, \Delta t, \widetilde{Q}^g);$

$\left[\begin{array}{cc} \Phi^a & \Gamma^a \end{array}\right] = diskretisering(F^a, G, \Delta t, \widetilde{Q}^a);$

$P_1^g = \Delta t * \left[\begin{array}{cc} P_\beta^g(0) & \underline{0} \\ \underline{0} & P_\gamma^g(0) \end{array}\right];$

$P_1^a = \Delta t * \left[\begin{array}{cc} P_\beta^a(0) & \underline{0} \\ \underline{0} & P_\gamma^a(0) \end{array}\right];$

$for \qquad k = 1 : N - 1$

$\{$

$\quad \underline{x}_{k+1}^g = \Phi^g \underline{x}_k^g + \Gamma^g * \mathrm{rand}(6,1);$

$\quad P_{k+1}^g = \Phi^g P_k^g \Phi^{gT} + \Gamma^g * \Gamma^{gT};$

$\quad \underline{x}_{k+1}^a = \Phi^a \underline{x}_k^a + \Gamma^a * \mathrm{rand}(6,1);$

$\quad P_{k+1}^a = \Phi^a P_k^a \Phi^{aT} + \Gamma^a * \Gamma^{aT};$

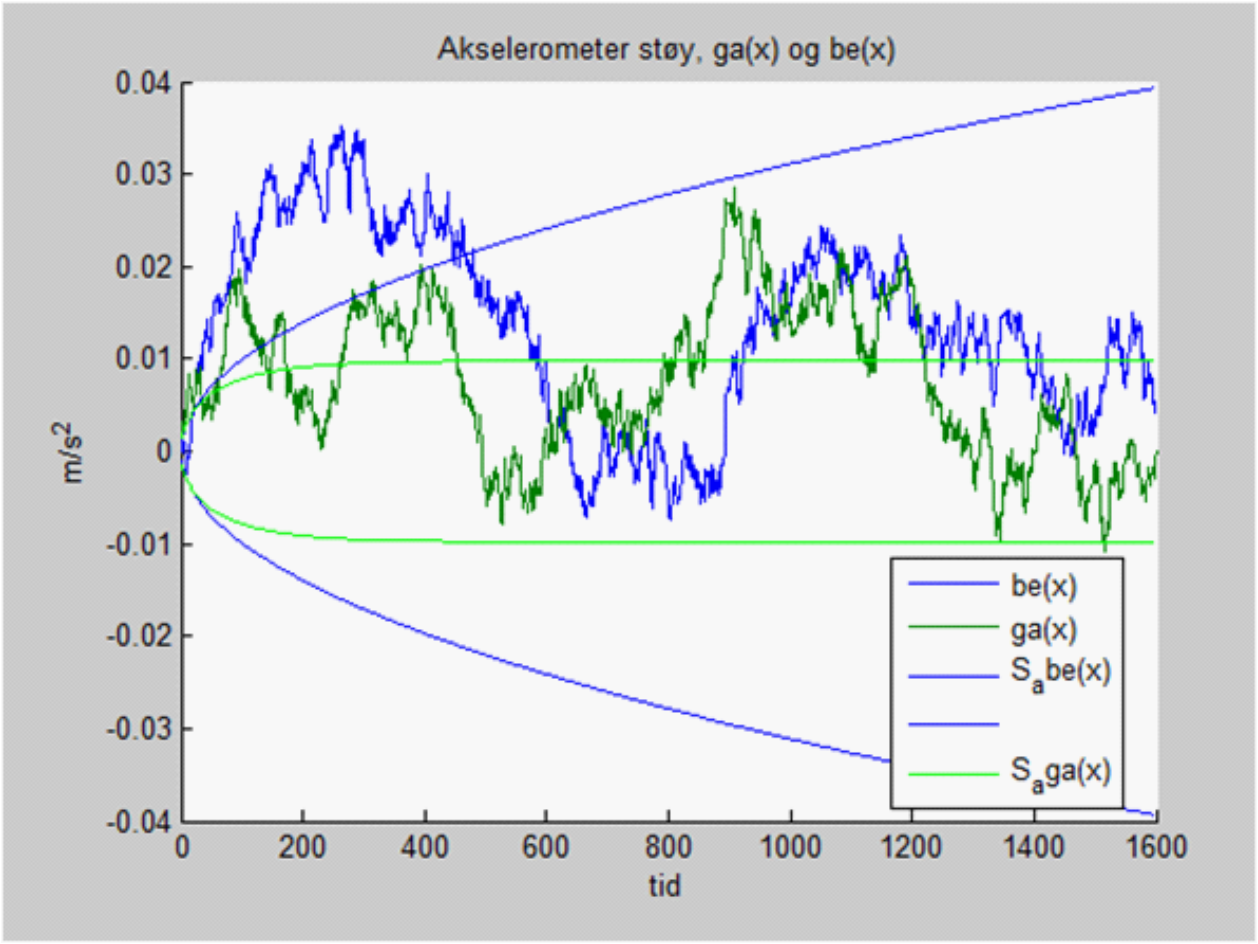
$\};$

$\delta \underline{f}_k = \underline{x}_k^a(4:6) + \underline{x}_k^a(1:3) + \sigma_a^2 * \Delta t * \mathrm{rand}(3,1);$

$\delta \underline{\omega}_k = \underline{x}_k^g(4:6) + \underline{x}_k^g(1:3) + \sigma_g^2 * \Delta t * \mathrm{rand}(3,1);$

C.2 Resultat

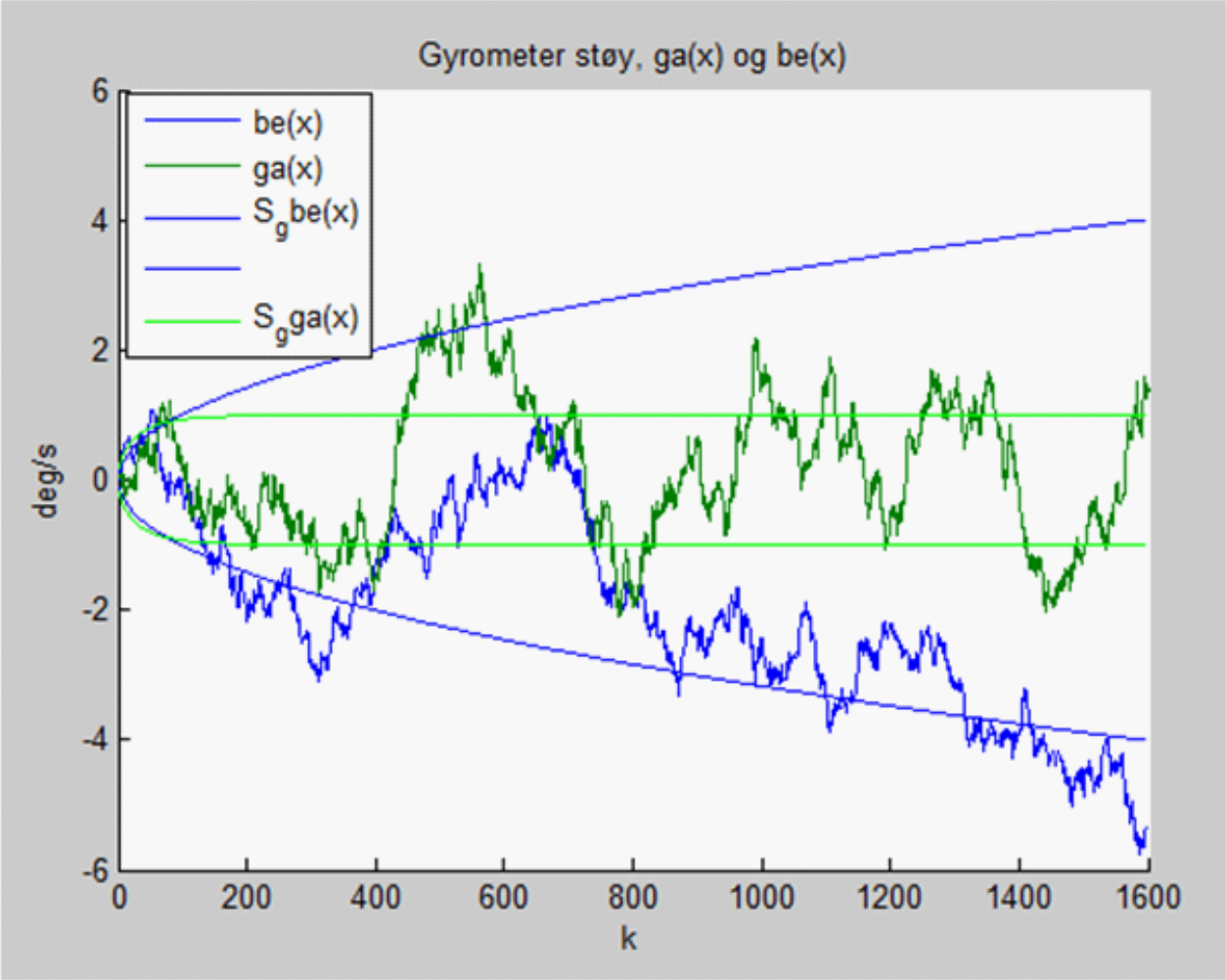
Forventet resultat fra støymodelleringen er å ha gamma og beta verdier innen for estimert standard avvik ca 1/3 av tiden. Figuren under viser støyen akselerometerstøy elementene gamma og beta i x-akse sensoren.



Modellering av gamma og beta for akselerometerstøyen.

Det som kan ses her er at over tid kan potensielt random walk prosessen (beta) vandre et stykke unna forventningsverdien. Mens Gauss-Markov prosessen (gamma) potensielt ikke vil være like dominerende, vil den fortsatt gi et solid bidrag til usikkerheten rundt forventningsverdien.

For gyrometerstøyen forventes det samme, men her burde støyen ligge rundt 1 deg/s. Resultatet er cirka det samme som for akselerometerstøyen, det kan ses i figuren under. Der ser også at støyen ligger rundt 1 deg/s.



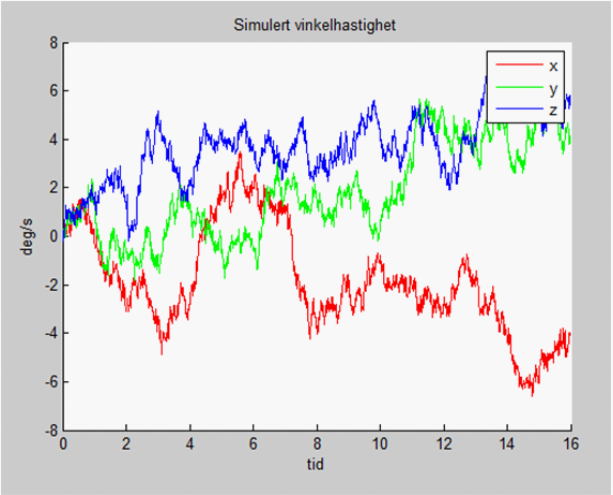
Modellering av gamma og beta for gyrometerstøyen.

Selve akselerometer og gyrometer målingene er gitt av

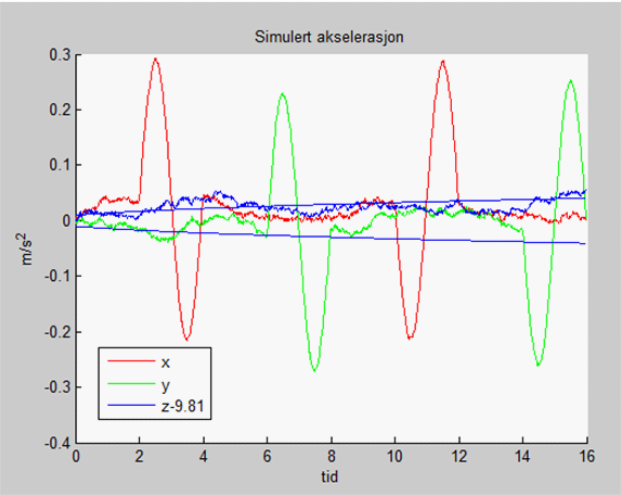
$$\underline{\tilde{f}}^b = \underline{f}^b - \delta\underline{f} \quad \text{(C- 78)}$$

$$\underline{\tilde{\omega}}_n^{nb} = \underline{\omega}_n^{nb} - \delta\underline{\omega} \quad \text{(C- 79)}$$

Når de blir addert sammen, blir resultatet som i figurene under. Der ser man at gjennomsnittet av støyen faktisk ligger litt over 1 deg/s og 1 mG/s. Noe som vil føre til litt større drift i systemet en antatt.



Gyrometermålinger



Akselerometermålinger

Del D: Treghetsnavigasjons Systemer

Treghetsnavigasjon er en posisjonerings metode der man over tid integrerer opp akselerasjon og vinkelhastighet for å bestemme posisjon, hastighet og orientering (Roll, Pitch, Yaw). Et TNS er brukt i systemer der man ikke kan/vil bruke eksterne kilder til posisjonering, som for eksempel undervannsfartøyer eller satteliter eller det blir brukt til og forbedre nøyaktigheten til eksisterende målinger som for eksempel GPS målinger. Problemet med TNS er at driften er en funksjon av t så hvis ikke sensorene er veldig nøyaktige vil driften bli betydelig over kort tid.

D.1 Eulers metode

Eulers metode for treghets navigasjon er som følger.

$$\frac{d}{dt}\widetilde{p}^n(t) = \widetilde{v}^n(t) \quad \text{(D- 80)}$$

$$\frac{d}{dt}\widetilde{v}^n(t) = R_n^b(t) * \widetilde{f}^b(t) - \underline{g}^n \quad \text{(D- 81)}$$

$$\frac{d}{dt}\widetilde{R}_n^b(t) = \widetilde{R}_n^b(t) * S(\widetilde{\underline{\omega}}_b^{nb}) \quad \text{(D- 82)}$$

diskretisert gir dette

$$\widetilde{p}_{k+1}^n = \widetilde{p}_k^n + \widetilde{v}_k^n * \Delta t \quad \text{(D- 83)}$$

$$\widetilde{v}_{k+1}^n = \widetilde{v}_k^n + R_{n,k}^b * \widetilde{f}^b(t) * \Delta t - \overline{g}^n(t) * \Delta t \quad \text{(D- 84)}$$

$$\widetilde{R}_{n,k+1}^b = \widetilde{R}_{n,k}^b + \widetilde{R}_{n,k}^b * S(\widetilde{\underline{\omega}}_{b,k}^{nb} * \Delta t) \quad \text{(D- 85)}$$

D.2 Eulers-Heuns metode

Eulers-Heuns metode for treghets navigasjon gir et mer nøyaktig resultat, Euler-Heuns kan utrykkes ved å først skrive Euler likningene på standardform, som blir gjort på følgende måte:

$$\dot{\underline{y}} = f(\underline{u}(t), \underline{y}(t), \widetilde{R}_n^b(t)) \quad \text{(D- 86)}$$

$$\frac{d}{dt}\widetilde{R}_n^b(t) = f^R(\underline{u}(t), \widetilde{R}_n^b(t)) \quad \text{(D- 87)}$$

Når $\underline{u}(t) = \begin{bmatrix} \widetilde{f}^b(t) \\ \widetilde{\underline{\omega}}_b^{bb}(t) \end{bmatrix}$ *og* $\underline{y}(t) = \begin{bmatrix} \widetilde{p}^n(t) \\ \widetilde{v}^n(t) \end{bmatrix}$ vektorene blir definert slik, kan systemmatrisen skrives slik.

$$f(\underline{u}(t), \underline{y}(t), \widetilde{R}_n^b(t)) = \begin{bmatrix} \widetilde{v}^n(t) \\ R_n^b(t) * \widetilde{f}^b(t) - \underline{g}^n \end{bmatrix} \quad \text{(D- 88)}$$

$$f^R(\underline{u}(t), \widetilde{R}_n^b(t)) = \begin{bmatrix} \widetilde{R}_n^b(t) * S(\widetilde{\underline{\omega}}_b^{nb}) \end{bmatrix} \quad \text{(D- 89)}$$

Euler-Heuns kan da utrykkes diskret slik

$$\widetilde{R}^m_{k+1} = \widetilde{R}^b_{n,k} + f^R(\underline{u}_k, \widetilde{R}^b_{n,k})\Delta t \quad \text{(D- 90)}$$

$$\underline{y}^m_{k+1} = y_k + f(\underline{u}_k, \underline{y}_k, \widetilde{R}^b_{n,k})\Delta t \quad \text{(D- 91)}$$

$$\widetilde{R}^b_{n,k+1} = \widetilde{R}^b_{n,k} + (f^R(\underline{u}_k, \widetilde{R}^b_{n,k}) + f^R(\underline{u}_{k+1}, \widetilde{R}^m_{k+1}))\frac{\Delta t}{2} \quad \text{(D- 92)}$$

$$\underline{y}_{k+1} = y_k + (f(\underline{u}_k, \underline{y}_k, \widetilde{R}^b_{n,k}) + f(\underline{u}_{k+1}, \underline{y}^m_{k+1}, \widetilde{R}^m_{k+1}))\frac{\Delta t}{2} \quad \text{(D- 93)}$$

som gir

$$R^m_{k+1} = \widetilde{R}^b_{n,k} + \widetilde{R}^b_{n,k} * S(\widetilde{\omega}^{nb}_{b,k} * \Delta t) \quad \text{(D- 94)}$$

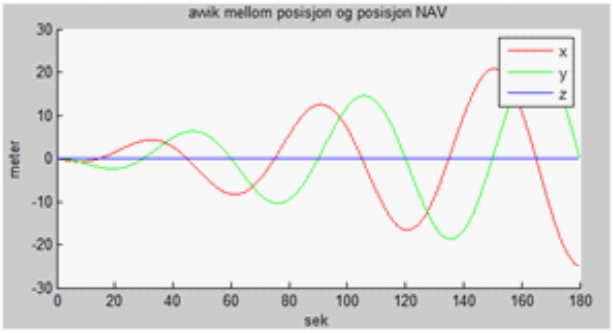
$$v^m_{k+1} = \widetilde{p}^n_k + \widetilde{v}^n_k * \Delta t \quad \text{(D- 95)}$$

$$\widetilde{R}^b_{n,k+1} = \widetilde{R}^b_{n,k} + (\widetilde{R}^b_{n,k} * S(\widetilde{\omega}^{nb}_{b,k}) + R^m_{k+1} * S(\widetilde{\omega}^{nb}_{b,k+1}))\frac{\Delta t}{2} \quad \text{(D- 96)}$$

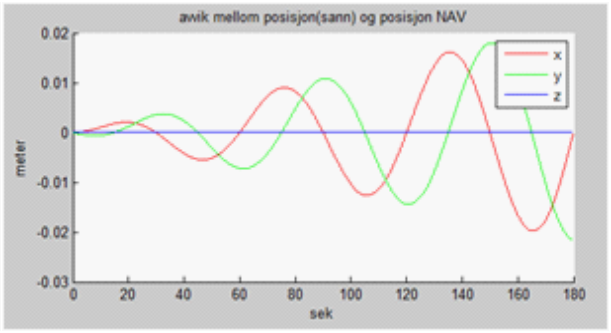
$$\widetilde{p}^n_{k+1} = \widetilde{p}^n_k + (\widetilde{v}^n_k + v^m_{k+1})\frac{\Delta t}{2} \quad \text{(D- 97)}$$

$$\widetilde{v}^n_{k+1} = \widetilde{v}^n_k + (R^b_{n,k} * \widetilde{f}^b_k + R^m_{k+1}\widetilde{f}^b_{k+1})\frac{\Delta t}{2} - \overline{g}^n \Delta t \quad \text{(D- 98)}$$

Deterministiske simuleringer av Euler og Euler-Heuns likningene viser at de gir en integreringsfeil over tid, der Euler metodens integreringfeil synker linært mens samplingsfrekvensen øker, synker Euler-Heuns metodens integreringsfeilen kvadratisk mens samplingsfrekvensen øker. I praksis blir integreringsfeilen til Euler likningene er ca. 1000 ganger større en Euler-Heuns likningene, der det simuleres på en bane som gir konstant bevegelse over 180 sekunder og en samplingsfrekvens på 40Hz. Det er da klart at simuleringer med Euler-Heuns metoden er og foretrekke.



Deterministisk Euler Nav



Deterministisk Euler-Heuns Nav

Del E: Kalmanfilter

Et Kalmanfilter er en rekursiv estimator som på bakgrunn av kunnskap om prosessmodell og initialverdier gir et forbedret estimat av tilstand og feilkovarians på en måling med tilfeldig støy(, en stokastisk måling). Den estimerte tilstanden Kalmanfilteret gir er ofte nærmer sannverdi fordi den har en bedre estimert usikkerhet enn målt tilstand. Filteret er rekursivt og er bare avhengig av forrige tidskrit, det gjør det til et veldig nyttig verktøy til tilstandsestimering og signalbehandling. For å gjøre dette er første skritt og prediktere tilstanden og feilkovariansen, deretter beregnes en minimums gjennomsnitts sum til alle lineære kombinasjoner av tilsdandsfeilene ("least mean square estimation") også korrigeres den prediktert tilstanden for gitt minimum.

E.0.1 KF

Den mattematiskemodellen for et lineær kontinuerlig-diskret system uttrykt slik

$$\dot{\underline{x}}(t) = F(t)\underline{x}(t) + L(t)\underline{u}(t) + G(t)\underline{v}(t) \quad \text{(E- 99)}$$
$$\underline{z}_k = H_k\underline{x}_k + \underline{w}_k \quad \text{(E- 100)}$$

De kontinuerlig-diskrete Kalmanfilterlikningen har formen

Tidsoppdatering, TO, er prediksjons fasen der likningene er gitt av:

$$\frac{d}{dt}\bar{\underline{x}}(t) = F(t)\underline{x}(t) + L(t)\underline{u}(t) \quad \text{(E- 101)}$$
$$\frac{d}{dt}\bar{P}(t) = F(t)P(t) + P(t)F^T(t) + G(t)QG^T(t) \quad \text{(E- 102)}$$

der $\bar{P}(t_k) = \hat{P}_k$ *og* $\bar{P}(t_0)$ *er gitt.*

Måleoppdatering, MO, er korrigeringsfasen der eventuelle predikterings feil blir korrigert:

$$\tilde{\underline{z}}_k = H_k\widetilde{\underline{x}}_k + \underline{w}_k \quad \text{(E- 103)}$$
$$K_k = \bar{P}_k H^T(H\bar{P}_k H^T - Rd)^{-1} \qquad , \bar{P}_k = \bar{P}(t_k) \quad \text{(E- 104)}$$
$$\hat{\underline{x}}_k = \bar{\underline{x}}_k + K_k(\tilde{\underline{z}}_k - H\bar{\underline{x}}_k) \quad \text{(E- 105)}$$
$$\hat{P}_k = (I - K_k H)\bar{P}_k \quad \text{(E- 106)}$$

E.0.2 LKF

Lineærisert Kalmanfilter er en lineærisert variant av Kalmanfilteret. Det beregner feilen mellom sann og målt verdi.

De mattematiskelikningene for en lineærisert kontinuerlig-diskret systemmodell uttrykt slik

$$\delta\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t), \underline{u}(t)) + G(t)\underline{v}(t) \quad \text{(E- 107)}$$
$$\delta\dot{\underline{x}}(t) = \underline{x}(t) - \widetilde{\underline{x}}(t) \quad \text{(E- 108)}$$
$$\underline{z}_k = H_k\underline{x}_k \quad \text{(E- 109)}$$
$$\widetilde{\underline{z}}_k = H_k\widetilde{\underline{x}}_k + \underline{w}_k \quad \text{(E- 110)}$$

De kontinuerlig-diskrete Kalmanfilterlikningen for da formen

Tidsoppdatering TO

$$\delta\dot{\underline{x}}(t) = F(t)\delta\underline{x}(t) \quad \text{(E- 111)}$$
$$P(t) = F(t)P(t) + P(t)F^T(t) + G(t)QG^T(t) \quad \text{(E- 112)}$$

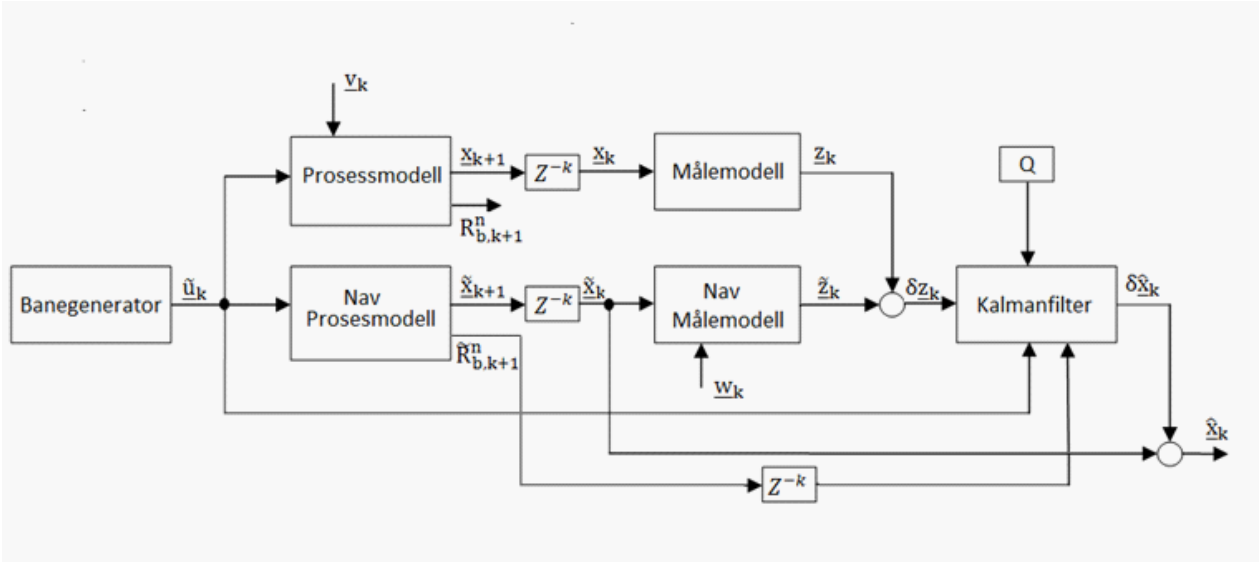
der $\bar{P}(t_k) = \hat{P}_k$ *og* $\bar{P}(t_0)$ *er gitt.*

Måleoppdatering MO

$$\delta\underline{z}_k = \underline{z}_k - \widetilde{\underline{z}}_k \quad \text{(E- 113)}$$
$$K_k = \bar{P}_k H^T (H\bar{P}_k H^T - Rd)^{-1},\ \bar{P}_k = \bar{P}(t_k) \quad \text{(E- 114)}$$
$$\delta\hat{\underline{x}}_k = \delta\bar{\underline{x}}_k + K_k(\delta\tilde{\underline{z}}_k - H\delta\bar{\underline{x}}_k) \quad \text{(E- 115)}$$
$$\hat{\underline{x}}_k = \widetilde{\underline{x}}_k + \delta\hat{\underline{x}}_k \quad \text{(E- 116)}$$
$$\hat{P}_k = (I - K_k H)\bar{P}_k \quad \text{(E- 117)}$$

E.1 Filtermodell

Blokkskjemaet til navigasjonssystemet er gitt ved.



Bloksjema, systemmodell uten tilbakekobling

Lineriseringen av navigasjonssystemetmodellen blir gjort fordi et Kalmanfilter er egnetlig en linær opperator, og et navigasjonssystem er per definisjon ulineært. Dette blir gjort som illustrert i blokkskjema for en systemmodell uten tilbakekobling Der det estimeres avvik fra den drift i systemet og det ukjente er sann tilstand. Det løses ved å ta måleoppdatering når man vet hva sann hastigheten eller posisjonen er. Sann posisjon er kjent ved start eller når man får en ekstern posisjons avlesning. Sann hastighet er kjent, når man vet at systemet står stille, eller når det blir gitt en ekstern hastighets måling. For å få til det trengs en sann prosessmodell, en nav-prosessmmodell og tilhørende målemodeller.

E.2 Nav Prosessmodell

Nav prosessmodellen er Euler navigasjonslikningene skrevet på standardform.

$$\frac{d}{dt}\widetilde{\underline{p}}^n(t) = \widetilde{\underline{v}}^n(t) \quad \text{(E- 118)}$$

$$\frac{d}{dt}\widetilde{\underline{v}}^n(t) = \widehat{R}_b^n(t)(\widetilde{\underline{f}}^b(t)) - \underline{g}^n \quad \text{(E- 119)}$$

$$\frac{d}{dt}\widetilde{R}_b^n(t) = \widehat{R}_b^n(t)S(\widetilde{\underline{\omega}}_b^{nb}(t)) \quad \text{(E- 120)}$$

Der det er ønskelig og utrykke det slik

F_{KP}^{US} $\quad \frac{d}{dt}\widetilde{\underline{x}}(t) = f(\widetilde{\underline{x}}, \widetilde{\underline{u}}, \widetilde{R}_b^n)$

$\quad \frac{d}{dt}\widetilde{R}_b^n(t) = f_R(\widetilde{\underline{u}}, \widetilde{R}_b^n)$

F_{DP}^{US} $\quad \widetilde{\underline{z}} = H\widetilde{\underline{x}}$

Tilstandsvektoren og pådragsvektoren blir her

$$\widetilde{\underline{x}} = \begin{bmatrix} \widetilde{\underline{p}}^n & \widetilde{\underline{v}}^n \end{bmatrix}^T \qquad \widetilde{\underline{u}} = \begin{bmatrix} \widetilde{\underline{f}}^b & \widetilde{\underline{\omega}}_b^{nb} \end{bmatrix}^T$$

Dermed kan man utrykke systemet på standardform slik

$$f(\widetilde{\underline{x}}, \widetilde{\underline{u}}, \widetilde{R}_b^n) = \begin{bmatrix} \widetilde{\underline{v}}^n \\ R_b^n(\widetilde{\underline{f}}^b) - \underline{g}^n \end{bmatrix} \quad \text{(E- 121)}$$

$$f_R(\widetilde{\underline{u}}, \widetilde{R}_b^n) = \left[\widehat{R}_b^n(t)S(\widetilde{\underline{\omega}}_b^{nb}(t))\right] \quad \text{(E- 122)}$$

E.2.1 Sann-prosessmmodell

Likningene til den sanne prosessmodell beskriver hvordan èn matematisk kan finne sann posisjon, hastighet og orientering, uti fra Euler-navigasjons likningene hvis de sanne hvitstøyverdiene hadde vært kjent.

$$\dot{\underline{p}}^n(t) = \underline{v}^n \quad \text{(E- 123)}$$

$$\dot{\underline{v}}^n(t) = R_b^n(\widetilde{\underline{f}}^b + \delta\underline{f}) - \underline{g}^n = R_b^n(\widetilde{\underline{f}}^b - \underline{\gamma}^a - \underline{\beta}^a - \underline{v}^f(t)) - \underline{g}^n \quad \text{(E- 124)}$$

$$\dot{\underline{\gamma}}^a(t) = -\frac{1}{T^a}\underline{\gamma}^a(t) + \underline{v}_\gamma^a(t) \quad \text{(E- 125)}$$

$$\dot{\underline{\gamma}}^g(t) = -\frac{1}{T^g}\underline{\gamma}^g(t) + \underline{v}_\gamma^g(t) \quad \text{(E- 126)}$$

$$\dot{\underline{\beta}}^a(t) = \underline{v}_\beta^a(t) \quad \text{(E- 127)}$$

$$\dot{\underline{\beta}}^g(t) = \underline{v}_\beta^g(t) \quad \text{(E- 128)}$$

$$\dot{R}_b^n(t) = R_b^n * S(\widetilde{\underline{\omega}}_b^{nb} + \delta\underline{\omega}) = R_b^n * S(\widetilde{\underline{\omega}}_b^{nb} - \underline{\gamma}^g - \underline{\beta}^g - \underline{v}^\omega(t)) \quad \text{(E- 129)}$$

Der det er ønskelig og definere systemmodellen slik

S_{KP}^{US} $\quad \dot{\underline{x}} = f^*(\underline{x}, \underline{u}, R_b^n) + G^*(R_b^n)\underline{v}$

$\quad \dot{R}_b^n(t) = f_R(\underline{x}, \underline{u}, R_b^n, \underline{v})$

S_{DP}^{US} $\quad \underline{z}_k = H^*\underline{x}_k$

hvis tilstands vektoren, pådragsvektoren og støyvektoren defineres slik

$$\underline{x} = \begin{bmatrix} \underline{p}^n & \underline{v}^n & \underline{\gamma}^a & \underline{\gamma}^g & \underline{\beta}^a & \underline{\beta}^g \end{bmatrix}^T \quad \text{(E- 130)}$$

$$\underline{v} = \begin{bmatrix} \underline{v}^f & \underline{v}^\omega & \underline{v}^a_\gamma & \underline{v}^g_\gamma & \underline{v}^a_\beta & \underline{v}^g_\beta \end{bmatrix}^T \quad \text{(E- 131)}$$

$$\widetilde{\underline{u}} = \begin{bmatrix} \widetilde{\underline{f}}^b & \widetilde{\underline{\omega}}^{nb}_b \end{bmatrix}^T \quad \text{(E- 132)}$$

vil systemet utryket på standardform bli slik

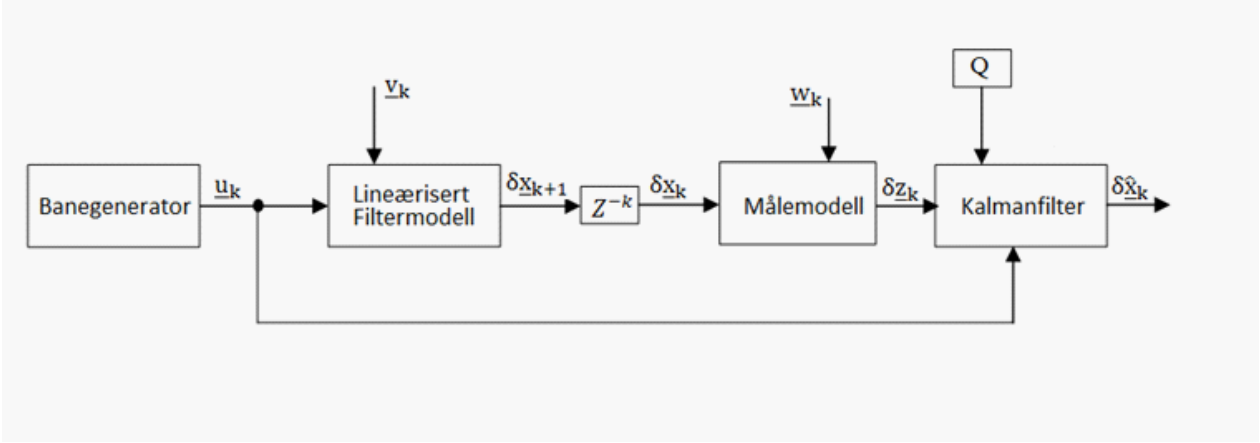
$$f^*(\underline{x}, \underline{u}, R^n_b) = \begin{bmatrix} \underline{v}^n \\ R^n_b(\widetilde{\underline{f}}^b - \underline{\gamma}^a - \underline{\beta}^a) - \underline{g}^n \\ -\frac{1}{T^a}\underline{\gamma}^a(t) \\ -\frac{1}{T^g}\underline{\gamma}^g(t) \\ \underline{0} \\ \underline{0} \end{bmatrix}$$

$$G^*(R^n_b) = \begin{bmatrix} \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & -R^n_b & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & I & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & I & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & I & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & I \end{bmatrix}$$

$$f^*_R(\underline{x}, \underline{u}, R^n_b, \underline{v}) = \begin{bmatrix} R^n_b * S(\widetilde{\underline{\omega}}^{nb}_b - \underline{\gamma}^g - \underline{\beta}^g - \underline{v}^\omega(t)) \end{bmatrix}$$

E.2.2 Filtermodell

Filtermodellen utledes ved å trekke sann tilstand fra beregnet tilstand. $\delta\underline{x} = \underline{x} - \widetilde{\underline{x}}$. Blokkskjemaet blir da slik som i figur sett fra Kalmanfilteret.



Blokkskjema av filtermodell

$$\delta\dot{\underline{p}} = \tfrac{d}{dt}\underline{p}^n - \tfrac{d}{dt}\widetilde{\underline{p}}^n = \underline{v}^n - \widetilde{\underline{v}}^n = \delta\underline{v}$$

$$\begin{aligned}\delta\dot{\underline{v}} &= \tfrac{d}{dt}\underline{v}^n - \tfrac{d}{dt}\widetilde{\underline{v}}^n = R^n_b(\widetilde{\underline{f}}^b + \delta f) - \underline{g}^n - (\widetilde{R}^n_b\widetilde{\underline{f}}^b + \underline{g}^n) \\ &= R^n_b\widetilde{\underline{f}}^b + R^n_b\delta f - \widetilde{R}^n_b\widetilde{\underline{f}}^b \\ &= R(\underline{\varepsilon})\widetilde{R}^n_b\widetilde{\underline{f}}^b + R(\underline{\varepsilon})\widetilde{R}^n_b\delta f - \widetilde{R}^n_b\widetilde{\underline{f}}^b \\ &= (I + S(\underline{\varepsilon}))\widetilde{R}^n_b\widetilde{\underline{f}}^b - \widetilde{R}^n_b\widetilde{\underline{f}}^b + (I + S(\underline{\varepsilon}))\widetilde{R}^n_b\delta f\end{aligned}$$

$$= (I + S(\underline{\varepsilon}) - I)\widetilde{R}_b^n \widetilde{\underline{f}}^b + (I + S(\underline{\varepsilon}))\widetilde{R}_b^n \delta f$$
$$= S(\underline{\varepsilon})\widetilde{R}_b^n \widetilde{\underline{f}}^b + \widetilde{R}_b^n \delta f + S(\underline{\varepsilon})\widetilde{R}_b^n \delta f$$
$$= -S(\widetilde{R}_b^n \widetilde{\underline{f}}^b)\underline{\varepsilon} + \widetilde{R}_b^n \delta f - S(\widetilde{R}_b^n \delta f)\underline{\varepsilon}$$

Det siste leddet i $-S(\widetilde{R}_b^n \delta f)\underline{\varepsilon}$ likningen $\delta\underline{\dot{v}}$, blir feilen kvadrert, som når feilen er liten blir ett veldig lite tall. Det antas derfor at $-S(\widetilde{R}_b^n \delta f)\underline{\varepsilon}$ går mot null og det kan ses bort i fra. Slik at

$\delta\underline{\dot{v}} = -S(\widetilde{R}_b^n \widetilde{\underline{f}}^b)\underline{\varepsilon} + \widetilde{R}_b^n \delta f$, der $\delta f = -\underline{\gamma}^a - \underline{\beta}^a - \underline{v}^f$

$\frac{d}{dt}R_b^n - \frac{d}{dt}\widetilde{R}_b^n = R_b^n S(\widetilde{\underline{\omega}}_b^{nb} + \delta\underline{\omega}) - \widetilde{R}_b^n S(\widetilde{\underline{\omega}}_b^{nb})$
$\frac{d}{dt}(R(\underline{\varepsilon})\widetilde{R}_b^n) - \frac{d}{dt}\widetilde{R}_b^n = R(\underline{\varepsilon})\widetilde{R}_b^n(S(\widetilde{\underline{\omega}}_b^{nb}) + S(\delta\underline{\omega})) - \widetilde{R}_b^n S(\widetilde{\underline{\omega}}_b^{nb})$
$S(\frac{d}{dt}\underline{\varepsilon})\widetilde{R}_b^n + (I + S(\underline{\varepsilon}))\frac{d}{dt}\widetilde{R}_b^n - \frac{d}{dt}\widetilde{R}_b^n = (I + S(\underline{\varepsilon}))\widetilde{R}_b^n(S(\widetilde{\underline{\omega}}_b^{nb}) + S(\delta\underline{\omega})) - \widetilde{R}_b^n S(\widetilde{\underline{\omega}}_b^{nb})$
$S(\frac{d}{dt}\underline{\varepsilon})\widetilde{R}_b^n + \frac{d}{dt}\widetilde{R}_b^n + S(\underline{\varepsilon})\frac{d}{dt}\widetilde{R}_b^n - \frac{d}{dt}\widetilde{R}_b^n = (\widetilde{R}_b^n + S(\underline{\varepsilon})\widetilde{R}_b^n)(S(\widetilde{\underline{\omega}}_b^{nb}) + S(\delta\underline{\omega})) - \widetilde{R}_b^n S(\widetilde{\underline{\omega}}_b^{nb})$
$S(\frac{d}{dt}\underline{\varepsilon})\widetilde{R}_b^n + S(\underline{\varepsilon})\widetilde{R}_b^n S(\widetilde{\underline{\omega}}_b^{nb}) = \widetilde{R}_b^n S(\widetilde{\underline{\omega}}_b^{nb}) + \widetilde{R}_b^n S(\delta\underline{\omega}) + S(\underline{\varepsilon})\widetilde{R}_b^n S(\widetilde{\underline{\omega}}_b^{nb}) + S(\underline{\varepsilon})\widetilde{R}_b^n S(\delta\underline{\omega}) - \widetilde{R}_b^n S(\widetilde{\underline{\omega}}_b^{nb})$
$S(\frac{d}{dt}\underline{\varepsilon})\widetilde{R}_b^n = \widetilde{R}_b^n S(\delta\underline{\omega}) + S(\underline{\varepsilon})\widetilde{R}_b^n S(\delta\underline{\omega})$

I det siste leddet i $S(\underline{\varepsilon})\widetilde{R}_b^n S(\delta\underline{\omega})$ likningen, blir feilen også kvadrert, når feilen er liten blir det ett veldig lite tall. Det antas derfor at $S(\underline{\varepsilon})\widetilde{R}_b^n S(\delta\underline{\omega})$ går mot null og det kan ses bort i fra. Slik at

$S(\frac{d}{dt}\underline{\varepsilon})\widetilde{R}_b^n = \widetilde{R}_b^n S(\delta\underline{\omega})$
$S(\frac{d}{dt}\underline{\varepsilon}) = \widetilde{R}_b^n S(\delta\underline{\omega})\widetilde{R}_n^b$
$\frac{d}{dt}\underline{\varepsilon} = \widetilde{R}_b^n \delta\underline{\omega}$[1], der $\delta\underline{\omega} = -\underline{\gamma}^g - \underline{\beta}^g - \underline{v}^\omega$

Dermed kan feil likningene utrykes slik

$$\frac{d}{dt}\delta\underline{p}(t) = \delta\underline{v}(t) \quad \text{(E- 133)}$$
$$\frac{d}{dt}\delta\underline{v}(t) = -S(R_b^n(t) * \widetilde{\underline{f}}^b(t)) * \underline{\varepsilon}(t) + R_b^n(t) * (-\underline{\gamma}^a(t) - \underline{\beta}^a(t) - \underline{v}^f(t)) \quad \text{(E- 134)}$$
$$\frac{d}{dt}\underline{\varepsilon}(t) = R_b^n(t)(-\underline{\gamma}^g(t) - \underline{\beta}^g(t) - \underline{v}^\omega(t)) \quad \text{(E- 135)}$$
$$\frac{d}{dt}\underline{\gamma}^a(t) = -\frac{1}{T^a}\underline{\gamma}^a(t) + \underline{v}_\gamma^a(t) \quad \text{(E- 136)}$$
$$\frac{d}{dt}\underline{\gamma}^g(t) = -\frac{1}{T^g}\underline{\gamma}^g(t) + \underline{v}_\gamma^g(t) \quad \text{(E- 137)}$$
$$\frac{d}{dt}\underline{\beta}^a(t) = \underline{v}_\beta^a(t) \quad \text{(E- 138)}$$
$$\frac{d}{dt}\underline{\beta}^g(t) = \underline{v}_\beta^g(t) \quad \text{(E- 139)}$$

Det er ønskelig og ha filtermodellen uttrykt på denne formen

K_{KP}^{US} $\quad \delta\underline{\dot{x}} = F(\underline{u}, \widetilde{R}_b^n)\delta\underline{x} + G(\widetilde{R}_b^n)\underline{v}$,

K_{DP}^{US} $\quad \delta\widetilde{\underline{z}} = \underline{z} - \widetilde{\underline{z}} + \underline{w}$ Siden z alltid er $\underline{0}$ blir $\delta\widetilde{\underline{z}} = -\widetilde{\underline{z}} + \underline{w}$

[1] TNS med lavkostsensorer, Diego Mugisha, 2012

Med tilstandsvektor, pådragsvektor og støyvektor deffinert slik

$$\delta\underline{x} = \left[\begin{array}{cccccccc} \delta\underline{p}^n & \delta\underline{v}^n & \underline{\varepsilon} & \underline{\gamma}^a & \underline{\gamma}^g & \underline{\beta}^a & \underline{\beta}^g \end{array}\right]^T \quad \text{(E- 140)}$$

$$\widetilde{\underline{u}} = \left[\begin{array}{cc} \widetilde{\underline{f}}^b & \widetilde{\underline{\omega}}_b^{nb} \end{array}\right]^T \quad \text{(E- 141)}$$

$$\underline{v} = \left[\begin{array}{ccccccc} \underline{v}^f & \underline{v}^\omega & \underline{v}_\gamma^a & \underline{v}_\gamma^g & \underline{v}_\beta^a & \underline{v}_\beta^g \end{array}\right]^T \quad \text{(E- 142)}$$

vil de tilhørende matrisene bli

$$F(\underline{u}, \tilde{R}_b^n) = \left[\begin{array}{ccccccc} \underline{0} & I & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & -S(\tilde{R}_b^n \underline{\tilde{f}}^b) & \tilde{R}_b^n & \underline{0} & \tilde{R}_b^n & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & \tilde{R}_b^n & \underline{0} & \tilde{R}_b^n \\ \underline{0} & \underline{0} & \underline{0} & -\frac{1}{T^a} I & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & -\frac{1}{T^g} I & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \end{array}\right]$$

$$G^K(\tilde{R}_b^n) = \left[\begin{array}{cccccc} \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \tilde{R}_b^n & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \tilde{R}_b^n & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & I & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & I & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & I & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & I \end{array}\right]$$

E.2.3 Målemodell

Det blir kun gjort måleoppdateringer når en av tilstandene til systemmodellen er kjent. Som når posisjonen eller hastigheten blir gitt av en eksternkilde eller man vet at systemet står stille. Målematrisene H og H^{*} er blir da gitt etter hva det gjøres en måleoppdatering på.

Der målemodellene skal være på formen S_{DP}^{US} $\underline{z}_k = H^* \underline{x}_k$, F_{DP}^{US} $\widetilde{\underline{z}} = H\widetilde{\underline{x}}$

Måleoppdateringer på posisjonen, gir, $H = \left[\begin{array}{cc} I & \underline{0} \end{array}\right]$, $H^* = \left[\begin{array}{ccccccc} I & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \end{array}\right]$

Måleoppdateringer på Hastigheten, gir, $H = \left[\begin{array}{cc} \underline{0} & I \end{array}\right]$, $H^* = \left[\begin{array}{ccccccc} \underline{0} & I & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \end{array}\right]$

E.2.4 Spektraltettheter og kovarians

Spektraltetthetmatrisen er gitt av

$$\widetilde{Q} = \left[\begin{array}{cccccc} \widetilde{q}_f^a I & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \widetilde{q}_\omega^g I & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \widetilde{q}_\gamma^a I & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \widetilde{q}_\gamma^g I & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & \widetilde{q}_\beta^a I & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \widetilde{q}_\beta^g I \end{array}\right]$$

Kovariansmatrisen er gitt av

$$P(t_0) = \begin{bmatrix} p_{\delta p}(0)I & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & p_{\delta v}(0)I & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & p_{\varepsilon}(0)I & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & P_{\gamma}^{a}(0) & \underline{0} & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & P_{\beta}^{a}(0) & \underline{0} & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & P_{\gamma}^{g}(0) & \underline{0} \\ \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & P_{\beta}^{g}(0) \end{bmatrix}$$

Standaravviket til initielposisjonsfeil settes til lite, og kan for mobiltelefonsensor programmet egentlig settes til null. Da initiel posisjon og orientering ikke er interessant siden det er konturen til et objekt som måles og da er banen i forhold til start interessant..

$p_{\delta p}(0) = (0.01\ m)^2$

$p_{\delta v}(0) = (0.01\ m/s)^2$

$p_{\varepsilon}(0) = (0.1\ deg)^2 = (0.00175\ rad)^2$

Standardavviket til hvitstøyelementen blir i første omgang i banegeneratoren satt til 1 deg/s og 1mG/s, for akselerometer og gyrometerelementene. Spektraltetthet verdiene kan da utledes på ved hjelp av kovarianslikningene.

$P_{\gamma}^{a}(0) = I * (1mG/s)^2$

$P_{\beta}^{a}(1) = I * (1mG/s)^2$

$P_{\gamma}^{g}(0) = I * (1\,\mathrm{deg}/s)^2$

$P_{\beta}^{g}(1) = I * (1\,\mathrm{deg}/s)^2$

Ut i fra det kan initialkovarians utledes på følgende måte:

$$\frac{d}{dt}P_{\gamma}^{g}(t) = -\frac{2}{T^g}P_{\gamma}^{g}(t) + \widetilde{q}_{\gamma}^{g} \tag{E- 143}$$

Kovariansen til γ^g går mot en stasjonær tilstand når t går mot uendelig, $\frac{d}{dt}P_{\gamma}^{g}(\infty) = 0$, slik at $\frac{2}{T^g}P_{\gamma}^{g}(\infty) = \widetilde{q}_{\gamma}^{g}$ der $P_{\gamma}^{g}(0) = P_{\gamma}^{g}(\infty)$, det gir $P_{\gamma}^{g}(0) = \frac{T^g}{2}\widetilde{q}_{\gamma}^{g}$

$$\frac{d}{dt}P_{\beta}^{g}(t) = \widetilde{q}_{\beta}^{g} => P_{\beta}^{g}(t) = \widetilde{q}_{\beta}^{g} * t \tag{E- 144}$$

Når vi vet at kovariansen til β^g utvikler seg slik $P_{\beta}^{g}(t) = -\widetilde{q}_{\beta}^{g} * t$, det gir at $P_{\beta}^{g}(0) = 0$

De samme formlene gjelder for akslerometerstøyen slik at.

$$P_{\gamma}^{a}(0) = \frac{T^a}{2}\widetilde{q}_{\gamma}^{a} \tag{E- 145}$$

$$P_{\gamma}^{g}(0) = \frac{T^g}{2}\widetilde{q}_{\gamma}^{g} \tag{E- 146}$$

$$P_{\beta}^{a}(0) = 0 * I \tag{E- 147}$$

$$P_{\beta}^{g}(0) = 0 * I \tag{E- 148}$$

Spektraltettheten blir da satt til.

$$\widetilde{q}_f^a = \sigma_f^2 \quad \text{(E- 149)}$$
$$\widetilde{q}_\omega^g = \sigma_\omega^2 \quad \text{(E- 150)}$$
$$P_\beta^g(t) = \widetilde{Q}_\beta^g * t => \widetilde{q}_\beta^g = P_\beta^g(1) \quad \text{(E- 151)}$$
$$P_\beta^a(t) = \widetilde{Q}_\beta^a * t => \widetilde{q}_\beta^a = P_\beta^a(1) \quad \text{(E- 152)}$$
$$P_\gamma^g(0) = \frac{T^g}{2}\widetilde{q}_\gamma^g => \widetilde{q}_\gamma^g = \frac{2}{T^g}P_\gamma^g(0) \quad \text{(E- 153)}$$
$$P_\gamma^a(0) = \frac{T^a}{2}\widetilde{q}_\gamma^a => \widetilde{q}_\gamma^a = \frac{2}{T^a}P_\gamma^a(0) \quad \text{(E- 154)}$$

Under en måleoppdatering så har det blitt antatt at systemet enten står i start/kjent posisjon eller at det står stille. I et ekte system vil alltid være en usikkerhet rundt gitt antagelser, som for eksempel at mobilen rister. Parameteren $\underline{w}_k \tilde{} N(\underline{0}, R\delta_{kl})$ representerer da usikkerheten rundt gitt antagelser, der R blir satt til $(0,001)^2$.

E.3 Lineærisert diskretisert filtermodell

Diskretiseringen av systemet blir gjort slik som står beskrevt i innledningen. Der man finner systemet

$$\delta x_{k+1} = \Phi_k(\underline{u}_k, \tilde{R}_{b,k}^n)\delta\underline{x}_k + \Gamma_k(\tilde{R}_{b,k}^n)\underline{\tilde{v}}_k \quad \text{(E- 155)}$$
$$P_{k+1} = \Phi_k(\underline{u}_k, \tilde{R}_{b,k}^n)P_k\Phi_k^T(\underline{u}_k, \tilde{R}_{b,k}^n) + \Gamma_k(\tilde{R}_{b,k}^n)\Gamma_k^T(\tilde{R}_{b,k}^n) \quad \text{(E- 156)}$$

E.3.1 LKF likningene

Med systemmodellen definert kan LKF likningene settes opp slik:

TO

$$\delta\overline{\underline{x}}_{k+1} = \Phi_k(\underline{u}_k, \tilde{R}_{b,k}^n)\delta\underline{\hat{x}}_k \quad \text{(E- 157)}$$
$$\bar{P}_{k+1} = \Phi_k(\underline{u}_k, \tilde{R}_{b,k}^n)\bar{P}_k\Phi_k^T(\underline{u}_k, \tilde{R}_{b,k}^n) + \Gamma_k(\tilde{R}_{b,k}^n)Q\Gamma_k^T(\tilde{R}_{b,k}^n) \quad \text{(E- 158)}$$
$$der\ Q = I,\ P_0 = \Delta t * P(t_0),\ \delta\underline{\hat{x}}_0 = \underline{0} \quad \text{(E- 159)}$$

MO

$$\delta\underline{z}_k = \underline{z}_k - \widetilde{\underline{z}}_k \quad \text{(E- 160)}$$
$$K_k = \bar{P}_k H_k^T(H_k\bar{P}_k H_k^T - Rd)^{-1},\ \bar{P}_k = \bar{P}(t_k) \quad \text{(E- 161)}$$
$$\delta\underline{\hat{x}}_k = \delta\overline{\underline{x}}_k + K_k(\delta\underline{\tilde{z}}_k - H_k\delta\overline{\underline{x}}_k) \quad \text{(E- 162)}$$
$$\underline{\hat{x}}_k = \widetilde{\underline{x}}_k + \delta\underline{\hat{x}}_k \quad \text{(E- 163)}$$
$$\hat{P}_k = (I - K_k H_k)\bar{P}_k \quad \text{(E- 164)}$$
$$\bar{P}_k = \bar{P}(t_0)\Delta t \quad \text{(E- 165)}$$

Måleoppdatering

Når måleoppdatering til mobilsensorprogrammet blir gjort er det valgt to forskjellige typer måleoppdateringer. Det er valgt posisjons oppdatering og hastighets oppdatering. Der kunnskap man har om den sanne banen til mobilen utnyttes. Systemet står stille ved 0 til 2 sek., 4 til 6 sek., 8 til 10 sek og 12 til 14 sekunder. Fra 0 til 2 sekunder står mobilen i en kjent start po-

sisjon, i programmet $\underline{p}_k^n = \begin{bmatrix} \underline{0} & \underline{0} & \underline{0} \end{bmatrix}^T$, og det blir gjort en måleoppdatering på posisjonen. Der $\delta\underline{z}_k = \delta\underline{p}_k^n = \underline{p}_k^n - \widetilde{\underline{p}}_k^n = -\widetilde{\underline{p}}_k^n$

$t < 0s, 2s > \Rightarrow k < 1, 200 >$

$H_k = \begin{bmatrix} I & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \end{bmatrix}$

Fra 4 til 6 sek., 8 til 10 sek og 12 til 14 sekunder står mobilen i ro, dvs. hastigheten $\underline{v}_k^n = \begin{bmatrix} \underline{0} & \underline{0} & \underline{0} \end{bmatrix}^T$, og det blir gjort måleoppdateringer på hastigheten. Der $\delta\underline{z}_k = \delta\underline{v}_k^n = \underline{v}_k^n - \widetilde{\underline{v}}_k^n = -\widetilde{\underline{v}}_k^n$

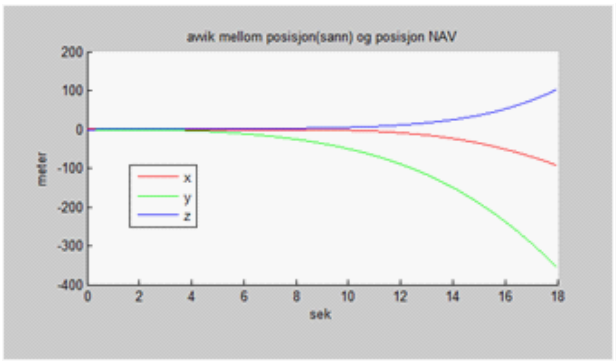
$t < 4s, 6s >< 8s, 10s >< 12s, 14s >< 16s, 18s >$

$\Rightarrow k < 1, 200 >< 801, 1000 >< 1201, 1400 >< 1601, 1800 >$

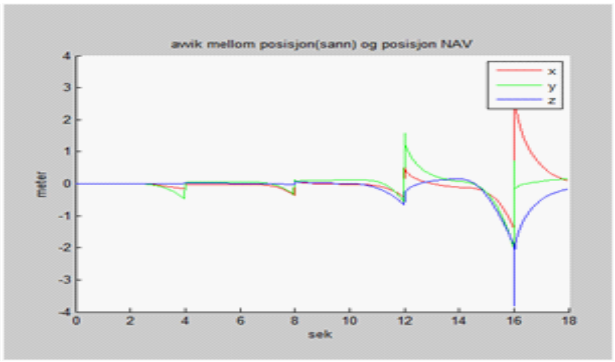
$H_k = \begin{bmatrix} \underline{0} & I & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \end{bmatrix}$

E.4 Resultater av TNS med et LKF

Etter å ha juster Kalmanfilteret for støy i sensorene til navigasjonssystemet, blir resultatet som forventet avviket går fra en posisjonsfeil på godt over 200 meter til en posisjonsfeil på 4 meter. Det er fortsatt et alt for stort avvik med tanke på at mobilen kun beveger seg 120 cm.

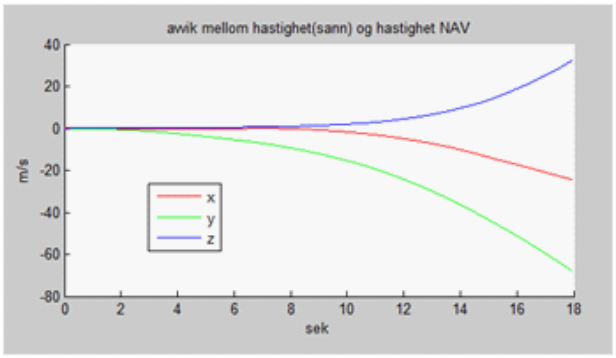


Avvik på posisjon Euler-Heuns

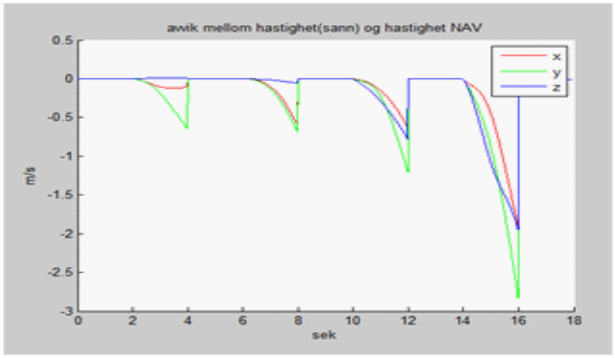


Avvik på posisjon Kalman

Hastighets estimering blir da også forbedret fra en presisjon på +- 40-50 meter til en presisjon på +-3 meter.

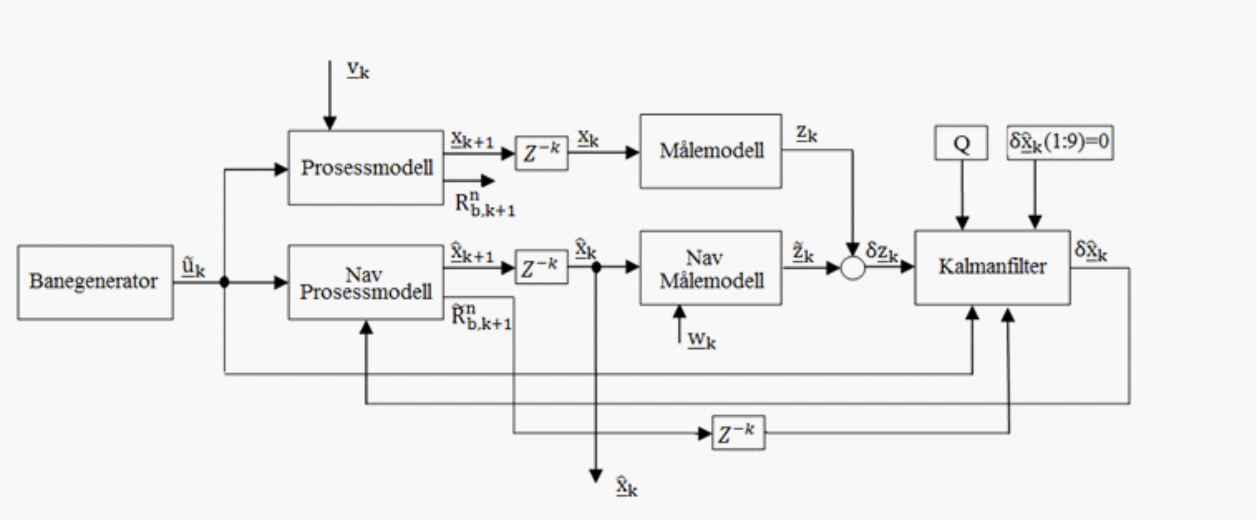


Avvik på hastighet, Euler-Heuns



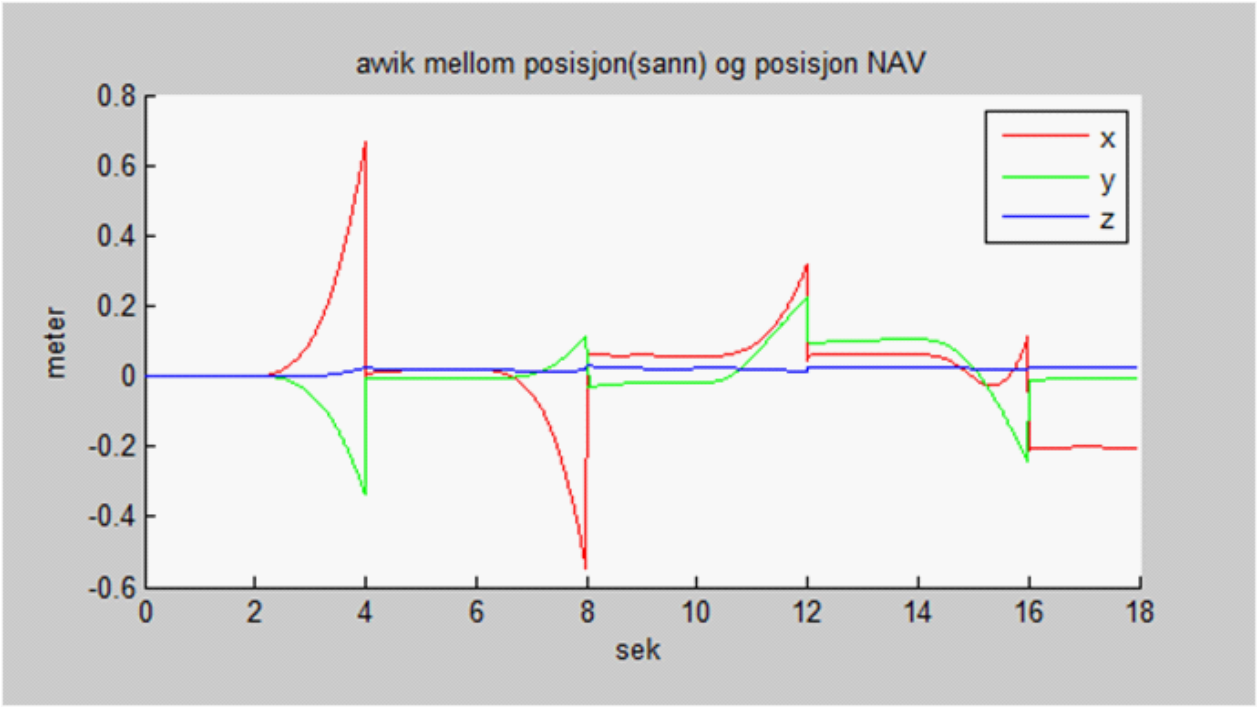
Avvik på hastighet, Kalman

Ut i fra dette resultatet kan jeg konkludere med at avviket fra sann posisjon er så stort at det vil være smart gjøre det om til et tilbakekoblet Kalmanfilter. Med en tilbakekobling vil Euler-Nav likningene få et bedre utgangspunkt til og prediktere neste tilstand. Tilbakekoblingen gir da systemmodellen som ses i blokkskjemaet for en systemmodell med tilbakekobling.



Blokkskjema, systemmodell med tilbakekobling

Tilbakekoblingen gir da et resultat der posisjonsestimatet har en presisjon på ca +- 60cm. Noe som er et greit utgangspunkt til kovariansanalysen.

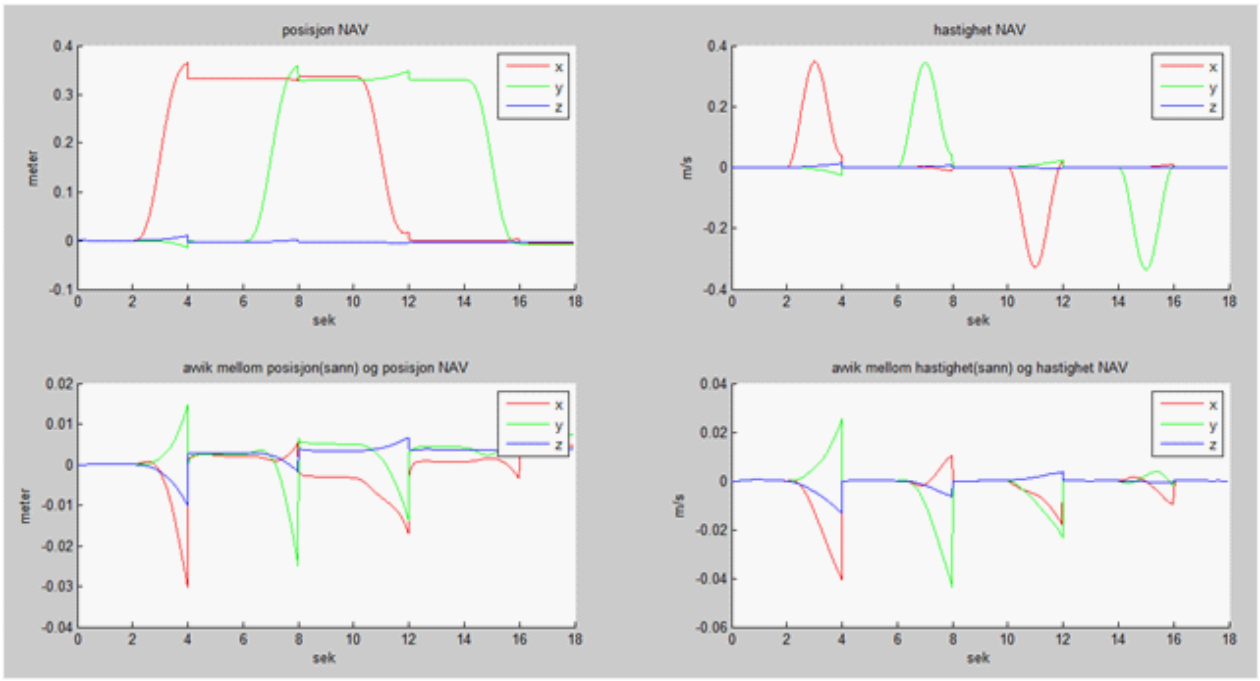


Kalman euler Nav error

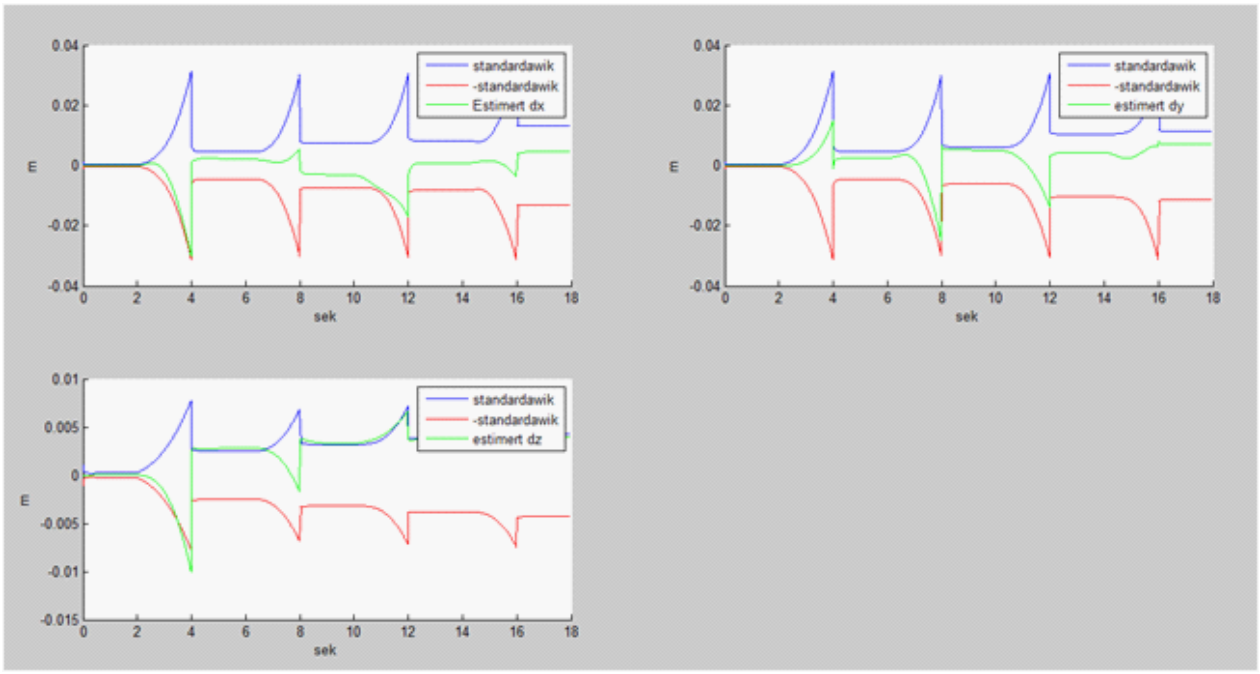
E.4.1 Euler mot Euler-Heuns metoden

Feillikningene til Kalman filteret er utledet med hensyn på Euler navigasjonsmetoden. Det vil si at når kalmanfiltert blir brukt til å minimere feilen fra Euler-Heuns likningene, kan det forekome numeriske feil. Ved å kjøre en simulering på begge alternativene ser man at det ikke er tilfelle og at det er liten forskjell i resultatet fra Euler likningene og Euler-Heuns likningene.

Euler:

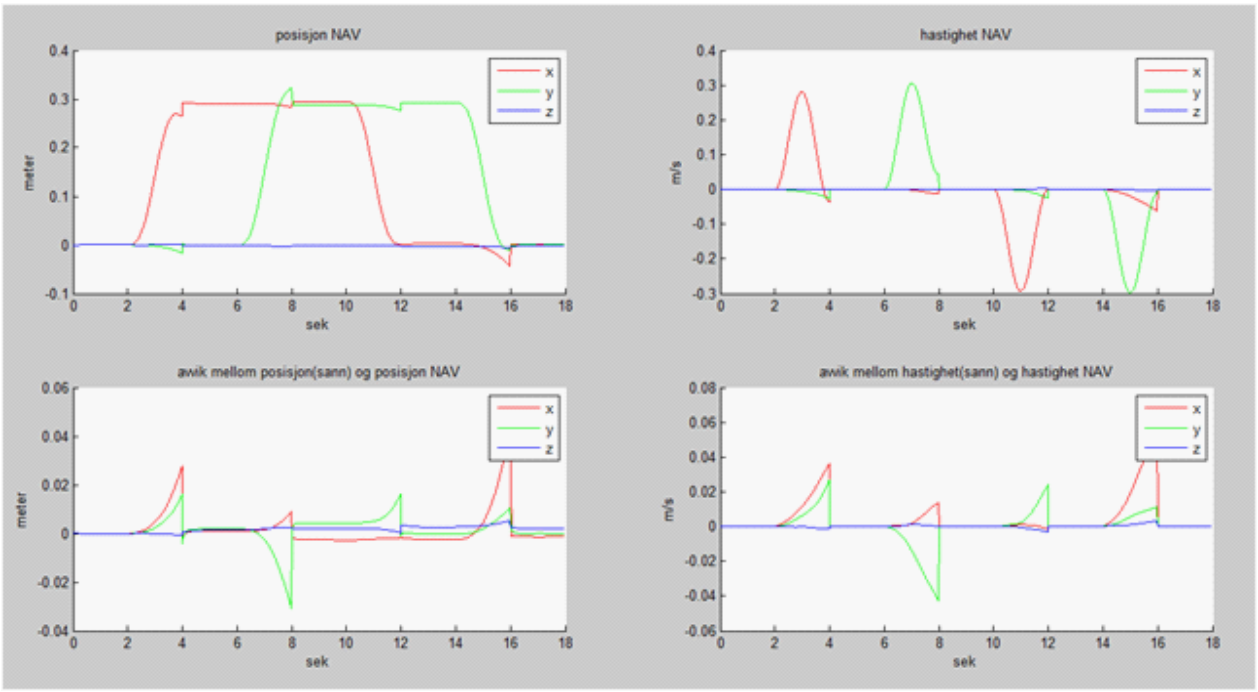


Kalman Euler

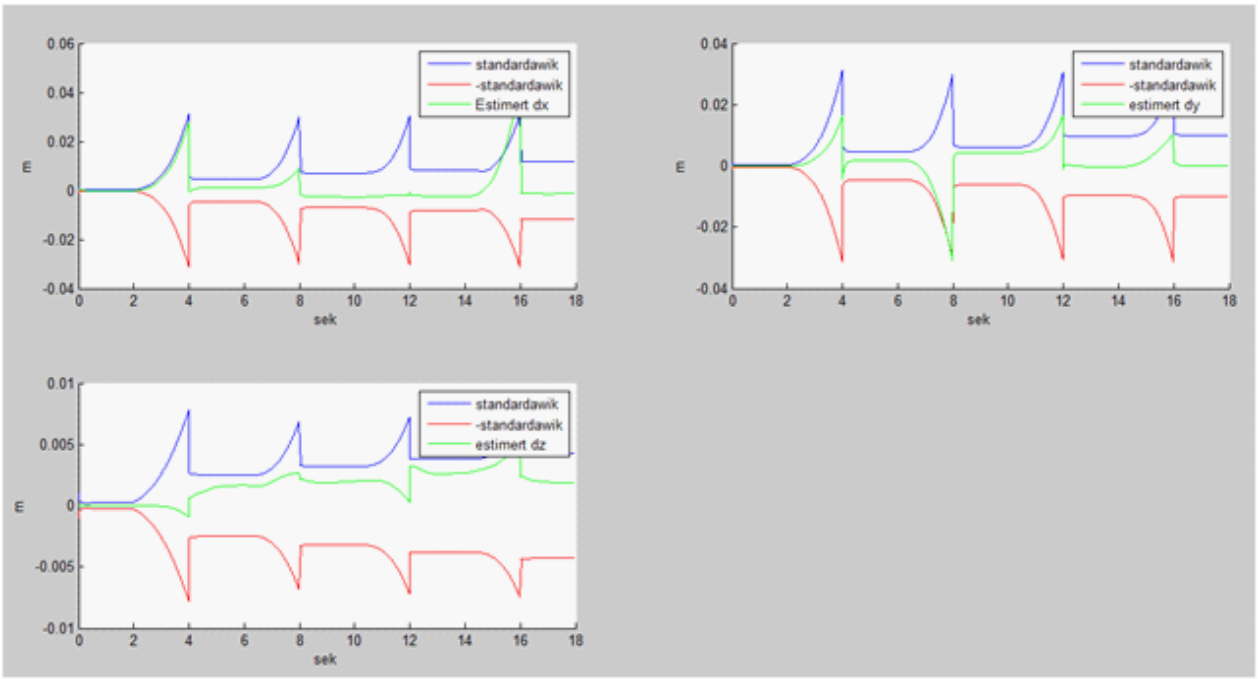


Euler posisjonsfeil

Euler-Heuns:



Kalman Euler-Heuns



Euler-Heuns posisjonsfeil

Feilen fra begge metodene ligger i samme størelses område.

Del F: Feilbudsjett

Et feilbudsjett er en analyse av hvordan forskjellige feilkilder eller feilgrupper påvirker totalfeilen til systemet. Når det lages så antar man at man har et lineæret Kalman-aktig system der valgt initialverdier, kovarianser og spektraltettheter har et eller annet bidrag til totalfeilen. Feilbudsjettet blir da en katalog over de bidragene. [2]

Feilbudsjetligningene blir som følger.

$$\frac{d}{dt}\overline{P}_\alpha = F_\alpha \overline{P}_\alpha + \overline{P}_\alpha F_\alpha^T + G_\alpha \widetilde{Q} G_\alpha^T \qquad (TO) \tag{F- 166}$$

$$\widehat{P}_\alpha(\widehat{t}_k) = (I_\alpha - K_\alpha H_\alpha)\overline{P}_\alpha(\bar{t}_k)(I_\alpha - K_\alpha H_\alpha)^T + K_\alpha \widetilde{R} K_\alpha^T \qquad (MO) \tag{F- 167}$$

$$\overline{P}_\alpha(\bar{t}_0) = \begin{bmatrix} N\overline{P}_0 N^T & N\overline{P}_0 \\ \overline{P}_0 N^T & \overline{P}_0 \end{bmatrix}$$

$$F_\alpha = \begin{bmatrix} F^* & \Delta F^* \\ \underline{0} & F \end{bmatrix} \qquad G_\alpha = \begin{bmatrix} NG \\ G \end{bmatrix} \qquad H_\alpha = \begin{bmatrix} H^* & \Delta H \end{bmatrix} \qquad K_\alpha = \begin{bmatrix} K_k^* \\ \underline{0} \end{bmatrix}$$

$$I_\alpha = \begin{bmatrix} I^* & \underline{0} \\ \underline{0} & I \end{bmatrix} \qquad \Delta F^* = NF - F^* Nds \qquad \Delta H = H - H^* N = \underline{0} \qquad N = \begin{bmatrix} I^* & \underline{0} \end{bmatrix}$$

Siden bare støybeskrivelsen er feil kan kovariansanalysen behandles som et spesialtilfelle [3] der $\Delta F = 0$ og $\Delta H = 0$ da kan likningene forenkles til følgende.

$$I_\alpha = I^* \tag{F- 168}$$

$$F_\alpha = F^* \tag{F- 169}$$

$$H_\alpha = \Delta H \tag{F- 170}$$

$$K_\alpha = K_k^* \tag{F- 171}$$

$$G_\alpha = G \tag{F- 172}$$

$$\frac{d}{dt}\overline{P} = F^* \overline{P} + \overline{P} F^{*T} + G \widetilde{Q} G^T \qquad (TO) \tag{F- 173}$$

$$\widehat{P}(\widehat{t}_k) = (I - K_k H^*)\overline{P}(\bar{t}_k)(I - K_k H^*)^T + K_k \widetilde{R} K_k^T \qquad (MO) \tag{F- 174}$$

$$\widehat{P}_0 = \widehat{P}(\widehat{t}_0) = \overline{P}_0$$

F.1 Feilgrupper

Feilgruppen ble valgt på grunnlag av gruppe tilhørlighet til den respektive initiale kovariansen, variansen eller spektraltettheten. Der initial kovariansen til posisjonen har blitt en gruppe, initial kovariansen til gammaen pluss spektraltettheten til hvitstøy elementet i gammaen har blitt en annen gruppe osv.

[2] (Arthur Gelb, Applied Optimal Estimation s.260)
[3] (Oddvar Hallingstad, Notat 7 Monte Carlo og kovarians analyse av Kalman filteret)

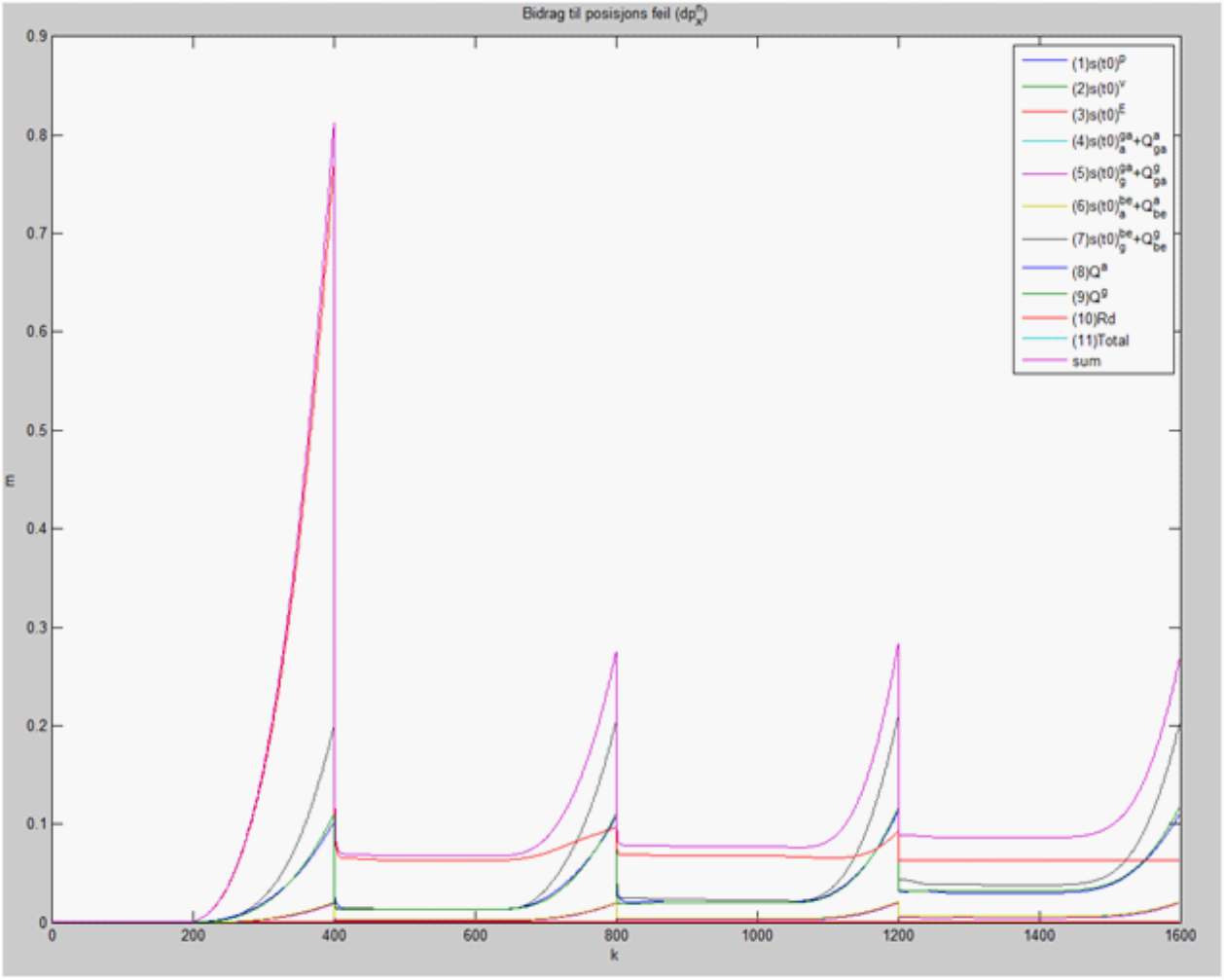
1) $\sigma^{p}(t_0)$ (F- 175)
2) $\sigma^{v}(t_0)$ (F- 176)
3) $\sigma^{\varepsilon}(t_0)$ (F- 177)
4) $\sigma^{\gamma^a}(t_0) + Q_{\gamma}^{a}$ (F- 178)
5) $\sigma^{\gamma^g}(t_0) + Q_{\gamma}^{g}$ (F- 179)
6) $\sigma^{\beta^a}(t_0) + Q_{\beta}^{a}$ (F- 180)
7) $\sigma^{\beta^g}(t_0) + Q_{\beta}^{g}$ (F- 181)
8) Q^a (F- 182)
9) Q^g (F- 183)
10) Rd (F- 184)

Selve kovarians analysen blir gjort ved og kjøre gjennom hele målesettet, en gang for hver feilgruppe der bidraget for alle andre feilgrupper er satt til null. Deretter må det kjøres igjennom engang med alle feilene for og finne totalfeilen som skal være lik kvadratroten av summen av alle de andre feilgruppene kvadrert.

Målet med dette programmet var å lage en MatLab funksjon som gjør en kovarians analyse og returnerer et feilbudsjett. Det skal fungere på minst to forskjellige NAV systemer. Det har derfor blitt antatt at tilstandsvektor og hvitstøyvektor ser slik ut, $\underline{x} = \begin{bmatrix} \underline{p}^n & \underline{v}^n & \underline{\gamma}^a & \underline{\gamma}^g & \underline{\beta}^a & \underline{\beta}^g \end{bmatrix}^T$, $\underline{v} = \begin{bmatrix} \underline{v}^f & \underline{v}^\omega & \underline{v}_{\gamma}^{a} & \underline{v}_{\gamma}^{g} & \underline{v}_{\beta}^{a} & \underline{v}_{\beta}^{g} \end{bmatrix}^T$, i alle tilfeller.

F.2 Resultat av kovarians analysen

Jeg hadde ikke noen forventninger til hvordan de forskjellige feilgruppene ville påvirke totalfeilen. Etter første simulering ble det klart at usikkerheten i antagelsen om at systemet står i sann p(0) ved start, ved den første måleoppdatering, det var det klart største bidraget til totalfeilen. Deretter er det usikkerheten i at systemet står i ro ved resten av måleoppdateringen som gir det største bidraget til totalfeilen. Tilslutt kan det også konkluderes med at støy i vinkelhastighetsmålingene gir et betydelig større bidrag en støy fra akselerasjonsmålingene. Figuren, Bidrag til posisjons feil, viser et plot av bidraget fra de respektive feilgruppene til posisjonsfeil om x-aksen.



Bidrag til posisjonsfeil.

Et fenomen som er bemerkelses verdig er at selv om en måleoppdatering på posisjonen gir bedre nøyaktighet til selve posisjonsestimeringen under måleoppdateringen, vil det føre til større usikkerhet i systemet når det skal prediktere neste tilstand, enn det en måleoppdatering på farten ville gjort.

Fra dette feilbudsjettet og tilhørende simulering av TNSet kan det konkluderes med at usikkerheten i antagelsen om at systemet står i faktisk start posisjon og at usikkerheten i at systemet står i ro under respektive måleoppdateringer er for dårlig og må forbedres. Fra mm til hvertfall mm/10 presisjon og tilsvarende mm/s til (mm/s)/10 presisjon på respektive måleoppdateringer. Det kan også konkluderes med at en gyrodrift på 1 deg/s er ikke bra nokk til å ha et TNS med ca 1 cm presisjon på posisjons usikkerheten etter 16 sekunder, og en eventuell gyrodrift burde ikke være noe mer en 1/15 deg/s.

Variansen i hvitstøy elementene til gyrometer og akslerometerstøyen blir dermed satt til følgende.

Gyrometer:

$$\underline{\widetilde{\omega}}^b(t_k) = \underline{\omega}^b(t) + \underline{\gamma}^a(t) + \underline{\beta}^a(t) + \underline{v}^\omega \quad \text{(F- 185)}$$

$$\frac{d}{dt}\underline{\gamma}^g(t) = -\frac{T^g}{2}\underline{\gamma}^g(t) + v_\gamma^g \quad \text{(F- 186)}$$

$$\frac{d}{dt}\underline{\beta}^g(t) = \underline{v}_\beta^g \quad \text{(F- 187)}$$

der

$\underline{v}^{\omega}(t)\tilde{}N(\underline{0}, \widetilde{Rd}_{\omega})$ og $\widetilde{Rd}_{\omega} = (\frac{0.017453}{15}\frac{rad}{s})^2$

$\underline{v}_{\gamma}^{g}(t)\tilde{}N(\underline{0}, \widetilde{Q}_{\gamma}^{g})$ og $\widetilde{Q}_{\gamma}^{g} = (\frac{0.017453}{15}\frac{rad}{s})^2$

$\underline{v}_{\beta}^{g}(t)\tilde{}N(\underline{0}, \widetilde{Q}_{\gamma}^{g})$ og $\widetilde{Q}_{\beta}^{g} = (\frac{0.017453}{15}\frac{rad}{s})^2$

Akslerometer:

$$\underline{\widetilde{f}}^{n}(t_k) = \underline{f}^{n}(t) + \underline{\gamma}^{a}(t) + \underline{\beta}^{a}(t) + \underline{v}_{f}(t) \quad \text{(F- 188)}$$

$$\frac{d}{dt}\underline{\gamma}^{a}(t) = -\frac{T^a}{2}\underline{\gamma}^{a}(t) + \underline{v}_{\gamma}^{a}(t) \quad \text{(F- 189)}$$

$$\frac{d}{dt}\underline{\beta}^{a}(t) = \underline{v}_{\beta}^{a}(t) \quad \text{(F- 190)}$$

der

$\underline{v}^{f}(t)\tilde{}N(\underline{0}, \widetilde{Rd}_{f})$ og $\widetilde{Rd}_{f} = (0.00981\frac{m/s^2}{s})^2$

$\underline{v}_{\gamma}^{a}(t)\tilde{}N(\underline{0}, \widetilde{Q}_{\gamma}^{a})$ og $\widetilde{Q}_{\gamma}^{a} = (0.00981\frac{m/s^2}{s})^2$

$\underline{v}_{\beta}^{a}(t)\tilde{}N(\underline{0}, \widetilde{Q}_{\beta}^{a})$ og $\widetilde{Q}_{\beta}^{a} = (0.00981\frac{m/s^2}{s})^2$

Initial kovarians blir da også justert tilsvarende for både gyrometerstøyen og asklerometerstøyen. I Kalmanfilteret må også usikkerheten i antagelsen om at systemet står i faktisk start posisjon og usikkerheten i antagelsen i at systemet står stille under respektive måleoppdateringer forbedres.

Kalmanfilterlikningene for måleoppdatering:

$$K_k = \bar{P}_k H^T (H\bar{P}_k H^T - Rd)^{-1} \quad \text{(F- 191)}$$

$$\underline{\hat{x}}_k = \underline{\bar{x}}_k + K_k(\underline{\tilde{z}}_k - H\underline{\bar{x}}_k) \quad \text{(F- 192)}$$

$$\hat{P}_k = (I - K_k H)\bar{P}_k \quad \text{(F- 193)}$$

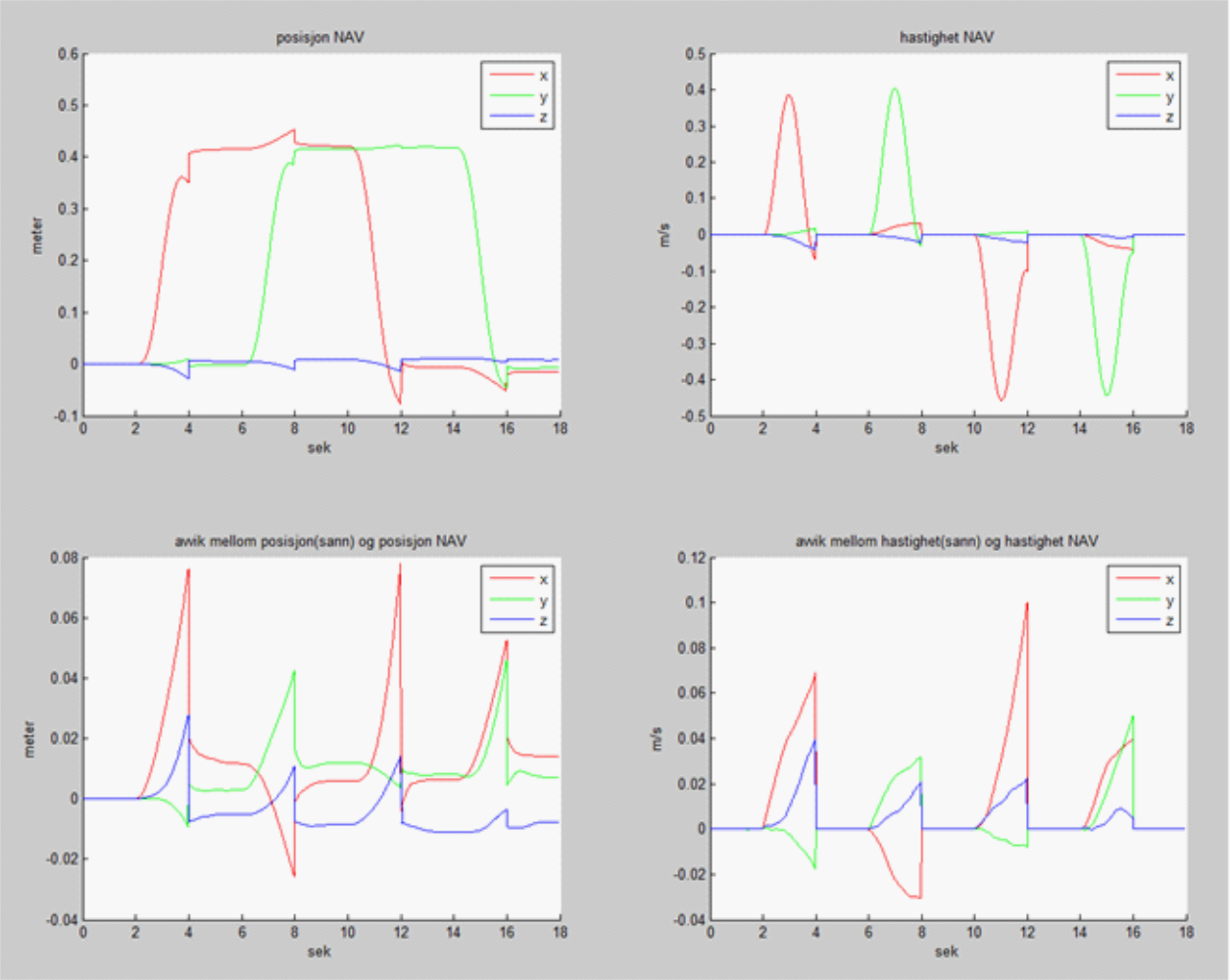
der Rd_1 dvs. usikkerheten i måloppdatering på posisjonen blir.

$$Rd_1 = (\frac{0.0001}{10})^2 \quad \text{(F- 194)}$$

der Rd_2 dvs. usikkerheten i måloppdatering på hastigheten blir.

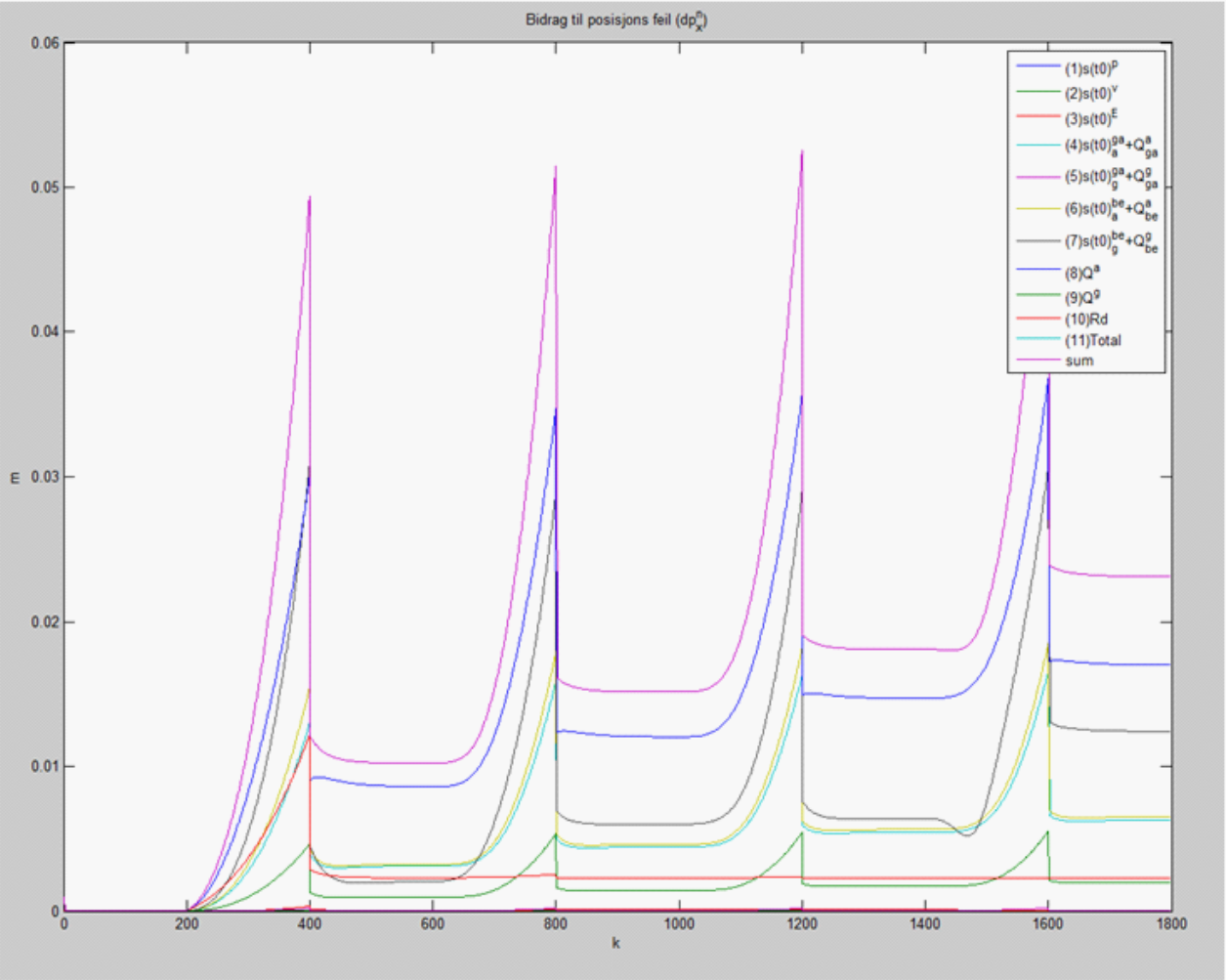
$$Rd_2 = (\frac{0.0001}{10})^2 \quad \text{(F- 195)}$$

Det ble også konkludert med at simuleringstiden burde forlenges til 18s og at blir lagt inn en måleoppdatering de 2 siste sekundene. Resultatet av disse justeringene gir da følgende simulerings resultat



Nav resultat

Det som kan observeres her i denne simuleringen med forbedret gyrometersensorer og strengere betingelser i Kalmanfilteret om initial posisjons sikkerhet og usikkerhet rundt antagelsen om at systemet står stille under måleoppdateringer er. Avviket mellom sann posisjon og estimert posisjoner har falt fra 1m presisjon etter 16s, til å ligge på 6-8cm presisjon etter 16s.



Bidrag til posisjonsfeil.

Resultatet fra feilbudsjettet viser nå at de ikke lenger er usikkerheten i antagelsen rundt måleoppdateringene som er det største bidraget. Det er nå blitt hvitstøy faktorene i akselerometer støyen som har de største bidragene til totalfeilen. Hvitstøy elementet i beta faktoren til gyrometerstøyen gir også et betydelig bidrag til totalfeilen.

Det konkluderes dermed med at driften som følge av akselerasjonsmålingene er det som begrenser systemet i og ha en ønsket presisjon på 1cm i usikkerheten på posisjonsmålingen etter 16 sekunder.

Variansen i hvitstøy elementene til gyrometer og akslerometerstøyen blir dermed satt til følgende.

Gyrometer:

$\underline{v}^{\omega}(t) \tilde{} N(\underline{0}, \widetilde{Rd}_{\omega})$ og $\widetilde{Rd}_{\omega} = (\frac{0.017453}{15}\frac{rad}{s})^2$

$\underline{v}^{g}_{\gamma}(t) \tilde{} N(\underline{0}, \widetilde{Q}^g_{\gamma})$ og $\widetilde{Q}^g_{\gamma} = (\frac{0.017453}{15}\frac{rad}{s})^2$

$\underline{v}^{g}_{\beta}(t) \tilde{} N(\underline{0}, \widetilde{Q}^g_{\gamma})$ og $\widetilde{Q}^g_{\beta} = (\frac{0.017453}{15}\frac{rad}{s})^2$

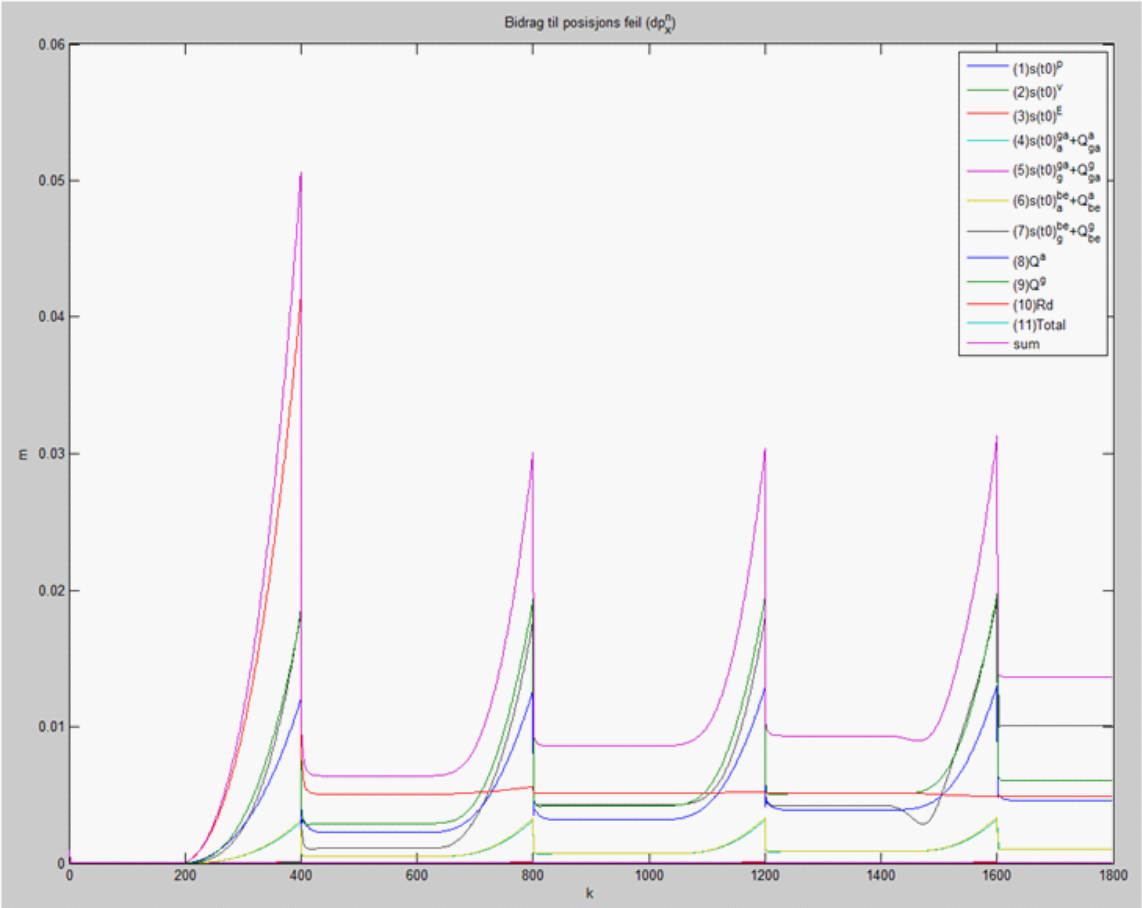
Akslerometer:

$\underline{v}^{f}(t) \tilde{} N(\underline{0}, \widetilde{Rd}_{f})$ og $\widetilde{Rd}_{f} = (\frac{0.00981}{5}\frac{m/s^2}{s})^2$

$\underline{v}^{a}_{\gamma}(t) \tilde{} N(\underline{0}, \widetilde{Q}^a_{\gamma})$ og $\widetilde{Q}^a_{\gamma} = (\frac{0.00981}{5}\frac{m/s^2}{s})^2$

$\underline{v}^{a}_{\beta}(t) \tilde{} N(\underline{0}, \widetilde{Q}^a_{\beta})$ og $\widetilde{Q}^a_{\beta} = (\frac{0.00981}{5}\frac{m/s^2}{s})^2$

Under ser man bilde av feilbudsjettet fra en simulering med akslerometerstøyen justert.



Bidrag til posisjonsfeil

Resultatet blir som forventet et redusert bidrag fra akslerometerstøy elementene. Et litt mindre forventet resultat er at usikkerheten i antagelsen rundt måleoppdateringene gir nå et større bidrag enn før justeringen av akslerometerstøyen.

Videre kan det ses at avviket mellom estimert og sann posisjon er nå redusert til å ligge under 2 cm etter 16 sekunder, det nærmer seg ønsket drift

Selve feilbudsjettet tabellen ser slik ut.

```
fbr =

    0.0000    0.0000    0.0000    0.0011    0.0000    0.0011    0.0065    0.0046    0.0064    0.0075    0.0128    0.0128
    0.0000    0.0000    0.0000    0.0011    0.0000    0.0011    0.0065    0.0046    0.0064    0.0075    0.0128    0.0128
    0.0000    0.0000    0.0000    0.0012    0.0000    0.0014    0.0001    0.0026    0.0001    0.0022    0.0039    0.0039
    0.0000    0.0000    0.0000    0.0013    0.0001    0.0014    0.0082    0.0049    0.0080    0.0081    0.0150    0.0150
    0.0000    0.0000    0.0000    0.0013    0.0001    0.0014    0.0080    0.0049    0.0080    0.0081    0.0148    0.0148
    0.0000    0.0000    0.0000    0.0010    0.0000    0.0015    0.0001    0.0017    0.0002    0.0013    0.0029    0.0029
    0.0000    0.0000    0.0000    0.0002    0.0000    0.0006    0.0011    0.0006    0.0010    0.0008    0.0019    0.0019
    0.0000    0.0000    0.0000    0.0002    0.0000    0.0006    0.0011    0.0006    0.0010    0.0008    0.0019    0.0019
    0.0000    0.0000    0.0002    0.0005    0.0000    0.0005    0.0253    0.0020    0.0045    0.0009    0.0258    0.0258
    0.0000    0.0000    0.0000    0.0019    0.0000    0.0000    0.0000    0.0000    0.0000    0.0001    0.0019    0.0019
    0.0000    0.0000    0.0000    0.0019    0.0000    0.0000    0.0000    0.0000    0.0000    0.0001    0.0019    0.0019
    0.0000    0.0000    0.0000    0.0018    0.0000    0.0004    0.0000    0.0003    0.0000    0.0002    0.0019    0.0019
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0059    0.0002    0.0000    0.0002    0.0001    0.0059    0.0059
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0059    0.0002    0.0000    0.0002    0.0001    0.0059    0.0059
    0.0000    0.0000    0.0000    0.0013    0.0000    0.0021    0.0000    0.0012    0.0000    0.0006    0.0028    0.0028
    0.0000    0.0000    0.0000    0.0001    0.0000    0.0001    0.0012    0.0003    0.0007    0.0004    0.0015    0.0015
    0.0000    0.0000    0.0000    0.0001    0.0000    0.0001    0.0012    0.0003    0.0007    0.0004    0.0015    0.0015
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0035    0.0002    0.0003    0.0001    0.0035    0.0035
```

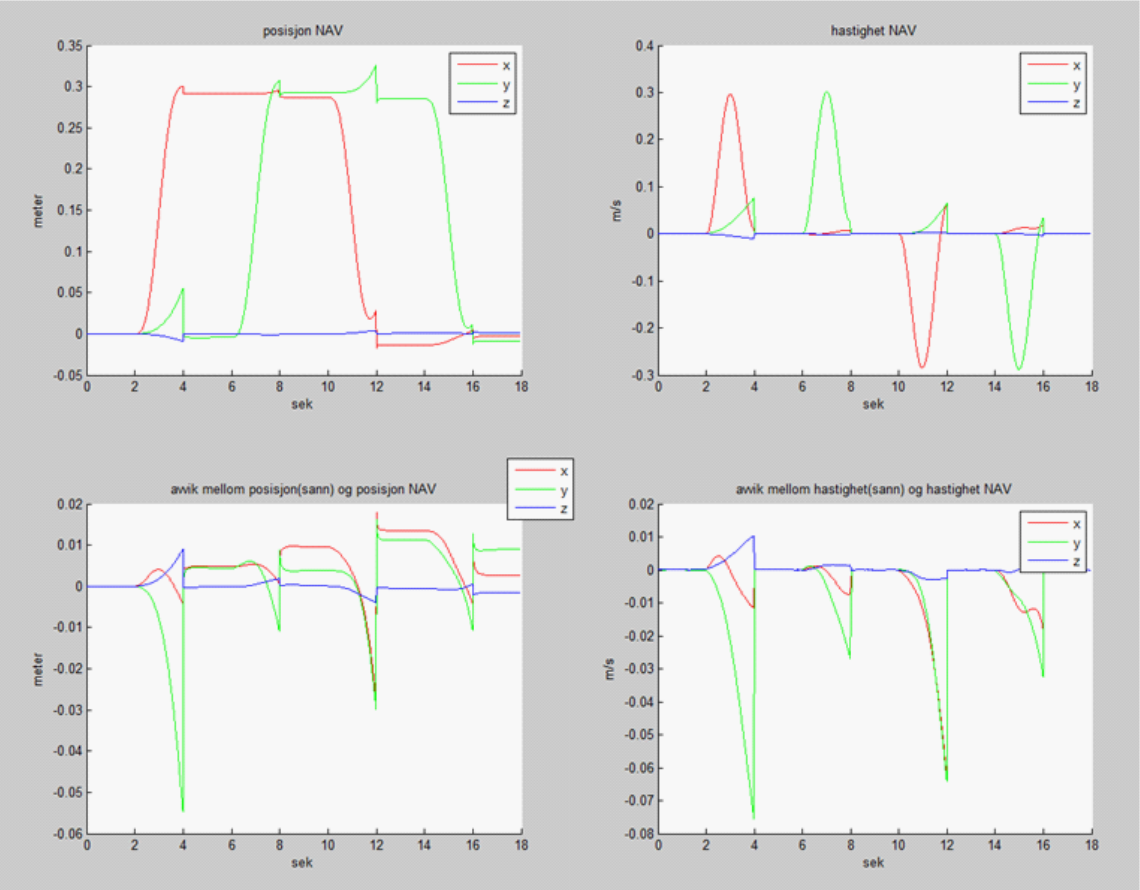


Figure 1: Nav resultat

Feilbudsjett

Det er en RMS summering av alle variablene, der radene viser de forskjellige variablene i x vektoren. Og kollonnene viser de forskjellige feilgruppene, de to siste kollonnene viser Totalfeilen og summen av alle feilgruppene. Der det kan tolkes som et sammendrag av kovariansanalysen.

F.2.1 Pseudo-kode Feilbudsjett

$function \quad [res] = errbud(F\_\_st, G\_\_st, H\_\_st, K\_\_st, Pe\_0, Qb, Rd, dt)$

$for \quad i = 1 : 11$

$Pe = Pe\_0 * 10^{-9};$

$Q_{temp} = Qb * 10^{-9};$

$R = 0;$

$if \quad i == 1$

$Pe(1:3,:) = Pe\_0(1:3,:) * dt;$

$elseif \quad i == 2$

$Pe(4:6,:) = Pe\_0(4:6,:) * dt;$

$elseif \quad i == 3$

$Pe(7:9,:) = Pe\_0(7:9,:) * dt;$

$elseif \quad i == 4$

$Pe(10:12,:) = Pe\_0(10:12,:) * dt;$

$Q_{temp}(7:9,:) = Qb(7:9,:);$

$elseif \quad i == 5$

$Pe(13:15,:) = Pe\_0(13:15,:) * dt;$

$Q_{temp}(10:12,:) = Qb(10:12,:);$

$elseif \quad i == 6$

$Pe(16:18,:) = Pe\_0(16:18,:) * dt;$

$Q_{temp}(13:15,:) = Qb(13:15,:);$

$elseif \quad i == 7$

$Pe(19:21,:) = Pe\_0(19:21,:) * dt;$

$Q_{temp}(16:18,:) = Qb(16:18,:);$

$elseif \quad i == 8$

$Q_{temp}(1:3,:) = Qb(1:3,:);$

$esleif \quad i == 9$

$Q_{temp}(4:6,:) = Qb(4:6,:);$

$esle \quad i == 10$

$R = Rd$

$end \quad if$

$for \quad k = 1 : n$

$F = F\_\_st(:,:,k);$

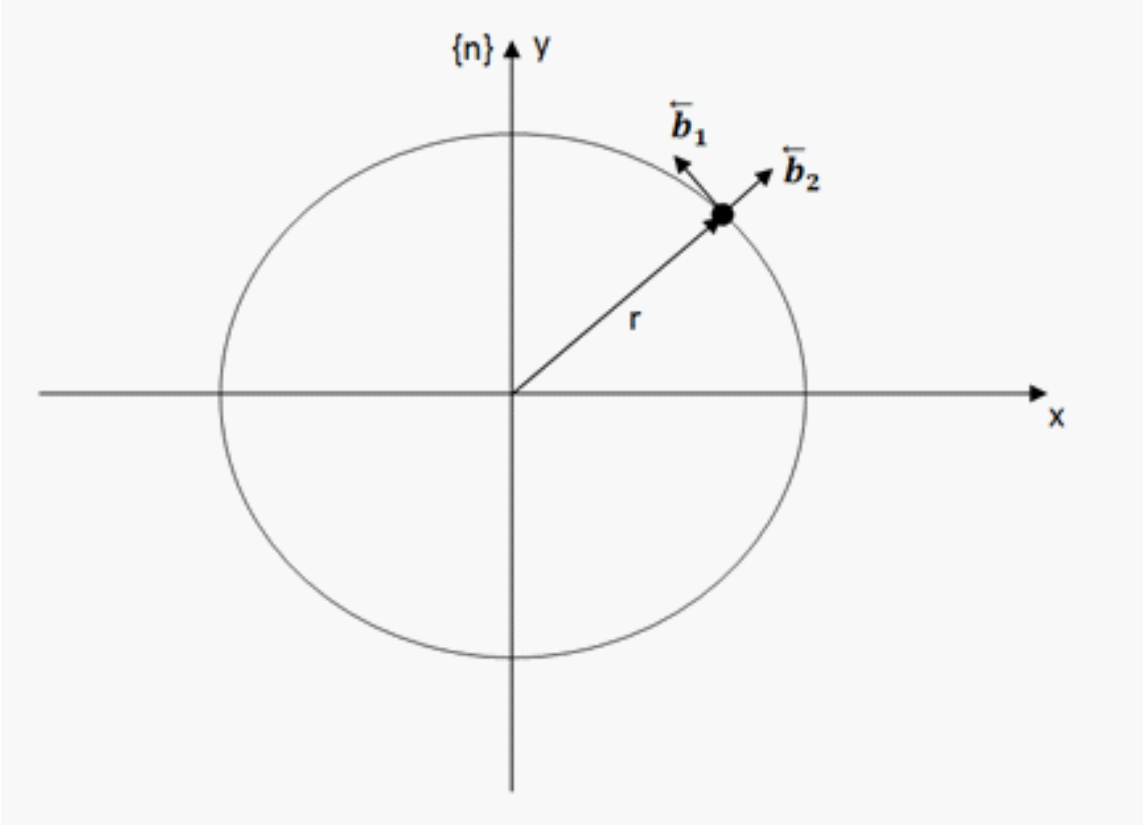
$$
\begin{aligned}
&K = K\_\_st(:,:,k);\\
&H = H\_\_st(:,:,k);\\
&G = G\_\_st(:,:,k);\\
&[\Phi,\Gamma] = diskretiser(dt,F,G,Q_{temp});\\
&\\
&Pp = \Phi Pe\Phi^T + \Gamma\Gamma^T;\\
&Pe = (I-KH)Pp(I-KH)^T + KRK^T;\\
&res(:,k,i) = sqrt(diag(Pe));\\
&end \quad for\\
&fbr(:,i) = rms(res(:,:,i));\\
end \quad &for\\
plot(&res)\\
print(&fbr)\\
end \quad &function
\end{aligned}
$$

Del G: Modelfly

G.1 Banegenerator for modell fly.

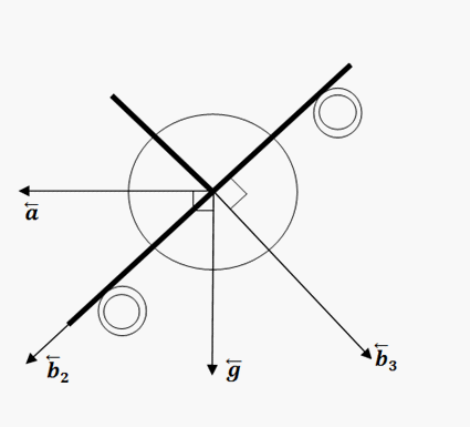
I denne delen videre utvikles det en banegenerator som genererer målinger fra et modelfly. Banene til modellflyet skal være en sirkel med en radius på minimum 1000 meter. Flyet skal ha en hastighet på minimum 100 m/s og omkretstiden skal være minst 1 minutt.

Ut ifra de opplysningene settes radiusen til 1000m og omkretstiden settes til 60 sekunder. Det gir en simuleringstid på 3*60 sek, 180 sekunder, da systemet først skal kjøre en runde med kontinuerlig måleoppdateringer, deretter minst en runde uten noen måleoppdatering. Måleoppdateringene som modelleres er GPS oppdateringer, måleoppdateringene blir da gjort på posisjonen siden GPS hovedsakelig gir en posisjons vektor. Den geografiske rammen som det skal navigeres fra blir definert som et kartesisk koordinat system med origo i null. Disse antagelsene vil gi en bane sett fra den geografiske rammen som illustrert i figuren under.

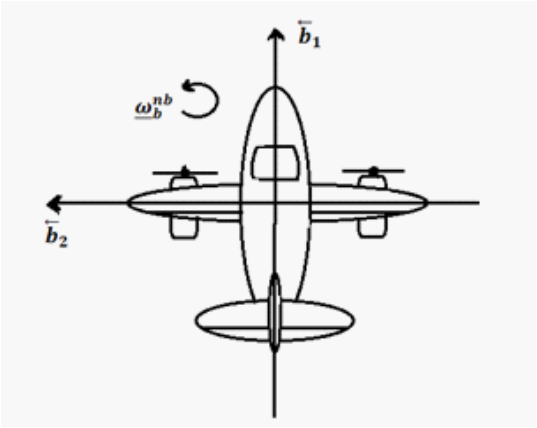


Modellflybanen representert i den geografiske rammen.

Når flyet flyr i en sirkulærbane må det antas at det er tiltet litt slik at det for passasjerene virker som tyngdekraften går rett ned. Målingene fra sensorene som antas og ligge i massesenteret til flyet med aksene langs roll, pitch og yaw aksen må da transformers tilsvarende. Figuren under til venster illustrerer hvordan tyngdekraften og sentripetalakselerasjonen virker på flyet. Figur ... illusterer



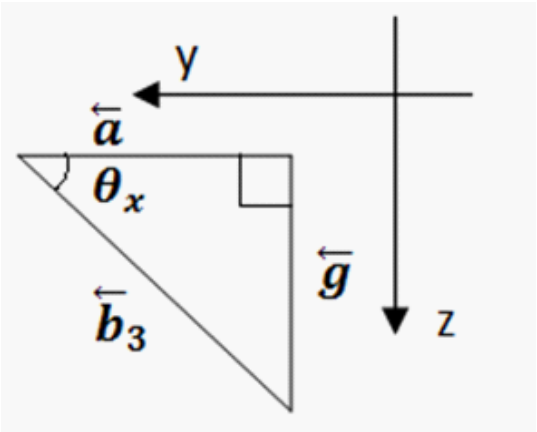
Tversnitt av modellfly med akser.



Modellfly med akser og rotasjons retning.

G.1.1 Generering av deterministiske målinger

Gitt at det er en sirkulærbane vil de deterministiske målingene kun være konstanter $\underline{\omega}_b^{bb}$ og $\underline{f}^b$. De kan finnes ved å spesifisere akselerasjonen i n rammen, for også transformere de til b rammen. Figuren under illustrerer geometrien mellom vektorene.



Geometri

Ved å definere banen til flyet, kan den deriveres ned slik at akselerasjonen kan blir funnet. Banen blir definert slik at den starter i x=radius og y lik null:

$$\underline{p}^n(t) = \begin{bmatrix} r\cos(\omega t) \\ r\sin(\omega t) \\ 0 \end{bmatrix} \tag{G- 196}$$

$$\underline{v}^n(t) = \dot{\underline{p}}^n(t) = \begin{bmatrix} -r * \omega \sin(\omega t) \\ r * \omega \cos(\omega t) \\ 0 \end{bmatrix} \tag{G- 197}$$

$$\underline{a}^n(t) = \ddot{\underline{p}}^n(t) = \begin{bmatrix} -r * \omega^2 \cos(\omega t) \\ -r * \omega^2 \sin(\omega t) \\ 0 \end{bmatrix} \tag{G- 198}$$

$r = 1000\ meter$

$\underline{f}^n = \underline{a}^n + \underline{g}^n\ der\ \underline{g}^n = [0, 0, g]^T\ og\ \omega = \omega_{yaw}^n$

Vinkelhastigheten til flyet sett fra n rammen er på 1 rotasjon rundt yaw-aksen per runde. Omkret-

stiden T er satt til 60 sekunder, det gjør at ω^n_{yaw} kan regnes ut slik $\omega^n_{yaw} = 360\,\mathrm{deg}/T = \frac{2\pi}{T}$ som gir $\omega^n_{yaw} = 6\,\mathrm{deg}/s$.Vinkel hastighets vektoren sett fra n rammen blir da: $\underline{\omega}^{nb}_b = [0, 0, \omega^n_{yaw}]^T$.

Koordinattransformasjonsmatrisen

Siden vi vet at hastighets vektoren alltid er en tangent på banen, kan koordinatransformasjonsmatrisen utrykkes som en stillingsmatrise ved hjelp av tre enhetsvektorer som står vinkelrett på hverandre. Slik, $R^b_n = [\underline{b}^T_1;\ \underline{b}^T_2;\ \underline{b}^T_3]$. Der $\underline{b}_1$ er enhetsvektoren til hastighetsvektoren. $\underline{b}_2$ blir da definert som enhetsvektoren til krysproduktet av akselerasjonsvektoren og $\underline{b}_1$. Tilslutt blir da $\underline{b}_3$ en enhetsvektor som står vinkel rett på $\underline{b}_1,\ \underline{b}_2$ planet, som er krysproduktet mellom $\underline{b}_1$ og $\underline{b}_2$.

$$\underline{b}_1 = \frac{\underline{v}^n}{||\underline{v}^n||} \quad \text{(G- 199)}$$

$$\underline{b}_2 = \frac{\underline{f}^n x \underline{b}_1}{||\underline{f}^n x \underline{b}_1||} \quad \text{(G- 200)}$$

$$\underline{b}_3 = \frac{\underline{b}_1 x \underline{b}_2}{||\underline{b}_1 x \underline{b}_2||} \quad \text{(G- 201)}$$

$$R^b_n(t) = \begin{bmatrix} \underline{b}^T_1(t) \\ \underline{b}^T_2(t) \\ \underline{b}^T_3(t) \end{bmatrix}$$

Når transforamsjonsmatrisen fra n rammen til b rammen er funnet. Finner man målt vinkelhastigheten og akslerasjonen sett fra b rammen ved å transformere $\underline{\omega}^{nb}_b$ og $\underline{f}^n$ slik:

$$\underline{f}^b = R^b_n \underline{f}^n \quad \text{(G- 202)}$$

$$\underline{\omega}^{bb}_b = R^b_n \underline{\omega}^{nb}_b \quad \text{(G- 203)}$$

G.1.2 Psudokode

$\Delta t = 1/100;$

$t_k = 0 : \Delta t : 180;$

$\omega = \frac{2\pi}{60};$

$g = 9.81;$

$N = length(t);$

$r = 1000;$

$\underline{p}^n_k = \left[\ r\cos(\omega t_k);\ r\sin(\omega t_k); 0 * t_k\ \right];$

$\underline{v}^n_k = \dot{\underline{p}}^n_k = \left[\ -r * \omega \sin(\omega t_k);\ r * \omega\cos(\omega t_k); 0 * t_k\ \right];$

$\underline{a}^n_k = \ddot{\underline{p}}^n_k = \left[\ -r * \omega^2\cos(\omega t_k);\ -r * \omega^2 \sin(\omega t_k)\ ;\ 0 * t_k\ \right];$

$for\ k = 1 : N$

$\quad \underline{\omega}^{nb}_b = [0; 0; \omega];$

$\quad \underline{f}^n = \underline{a}^n(k) + \underline{g}^n;$

$\quad \underline{b}_1 = \frac{\underline{v}^n(k)}{\sqrt{v^n_1(k)^2 + v^n_2(k)^2 + v^n_3(k)^2}};$

$\quad temp = S(\underline{f}^n) * \underline{b}_1;$

$\underline{b}_2 = \frac{temp}{||temp||}$;

$temp = S(\underline{b}_1) * \underline{b}_2$;

$\underline{b}_3 = \frac{temp}{||temp||}$;

$R_n^b = \left[\underline{b}_1^T; \underline{b}_2^T; \underline{b}_3^T\right]$;

$\underline{f}_k^b = R_n^b \underline{f}^n$;

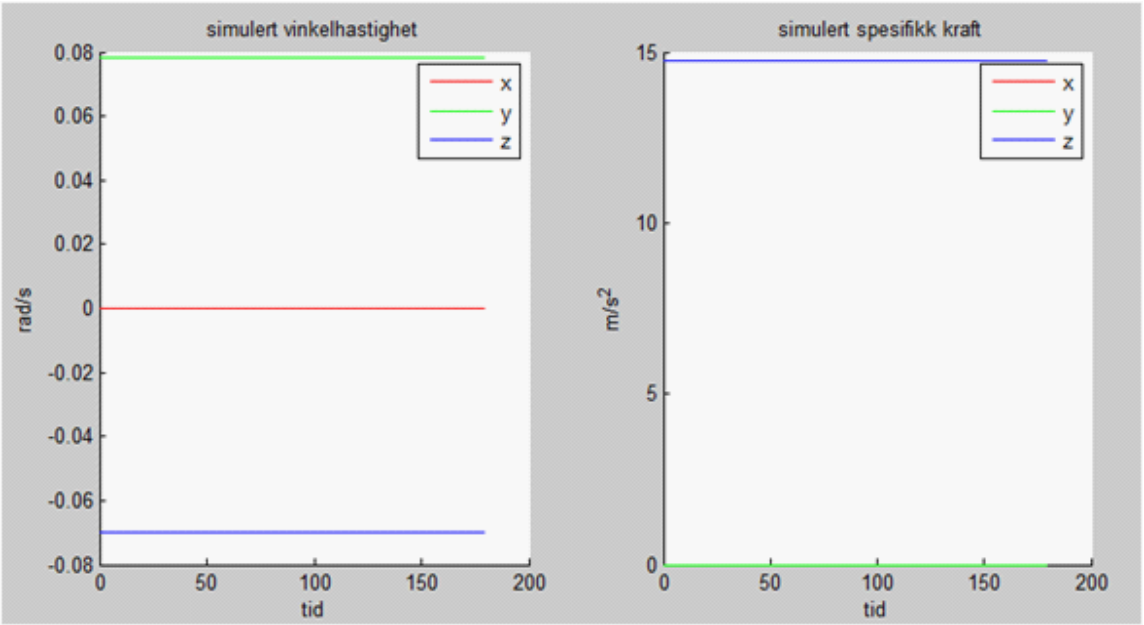
$\underline{\omega}_{b,k}^{bb} = R_n^b \underline{\omega}_b^{nb}$;

end

$plot(result)$

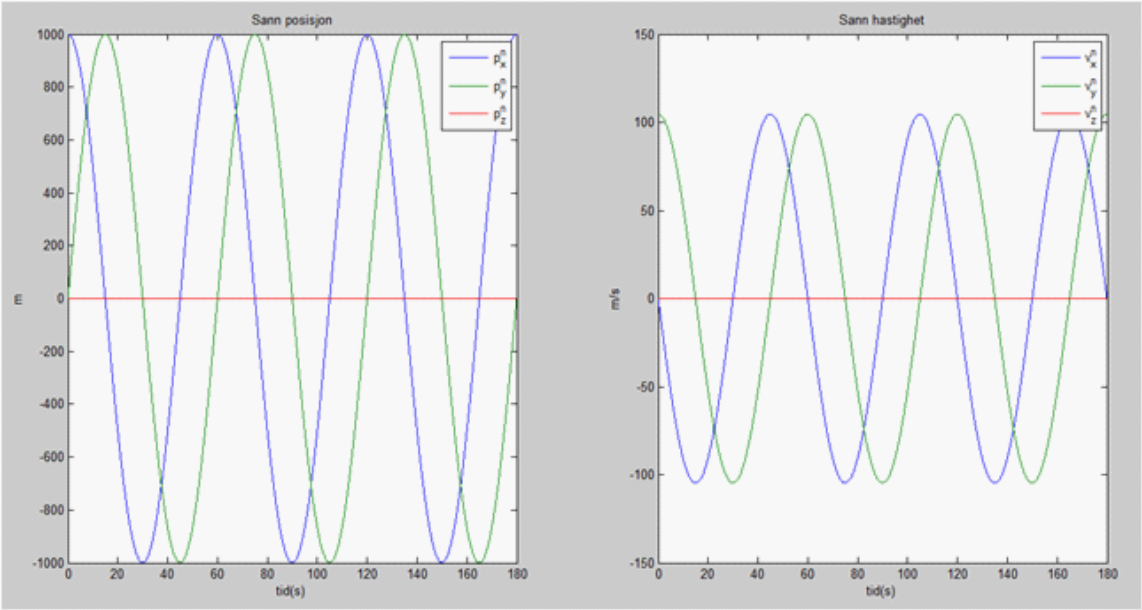
G.1.3 Resultat fra den deterministiske banegenereringen for modellfly

Den simulerte akselerasjonen og vinkelhastigheten blir som forventet en konstant, der simuleringstiden er 180s.



Generert vinkelhastighet og akselerasjons målinger.

Posisjonslikningene gir følgende posisjon og hastighet.



Sann posisjon og sann hastighet.

G.1.4 Deterministisk simuleringer av TNS

Deterministiske simuleringer blir gjort som proof of consept simuleringer, for å bekrefte at kalman filteret fungerer. Simuleringene blir gjort med de samme verdiene på standardavvik og spektraltettheter som det ble konkludert med at fungerte optimalt på mobiltelefonsensorene, men det blir simulert uten tilbakekobling. Standardavviket til usikkerheten i måleoppdateringen blir satt til 5m siden et bra sivilt GPS system vil gi en nøyaktighet på ca 5 meter.

De verdiene som følger:

$\Delta t = 1/100$

Spektraltettheter og standardavvik for kalman filteret:

$q_a = (\frac{1}{40} mg/s)^2,\ q_a^\gamma = (\frac{1}{40} mg/s)^2,\ q_a^\beta = (\frac{1}{40} mg/s)^2$

$q_g = (\frac{1}{120} \deg/s)^2,\ q_g^\gamma = (\frac{1}{120} \deg/s)^2,\ q_g^\beta = (\frac{1}{120} \deg/s)^2$

$Rd = 0.001^2$

Initalbetingelsene til posisjon, hastighet og orientering blir:

$p(t_0) = [r; 0; 0]$

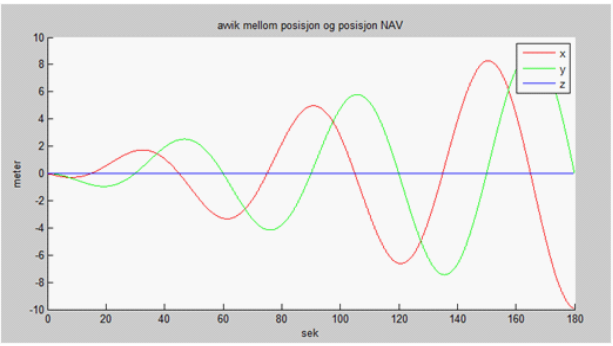
$v(t_0) = [0; \sqrt{||f^b(t_0)|| * r}; 0]$

$R_b^n(t_0) = R_{321}(\theta(t_0))$

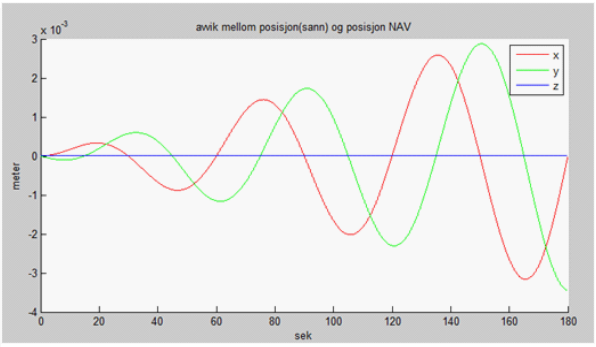
$\theta(t_0) = [\arccos(\frac{g}{||f^b(t_0)||}), 0, -\frac{\pi}{2}]^T$

Resultat av deterministiske simuleringer

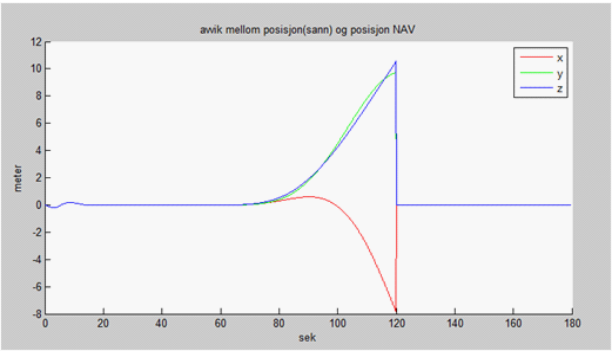
De deterministiske simuleringene viser at en navigasjonssimulering etter Euler likningene, uten støy, vill gi en drift naturlig integrerings drift på ca 10 meter etter 180 sekunder. Euler-Huens likningene forbedrer det resultatet til en drift på ca. 4 millimeter etter 180 sekunder. Det kan ses i figurene under. Simuleringen av TNS'et med Kalman filteret og euler huens likningene gir en drift som øker gradvis til ca 10 meter, driften er ikke optimal men er et resulat av relativt store initial betingelser til kamlan filtert.



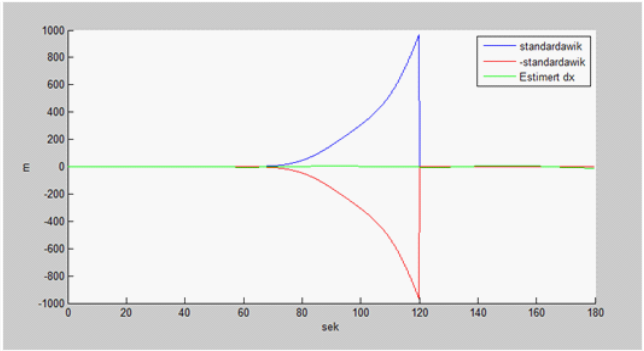
Euler NAV error



Euler-Heuns Nav, error.



Euler-Huens NAV with Kalman, error.



Estimert posisjons feil

G.1.5 Stokastiske simuleringer av TNS

Fra de deterministiske simuleringene ser man at TNS'et fungerer, men det kan gjøres noen forbedringer. En samplingsfrekvens på 100 hz er unødvendig mye da det tar opp mye prosseror kraft og ville hvert et fordyrende ledd i et realisert system og senkes da til 20 Hz. Måleoppdatering hver sample er urealistisk med måleoppdateringene basert på et GPS system og senkes til 1Hz. Støyen som blir generert er også urealistisk, da kovariansanaylsen fra mobiltelefonsensormodelleringen viser at de fargete støyelementene gir et ganske stor bidrag til feilen. Jørn S. Grahn skrev en masteroppgave, Estimering av MEMS-gyroparametere[4]. Der det ble målt at farget støyelementene i MEMS-gyrometre er veldig små. Dermed senkes gitt elementer med en faktor på 10^{-4}. Samtidig som den deterministiske simuleringen viser at driften er alt for stor og må minskes. Initialbetingelser banegenerering, støygenering og kalman filteret settes da til:

Banegenerering:

$\Delta t = 1/20$

Støygenering:

$\sigma_a = \frac{1}{40} mg/s,\ \sigma_a^\gamma = \frac{1}{40} 10^{-4} mg/s,\ \sigma_a^\beta = \frac{1}{40} 10^{-4} mg/s$

$\sigma_g = \frac{1}{120} \deg/s,\ \sigma_g^\gamma = \frac{1}{120} 10^{-4} \deg/s,\ \sigma_g^\beta = \frac{1}{120} 10^{-4} \deg/s$

Euler og Euler-Heuns NAV. int:

$p(t_0) = [r; 0; 0]$

$v(t_0) = [0; \sqrt{||f^b(t_0)|| * r}; 0]$

$R_b^n(t_0) = R_{321}(\theta(t_0))$

[4] Jørn Skarbø Grahn (2011), Estimering av MEMS-gyroparamtre. p. 57

$\theta(t_0) = [\arccos(\frac{g}{||f^b(t_0)||}), 0, -\frac{\pi}{2}]^T$

Kalmanfilter int:

$q_a = (\frac{1}{40} mg/s)^2,\ q_a^\gamma = (\frac{1}{40} 10^{-4} mg/s)^2,\ q_a^\beta = (\frac{1}{40} 10^{-4} mg/s)^2$

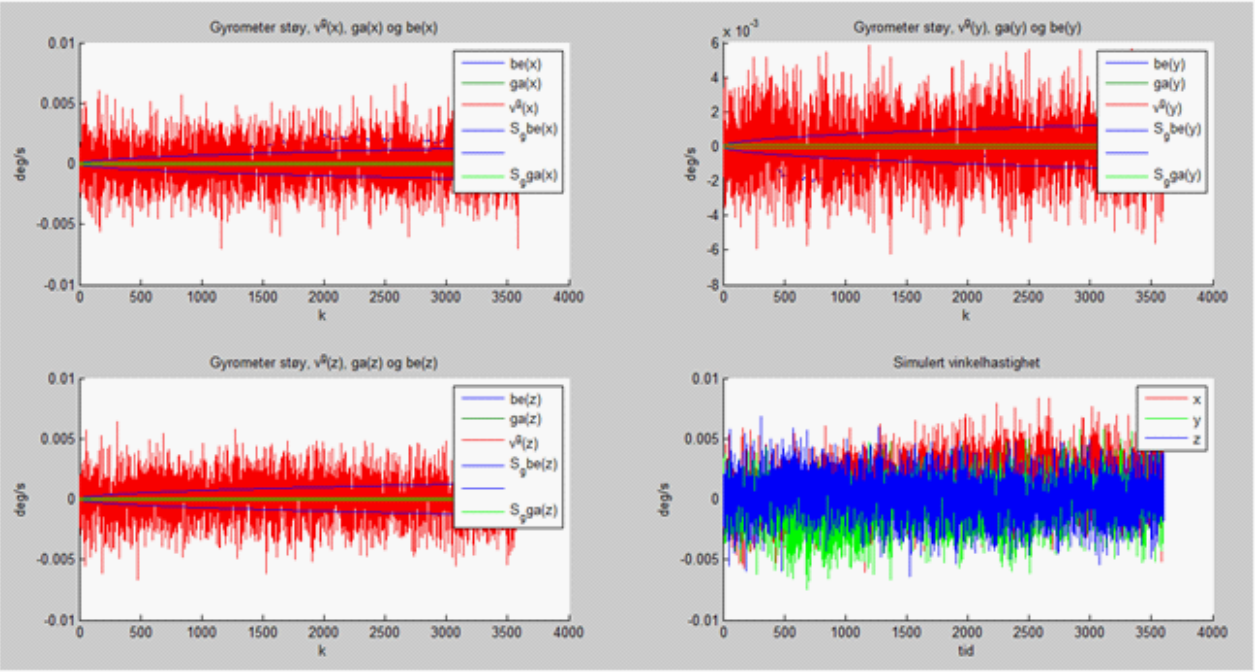
$q_g = (\frac{1}{120} \deg/s)^2,\ q_g^\gamma = (\frac{1}{120} 10^{-4} \deg/s)^2,\ q_g^\beta = (\frac{1}{120} 10^{-4} \deg/s)^2$

$Rd = 5^2$

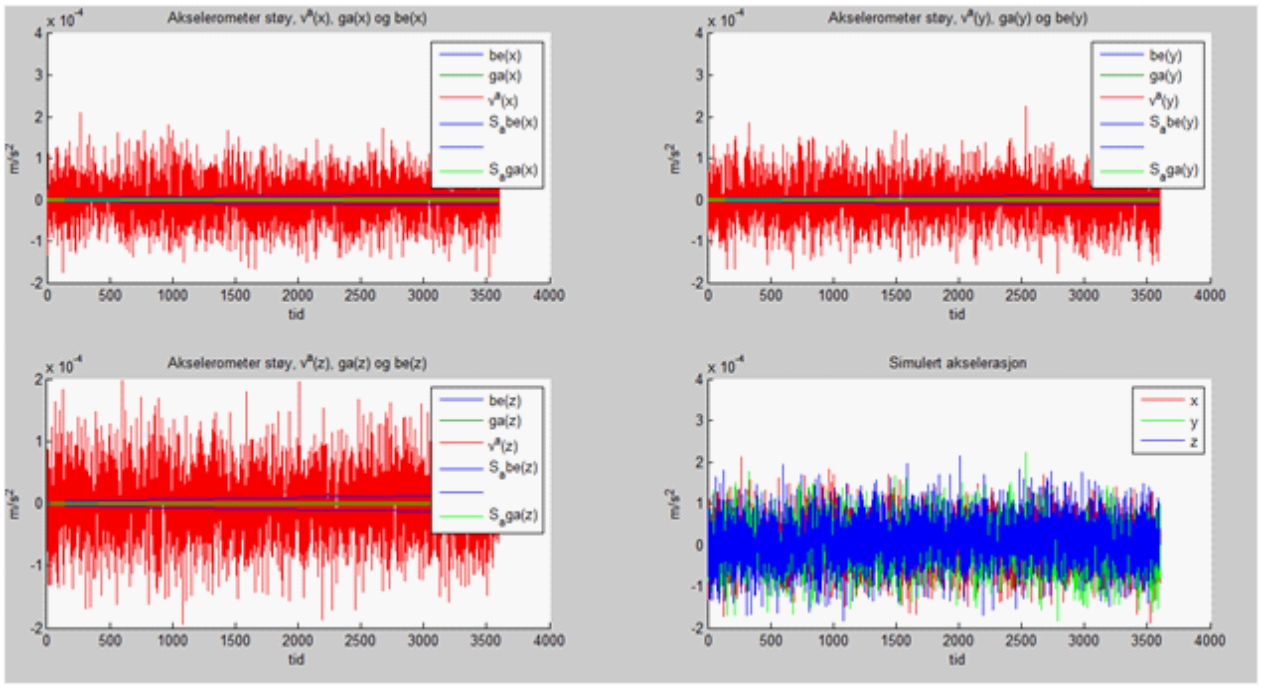
$MO = [k_0 : 20 : k_{60},\ k_{120} : 20 : k_{180}]$

G.1.6 Resultat fra stokastiske simuleringer

Støyen er blitt mer realistisk å viser store bidrag fra hvitsøyelementene og et lite bidrag fra farget-støyelementene.

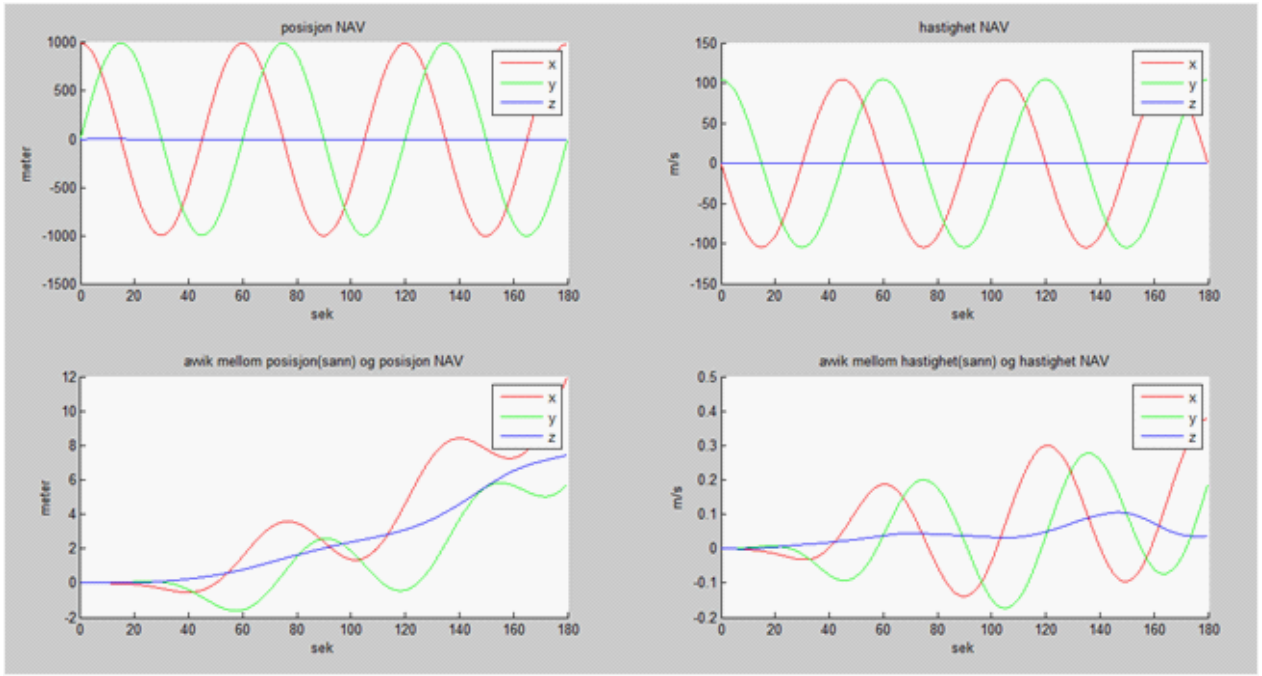


Gyrometerstøy

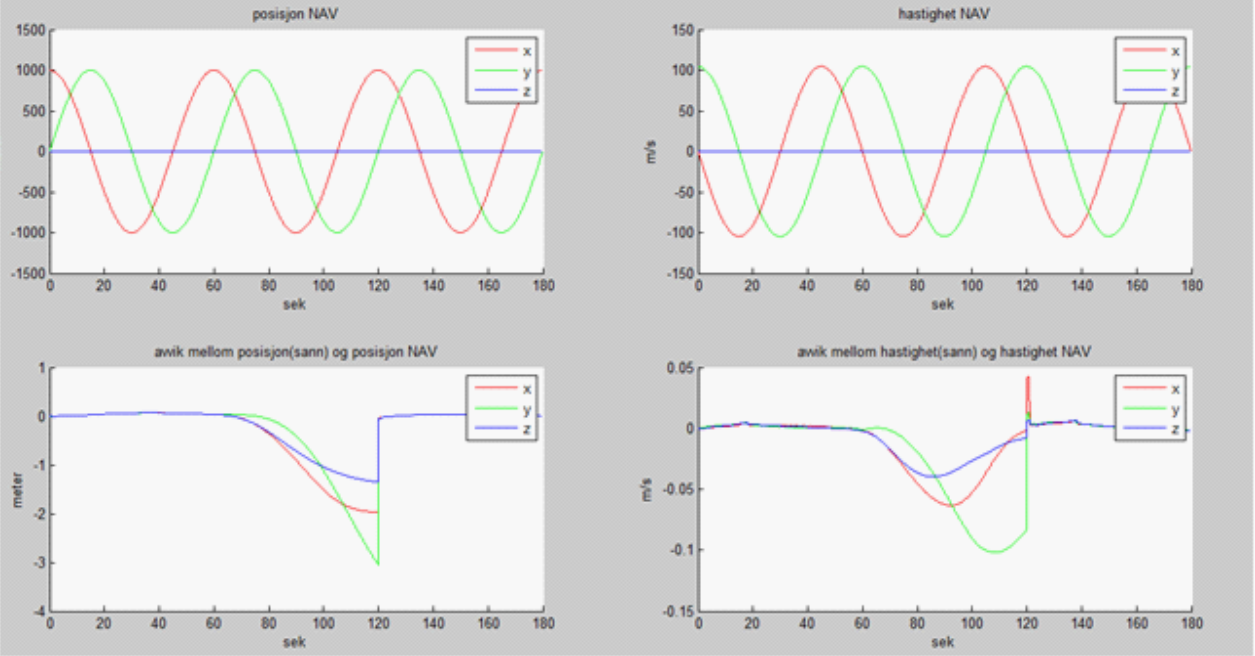


Akselerometerstøy

Navigasjonene med og uten kalman filteret med Euler-Heuns likningene viser en veldig stor forbedring mens måleoppdateringene er på, men når Kalmanfilteret ikke blir oppdatert ser man at avvik blir stort relativt fort. Det kan skyldes for dårlige sensorer eller uoptimale initiall betingelser.



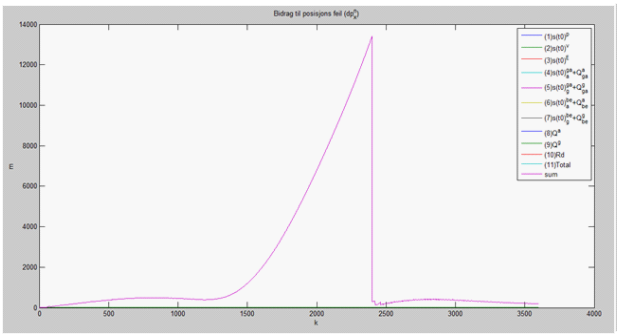
Euler-Heuns Nav



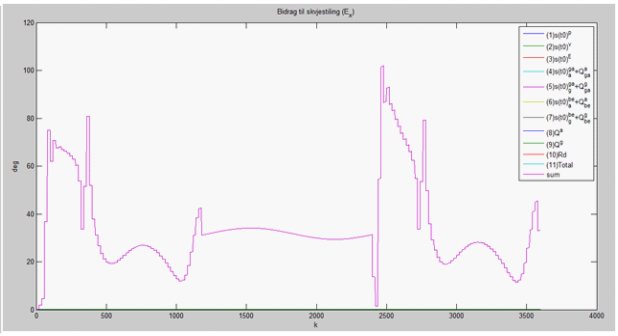
Kalman Euler-Heuns Nav

Del H:
Kovariansanalyse av TNS for modelfly

Programmet er feilbudsjettprogrammet som ble laget for analyse av mobiltelefonbasert TNS, feilgruppene er dermed de samme. Kovariansanalysen er gjort på stokastiske simuleringen fra siste avsnitt i forrige kapittel. Resulatet kan ses i figurene under.

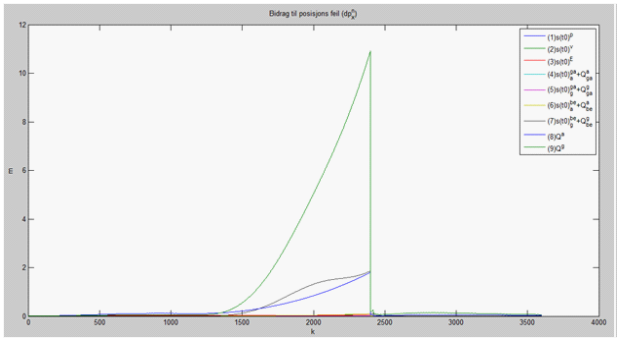


Kovariansanalyse, posisjon

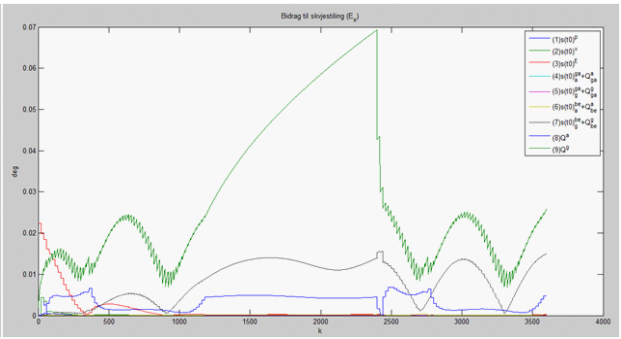


Kovariansanalyse, orientering

Her ser man at feilgruppen nr 9, dvs usikkerheten i nøyaktigheten til måleoppdateringen. Gir et så stor bidrag at de andre feilgruppene ikke kan ses i disse plotene. Ved å utelate de fra plottene vil det gi et bedre utgangspunkt for analysen.



Kovariansanalyse, posisjon oppdater plot



Kovariansanalyse, orientering. Oppdatert plot

Plottene over viser mye av de samme resultatene som fra kovariansanalysen for TNS'et bassert på mobiltelefon sensorer. Gyrometersensorene gir et stort bidrag til feilen. En lav samplingstid vil også gi et unøyaktigsystem. Når det korigeres for det resultatet og kalmanfilteret tilbakekobles. Slik at samplingfrekvensen $\Delta t = 40Hz$, måleoppdaterings frekvensen er fortsatt $1Hz$ og litt bedre gyroskop med standaravvik på hvitstøyelementene gitt av $\sigma_\gamma^g = 1 * 10^{-5}\,\mathrm{deg}/s$, $\sigma_\beta^g = 1 * 10^{-5}\,\mathrm{deg}/s$ *og* $\sigma^g = 1/150\ \ \mathrm{deg}/s$.Disse endringene gir et system som oppfyller kravene på nøyaktighet. Som er en drift på maks en meter etter 60 sekunder.

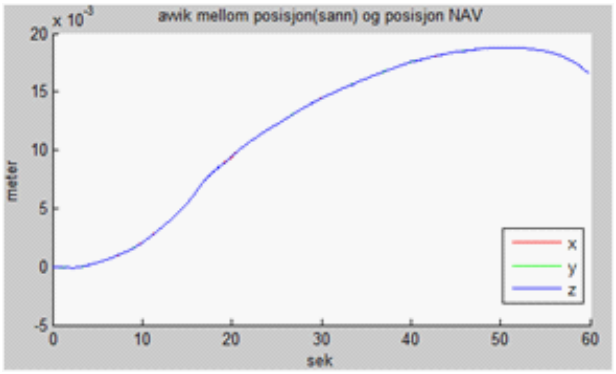
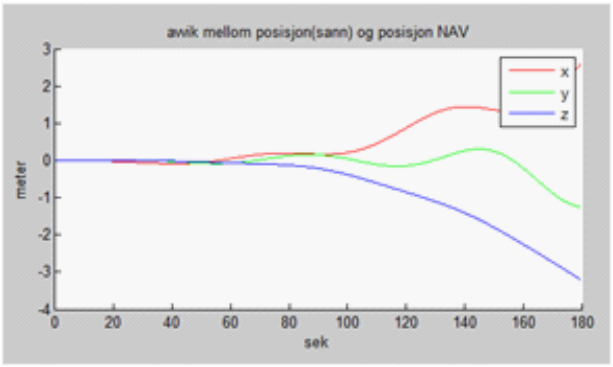
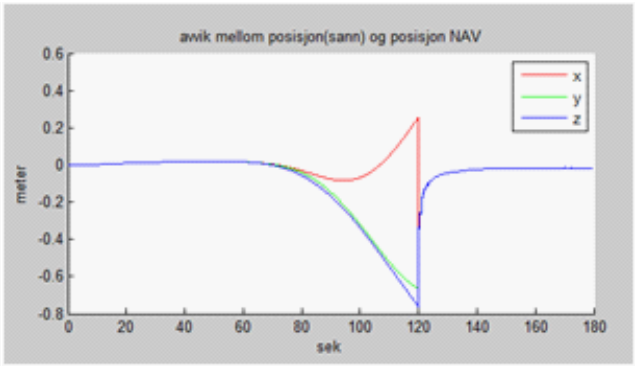


Figure 2: 0-60s. Kalman E-H Nav



Euler-Heuns Nav



Kalman Euler-Heuns Nav

Hvis man ser på driften mens det kjøres med en måleoppdatering på $1Hz$ ser man at kalman filtert redusere usikkerheten i posisjonen i forhold til start ned til milimeter i stede for 5 meter som er usikkerheten i en GPS-posisjons måling.

Banegenerering:

$\Delta t = 1/40$

Støygenerering:

$\sigma_a = \frac{1}{40} mg/s,\ \sigma_a^\gamma = 10^{-5} mg/s,\ \sigma_a^\beta = 10^{-5} mg/s$

$\sigma_g = \frac{1}{150} \deg/s,\ \sigma_g^\gamma = 10^{-5} \deg/s,\ \sigma_g^\beta = 10^{-5} \deg/s$

Euler og Euler-Heuns NAV. int:

$p(t_0) = [r; 0; 0]$

$v(t_0) = [0; \sqrt{||f^b(t_0)|| * r}; 0]$

$R_b^n(t_0) = R_{321}(\theta(t_0))$

$\theta(t_0) = [\arccos(\frac{g}{||f^b(t_0)||}), 0, -\frac{\pi}{2}]^T$

Kalmanfilter int:

$q_a = (\frac{1}{40} mg/s)^2,\ q_a^\gamma = (10^{-5} mg/s)^2,\ q_a^\beta = (10^{-5} mg/s)^2$

$q_g = (\frac{1}{150} \deg/s)^2,\ q_g^\gamma = (10^{-5} \deg/s)^2,\ q_g^\beta = (10^{-5} \deg/s)^2$

$Rd = 5^2$

$MO = [k_0 : 1/\Delta t : k_{60},\ k_{120} : 1/\Delta t : k_{180}]$

Del I:
Konklusjon og videre arbeid

I.1 Konklusjon

Oppgave har gått ut på analysere treghets navigasjons systemer, kjøre simuleringer og trekke konklusjoner på bakgrunn av de simuleringene. Det som er mest åpenbart og konkludere med er at det er mer kritisk og ha bra gyrometersensorer enn bra akselerometersensorer. Spesielt et høyt fargetstøy element i vinkelhastighet målingene vil gi et stort bidrag til driften i et TNS.

Videre ble det gjort simuleringer av to forskjellige systemer. For modellering av mobiltelefonsystemet kom det fram at måleoppdateringer på hastigheten gir et bedre resultat en måleoppdatering på posisjonen når de har samme usikkerhet i nøyaktigheten. For å nå en drift på mindre en 1 cm etter 16 sekunder, ble det gjort en kovariansanalyse som viste at standardavviket til hvitstøyelementene må ligge på rundt 6,7*10^-2 deg/s for gyrometrene og ca. 2*10^-1 mG.

Modellering av modellflysystemet viste at de optimale sensorene må ha et standardavvik i hvitstøyelementene på ca. 6,7*10^-3 deg/s i gyrostøyen, og et standardavvik på 2,5*10^-2 mG i akselerometerstøyen, med fargetstøyelementer som er mye mindre. Med Kalmanfilteret stilt inn på de parametrene, kan man også se at kalmanfilteret greier og forbedre vanlig kommersielle GPS målinger fra en usikkerhet på 5 meter til en usikkerhet på 20 millimeter.

I.2 Videre arbeid

Denne oppgaven vil være et greit grunnlag for en oppgave eller et prosjekt der det blir lagd et TNS program for implementering i en reell mobiltelefon, med måledata for gitt telefon sine sensorer. Der det blir brukt nullhastighets måleoppdateringer og målet er å måle konturene til et lite objekt, for eksempel en boks, med mobiltelefonen.

I denne oppgaven har det blitt antatt at det navigeres på en flat ikke roterende jord. Det som kan bli gjort videre er og se hvordan jordrotasjonen virker inn på gyrometermålingene og hvor stort bidrag det hadde gitt til totalfeilen til et navigasjonssystem for modelfly.

Gyrometer og akselerometer-målingene er her blitt definert som et sett med kontinuerlige målinger og deretter diskretisert. Ved å gjøre det slik kan det oppstå diskretiseringsfeil siden målingen vil egentlig være i diskrete domenet. Det som kan gjøres videre da er og definere de som diskrete målinger.

Referanser

1. Sami Hamra: *Treghetsnavigasjon implentert i Android*, Masteroppgave ved Universitetet i Oslo 2011.
2. Jørn Skarbø Grahn: *Estimering av MEMS-gyroparametre*, Masteroppgave ved Universitetet i Oslo 2011.
3. Diego Mugisha: , *TNS med lavkostsensorer,* Masteroppgave ved Universitetet i Oslo 2012.
4. Oddvar Hallingstad, Notat 7 Monte Carlo og kovarians analyse av Kalman filteret
5. Arthur Gelb, Applied Optimal Estimation

Appendiks A: BKG3.m:

```
% % ————————- Banegenrator - BGK3 ————————————————-
% Banegenerator v1.0 - uten støy. (laget Diego Mugisha.)
% v1.1.0 Lagt på realistisk støy på f og w målingene.
% v1.2.0 Forbedert kalman filteret til å kjøre med støy verdier.
% v1.3.0 Tilbakekoblet kalman filter
%
%
% v2.0 Banegenerator med støy. Laget av Petter Lefoka
%% Int
clear all; clc; close all;
% For generering av mobiltelefonbane sette bane=1
% For generering av modellflybane sette bane=2
% bane=1;
bane=1;
tilbakekobling=1;
%% Banegenerator
%Generere determenistiske ref bane
DBG;
N=length(f__b);
g__n=[0;0;g];
% Målefeil
% sa_g=[0.017453*10^(-5);0.017453*10^(-5);0.017453/150]; %standaraviket rad/s
% sa_a=[0.00981*10^(-5);0.00981*10^(-5);0.00981/40]; %standaraviket (m/s^2)/s
sa_g=[0.017453/15;0.017453/15;0.017453/15]; %standaraviket rad/s
sa_a=[0.00981/5;0.00981/5;0.00981/5]; %standaraviket (m/s^2)/s
[df,dw]=stoy(sa_a,sa_g,N,dt);
wb1_b_nb=w_b_bb+dw; % målt vinkelhastighet
fb__b=f__b+df; % målt spesifikk kraft
%% Navigasjonssimulering
% navigasjonslikninger
pb__n=zeros(3,N);
vb__n=zeros(3,N);
Rb_b_n=zeros(3,3,N);
if bane==1
```

```
    pb__n(:,1)=[0;0;0]; % Inietiell posisjon

    vb__n(:,1)=[0;0;0]; % Initiell hastighet
    evb=[0;0;0]; % Initielle eulervinkler
    Rb_b_n(:,:,1) = e2R(evb); % Initiell RKM
else
   % pb__n(:,1)=[-1000,0,0]';
   % vb__n(:,1)=[0;-105;0];
    pb__n(:,1)=p__n(:,1);
    vb__n(:,1)=v__n(:,1);
   Rb_b_n(:,:,1)=R_321(-[acos(g/f__b(3,1));0;-pi/2]);
end
% % essai
evbm=[0;0;0];
%% Eulers metode
for k=1:N-1
    pb__n(:,k+1)=pb__n(:,k)+dt*vb__n(:,k);
    vb__n(:,k+1)=vb__n(:,k)+dt*(Rb_b_n(:,:,k)*f__b(:,k)-g__n);
    % Beregn eulervinkler
    Rb_b_n(:,:,k+1)=Rb_b_n(:,:,k)+dt*(Rb_b_n(:,:,k)*S(w_b_bb(:,k)));
    evb(:,k+1)=R2e(Rb_b_n(:,:,k+1));
end
% avvik mellom 'posisjon-p__n 'og 'posisjon(NAV)-p_b_n'—(euler's metode)
dv__n=v__n-vb__n;
dp__n=p__n-pb__n;
% Her kjøres Eulers metode uten støy for og se at metoden virker i
% prinsipp.
%% plot av NAV (euler)
plot_nav_euler
%
%% Heuns metode
% ————————-Nav - euler huens————————————-
% Alle 'merket' variabler er k+1 i tid.
for k=1:N-1
    pbm=pb__n(:,k)+dt*vb__n(:,k);
    vbm=vb__n(:,k)+dt*(Rb_b_n(:,:,k)*fb__b(:,k)-g__n);
    %3-2-1 Eulervinkler
    Rbm_b=Rb_b_n(:,:,k)+dt*Rb_b_n(:,:,k)*S(wb1_b_nb(:,k));
```

```
        evbm=R2e(Rbm_b);
        pb__n(:,k+1)=pb__n(:,k)+(dt/2)*(vb__n(:,k)+vbm);
        vb__n(:,k+1)=vb__n(:,k)+(dt/2)*(Rb_b_n(:,:,k)*fb__b(:,k)-g__n+Rbm_b*fb__b(:,k+1)-
g__n);
        %3-2-1 Eulervinkler
        Rb_b_n(:,:,k+1)=Rb_b_n(:,:,k)+(dt/2)*(Rb_b_n(:,:,k)*S(wb1_b_nb(:,k))+Rbm_b*S(wb1_b_nb(:,
        evb(:,k+1)=R2e(Rb_b_n(:,:,k+1));
end
dv1__n=v__n-vb__n;
dp1__n=p__n-pb__n;
plot_nav_heun
%% Kalmanfilter
I3=eye(3);
c=[1 1 1];
g2r=pi/180;
r2g=180/pi;
T__a=2;
T__g=1;
% x=Fx+Gv
% z=Hx+w
% Systemmatriser
% F og G har tidsinvariante undermatriser
F=zeros(21,21);
F(1:3,4:6)=I3;
F(10:12,10:12)=-1/T__a*I3;
F(13:15,13:15)=-1/T__g*I3;
G=zeros(21,18);
G(10:21,7:18)=eye(12);
% Initiell kovarians
% % bane 1
sp_0_n=0.01; % [p]=m
sv_0_n=0.01; % [v]=m/s
sep_0_n=0.1*g2r; % [ep]=rad
sbe_0_a=0.001/5*g; % [be__a]=m/s^2
sga_0_a=0.001/5*g; % [ga__a]=m/s^2
sbe_0_g=0.01/15*g2r; % [be__g]=rad/s
sga_0_g=0.01/15*g2r; % [ga__g]=rad/s
% % Bane 2
```

Appendix A BKG3.m:

```
% sp_0_n=0.01; % [p]=m
% sv_0_n=0.01; % [v]=m/s
% sep_0_n=0.1*g2r; % [ep]=rad
% sbe_0_a=0.001*10^(-5)*g; % [be__a]=m/s^2
% sga_0_a=0.001*10^(-5)*g; % [ga__a]=m/s^2
% sbe_0_g=0.01*10^(-5)*g2r; % [be__g]=rad/s
% sga_0_g=0.01*10^(-5)*g2r; % [ga__g]=rad/s
% p v ep ga_a ga_g be_a be_g
Pe_0=diag([(sp_0_n^2)*c,(sv_0_n^2)*c,(sep_0_n^2)*c,(sga_0_a^2)*c,(sga_0_g^2)*c,(sbe_0_a^2)*c,(sbe
Pe=dt*Pe_0;
% Initiell spektraltettheter
qb__a=(0.00981/5)^2;
qb__g=(0.017453/15)^2;
qb_be_a=(0.00981/2)^2;
qb_ga_a=2*sga_0_a^2/T__a;
qb_be_g=(0.017453/15)^2;
qb_ga_g=2*sga_0_g^2/T__g;
% Bane 2
% qb__a=(0.00981/40)^2;
% qb__g=(0.017453/150)^2;
% qb_be_a=(0.00981*10^(-5))^2;
% qb_ga_a=2*sga_0_a^2/T__a;
% qb_be_g=(0.017453*10^(-5))^2;
% qb_ga_g=2*sga_0_g^2/T__g;
% v_a v_g v_ga_a v_ga_g v_be_a v_be_g
Qb=diag([qb__a*c,qb__g*c,qb_ga_a*c,qb_ga_g*c,qb_be_a*c,qb_be_g*c]);
% dz1=-pb__n; % Posisjonsmåling
H1=[I3 zeros(3,18)];
if bane==1
    Rd1=(0.001/5)^2;
else
    Rd1=(5)^2;
    H=H1;
end
% dz2=-vb__n; % Hastighetsmåling
H2=[zeros(3,3) I3 zeros(3,15)];
Rd2=(0.001/15)^2;
```

```
% H2=[eye(6) zeros(6,15)];
% ———Kalmanfilter———
dxe0=zeros(21,1);
dxe=dxe0;
dxem=zeros(21,N);
dxem(:,1)=dxe0;
eps(1:3,1)=[0,0,0]';
s(:,1)=sqrt(diag(Pe));
if bane==1;
    MOH1=[1:200];
    MOH2=[401:600 801:1000 1201:1400 1601:1800];
else
    MOH1=[1:20:(60/dt) (120/dt+1):20:(180/dt)];
    MOH2=0;
    %intial posisjons betingelser
    pb__n(:,1)=[1000,0,0]';
    vb__n(:,1)=[0;104.7198;0];
    Rb_b_n(:,:,1)=R_321(-[acos(g/f__b(3,1));0;-pi/2]);
end
for k=1:N-1
    if tilbakekobling == 1
        pb__n(:,k)=pb__n(:,k)+dxe(1:3);
        vb__n(:,k)=vb__n(:,k)+dxe(4:6);
        Rb_b_n(:,:,k)=(eye(3)+S(dxe(7:9)))*Rb_b_n(:,:,k);
        % fb__b(:,k)=fb__b(:,k)+dxe(10:12)+dxe(16:18);
        % wb1_b_nb(:,k)=wb1_b_nb(:,k)+dxe(13:15)+dxe(19:21);
        dxe(1:9)=zeros(9,1);
    end
    % ——————-Nav - euler huens ————————————
    pbm=pb__n(:,k)+dt*vb__n(:,k);
    vbm=vb__n(:,k)+dt*(Rb_b_n(:,:,k)*fb__b(:,k)-g__n);

    %3-2-1 Eulervinkler
    Rbm_b=Rb_b_n(:,:,k)+dt*Rb_b_n(:,:,k)*S(wb1_b_nb(:,k));
    evbm=R2e(Rbm_b);

    pb__n(:,k+1)=pb__n(:,k)+(dt/2)*(vb__n(:,k)+vbm);
```

```
vb__n(:,k+1)=vb__n(:,k)+(dt/2)*(Rb_b_n(:,:,k)*fb__b(:,k)-g__n+Rbm_b*fb__b(:,k+1)-
g__n);

%3-2-1 Eulervinkler
Rb_b_n(:,:,k+1)=Rb_b_n(:,:,k)+(dt/2)*(Rb_b_n(:,:,k)*S(wb1_b_nb(:,k))+Rbm_b*S(wb1_b_nb(:,k+
evb(:,k+1)=R2e(Rb_b_n(:,:,k+1));

% ———— Nav slutt ————————————

% ———— Tidsoppdatering - TO ————————
% F og G oppdateres før diskretisering
% — fb__b=f__b+ga_a+be_a+støy
% — wb1_b_nb=w_b_nb+ga_g+be_b+støy

%————————————————————————————-
% Rotasjonsmatrisen 3-2-1
R=Rb_b_n(:,:,k);
%————————————————————————————-
F(4:6,7:9)=-S(R*(fb__b(:,k)));
F(4:6,10:12)=R;
F(4:6,16:18)=R;
F(7:9,13:15)=R;
F(7:9,19:21)=R;
G(4:6,1:3)=R;
G(7:9,4:6)=R;
[Fi,Ga]=k2dp(dt,F,G,Qb); %Diskretisert F og G matrisene.
dxp=Fi*dxe;
Pp=Fi*Pe*Fi'+Ga*Ga'; %Prediktert Kovarians matrise.

% Lagre verdier
dxe=dxp; %Lagres dersom det bare er TO.
Pe=Pp; %Lagres dersom det bare er TO.
if bane==1
    epsm=eye(3)-R*eye(3)';
else
    epsm=(R_n_b(:,:,k)'-R)*R';
end
eps(:,k)=[-epsm(2,3);epsm(1,3);-epsm(1,2)];
```

```
% ————————- TO slutt ————————————
 % ————————— Måleoppdatering - MO ———————————
 % ———— k+1 ————
 MO=0;
K=zeros(21, 3);
if ismember(k+1,MOH1)
    if bane==1
        dz=-pb__n(:,k+1); % når det er en MO, er [p__n-pb__n=0-dpb__n]
    else
         dz=p__n(:,k+1)-pb__n(:,k+1);
    end
    H=H1;
    Rd=Rd1;
    MO=1;
elseif ismember(k+1,MOH2)
    if bane==1
        dz=-vb__n(:,k+1); % når det er en MO, er [v__n-vb__n=0-dvb__n]
    else
        dz=v__n(:,k+1)-vb__n(:,k+1);
    end
    H=H2;
    Rd=Rd2;
     MO=1;
 end
if MO
    K=Pp*H'/(H*Pp*H'+Rd); % diskret kalmanfilterforsterkning
    dxe=dxp+K*(dz-H*dxp);
    Pe=(eye(21)-K*H)*Pp; % Estimerte kovarianse
 end
 dxem(:,k+1)=dxe;
 s(:,k+1)=sqrt(diag(Pe)); % Lagring av standarsavviket
 K__st(:,:,k)=K;
 F__st(:,:,k)=F;
 G__st(:,:,k)=G;
 H__st(:,:,k)=H;
 % ————————— MO slutt ————————————
end
```

Appendix A BKG3.m:

```
I__st(:,:)=eye(6);
% ————plot av NAV——- med heun's metode
if tilbakekobling == 0
      pb__n=pb__n+dxem(1:3,:); %dette er egentling p__n estimert
      vb__n=vb__n+dxem(4:6,:); %dette er egentling v__n estimert
end
% avvik mellom 'posisjon-p__n 'og 'posisjon(NAV)-p_b_n'—(heun's metode)
dp1__n=p__n-pb__n;
dv1__n=v__n-vb__n;
plot_nav_heun
%% Kalman plott
plot_kalman
plot_ny_bane
%% Feilbudjsett.
res=errbud(F__st,G__st,H__st,K__st,Pe_0,Qb,Rd,dt);
```

Appendiks B: DBG.m:

```
% Genereing av determenistisk Bane
%
% Funksjonen DBG er en determenistisk bane generator som gir en bane for et
% mobiltelefonsystem (bane=1) eller en bane for et modellflysystem (bane =
% 2). f og w vektorene har lengden N
if bane == 1
    % Initialisering
    % tidsintervaller
    Ni=200;
    dti=2;
    N=9*Ni;
    dt=dti/(Ni-1);
    t=(0:9*Ni)*dt;
    time=0:dt:(N-2)*dt;
    % Parametre
    w=pi;
    A=0.3*pi/2+randn(1,1)*0.1; % amplitude
    g=9.81;
    f__n=zeros(3,N); % spesifikk kraft i n-ramma
    w_b_bb=zeros(3,N); % vinkelhastistighet i n-ramma
    R_n_b=e2R([0;0;0]); % Initiell koordinattransformasjonsmatrise.
    % Bane
    p=-(A/w)*(sin(w*t(1:Ni))/w-t(1:Ni));
    v=-(A/w)*(cos(w*t(1:Ni))-1);
    a=A*sin(w*t(1:Ni));
    % 0-2s Står i ro i A
    p__n(1,1:Ni)=0;
    v__n(1,1:Ni)=0;
    a__n(1,1:Ni)=0;
    % 2-4s Flyttes fra A til B
    p__n(1,Ni+1:2*Ni)=p;
    v__n(1,Ni+1:2*Ni)=v;
    a__n(1,Ni+1:2*Ni)=a;
    % 4-6s Står i ro i B
```

```
p__n(1,2*Ni+1:3*Ni)=p__n(1,2*Ni);

% 6-8s Flyttes fra B til C
p__n(1,3*Ni+1:4*Ni)=p__n(1,2*Ni);
p__n(2,3*Ni+1:4*Ni)=p;
v__n(2,3*Ni+1:4*Ni)=v;
a__n(2,3*Ni+1:4*Ni)=a;
% 8-10s Står i ro i C
p__n(1,4*Ni+1:5*Ni)=p__n(1,4*Ni);
p__n(2,4*Ni+1:5*Ni)=p__n(2,4*Ni);
% 10-12s Flyttes fra C til D
p__n(1,5*Ni+1:6*Ni)=p__n(1,5*Ni)-p;
v__n(1,5*Ni+1:6*Ni)=-v;
a__n(1,5*Ni+1:6*Ni)=-a;
p__n(2,5*Ni+1:6*Ni)=p__n(2,5*Ni);
% 12-14s Står i ro i D
p__n(1,6*Ni+1:7*Ni)=p__n(1,6*Ni);
p__n(2,6*Ni+1:7*Ni)=p__n(2,6*Ni);
% 14-16s Flyttes fra C til D
p__n(1,7*Ni+1:8*Ni)=p__n(1,7*Ni);
p__n(2,7*Ni+1:8*Ni)=p__n(2,7*Ni)-p;
v__n(2,7*Ni+1:8*Ni)=-v;
a__n(2,7*Ni+1:8*Ni)=-a;
% 16-18s Står i ro i A
p__n(1,8*Ni+1:9*Ni)=0;
v__n(1,8*Ni+1:9*Ni)=0;
a__n(1,8*Ni+1:9*Ni)=0;
a0=randn(2,1)*0;
v0=randn(2,1)*0;
p0=randn(2,1)*0;
for k = 1:N
    a__n(1:2,k)=a__n(1:2,k)+a0; % matrise for akselerasjon
    v__n(1:2,k)=v__n(1:2,k)+(a0*t(k)+v0); % matrise for hastighet
    p__n(1:2,k)=p__n(1:2,k)+((a0*t(k)^2)/2+v0*t(k)+p0); % matrise for posisjon
    a__n(3,k)=0;v__n(3,k)=0;p__n(3,k)=0;
end
% Spesifikk kraft
f__n(1:2,1:9*Ni)=a__n(1:2,1:9*Ni);
```

```
    f__n(3,1:9*Ni)=g;
    % R_n_b = eye(3);
    f__b=R_n_b*f__n;
elseif bane==2
    dt=1/40;
    t=0:dt:180;
    w=2*pi/60;
    g=9.81;
    N=length(t);
    time=0:dt:(N-2)*dt;
    p__n=[1000*cos(w*t);1000*sin(w*t);0*t];
    v__n=[-1000*w*sin(w*t);1000*w*cos(w*t);0*t];
    a__n=[-1000*(w^2)*cos(w*t);-1000*(w^2)*sin(w*t);0*t];
    for k=1:length(t)
        w_b_nb(:,k)=[0;0;w];
        f__n(:,k)=a__n(:,k)+[0;0;g];
        b_1_n(:,k)=v__n(:,k)/sqrt(v__n(1,k)^2+v__n(2,k)^2+v__n(3,k)^2);
        temp=S(f__n(:,k))*b_1_n(:,k);
        b_2_n(:,k)=temp/sqrt(temp(1)^2+temp(2)^2+temp(3)^2);
        temp=S(b_1_n(:,k))*b_2_n(:,k);
        b_3_n(:,k)=temp/sqrt(temp(1)^2+temp(2)^2+temp(3)^2);
        R_n_b(:,:,k)=[b_1_n(:,k)';b_2_n(:,k)';b_3_n(:,k)'];
         w_b_bb(:,k)=R_n_b(:,:,k)*w_b_nb(:,k);
        f__b(:,k)=R_n_b(:,:,k)*f__n(:,k);
    end
else
    disp('Feil, bane kan kun være 1 eller 2.')
end
figure()
clf
subplot(231)
hold on;
title('simulert akselerasjon')
plot(time,a__n(1,1:size(time,2)),'red');
plot(time,a__n(2,1:size(time,2)),'green');
plot(time,a__n(3,1:size(time,2)),'blue');
legend('x','y','z');
```

```
xlabel('tid')
ylabel('m/s^2')
subplot(232)
hold on;
title(' simulert hastighet')
plot(time,v__n(1,1:size(time,2)),'red');
plot(time,v__n(2,1:size(time,2)),'green');
plot(time,v__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('tid')
ylabel('m/s')
subplot(233)
hold on;
title(' simulert posisjon')
plot(time,p__n(1,1:size(time,2)),'red');
plot(time,p__n(2,1:size(time,2)),'green');
plot(time,p__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('tid')
ylabel('meter')
subplot(234)
hold on;
title('simulert vinkelhastighet')
plot(time,w_b_bb(1,1:size(time,2)),'red');
plot(time,w_b_bb(2,1:size(time,2)),'green');
plot(time,w_b_bb(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('tid')
ylabel('rad/s')
subplot(235)
hold on;
title(' simulert spesifikk kraft')
plot(time,f__b(1,1:size(time,2)),'red');
plot(time,f__b(2,1:size(time,2)),'green');
plot(time,f__b(3,1:size(time,2)),'blue');
legend('x','y');
legend('z')
```

```
legend('x','y','z');
xlabel('tid')
ylabel('m/s^2')
```

Appendiks C: stoy.m:

```
function [df,dw]=stoy(sa_a,sa_g,N,dt)
    % T=dt;
    % Ts=0.04;
    T__g=1; % Tidskonstant for gyrometer
    T__a=2; % Tidskonstant for akselerometer
    time=0:dt:(N-2)*dt;
    if length(sa_a)==1
        P_gamma_a=diag([sa_a^2,sa_a^2,sa_a^2]); %P_ga_a(0)
        P_beta_a=diag([sa_a^2,sa_a^2,sa_a^2]); %P_be_a(1)
        R_a=sa_a^2;
    else
        P_gamma_a=diag([sa_a(1)^2,sa_a(1)^2,sa_a(1)^2]); %P_ga_a(0)
        P_beta_a=diag([sa_a(2)^2,sa_a(2)^2,sa_a(2)^2]); %P_be_a(1)
        R_a=sa_a(3)^2;
    end
    if length(sa_g)==1
        P_gamma_g=diag([sa_g^2,sa_g^2,sa_g^2]); %P_ga_g(0)
        P_beta_g=diag([sa_g^2,sa_g^2,sa_g^2]); %P_be_g(1)
        R_g=sa_g^2;
    else
        P_gamma_g=diag([sa_g(1)^2,sa_g(1)^2,sa_g(1)^2]); %P_ga_g(0)
        P_beta_g=diag([sa_g(2)^2,sa_g(2)^2,sa_g(2)^2]); %P_be_g(1)
        R_g=sa_g(3)^2;
    end
    tilde_Q_gamma_g=(2/T__g)*diag(P_gamma_g);
    tilde_Q_beta_g=diag(P_beta_g);
    tilde_Q_gamma_a=(2/T__a)*diag(P_gamma_a);
    tilde_Q_beta_a=diag(P_beta_a);
    %Diskretisering
    P_gamma_a=dt*P_gamma_a;
    P_gamma_g=dt*P_gamma_g;
    P_beta_g=zeros(3,3);
    P_beta_a=zeros(3,3);
    %(d/dt)x=Fx+Gv
```

Appendix C stoy.m:

```
F__g=diag([0,0,0,-1/T__g,-1/T__g,-1/T__g]); % F matrise for gyrometer

F__a=diag([0,0,0,-1/T__a,-1/T__a,-1/T__a]); % F matrise for akselerometer
G=eye(6);
% Setter x_g_0 og x_a_0
% X__g(1:3,1)=chol(P0_beta_g,'lower')*randn(3,1);
X__g(1:3,1)=zeros(3,1);
X__g(4:6,1)=chol(P_gamma_g,'lower')*randn(3,1);
% X__a(1:3,1)=chol(P0_beta_a,'lower')*randn(3,1);
X__a(1:3,1)=zeros(3,1);
X__a(4:6,1)=chol(P_gamma_a,'lower')*randn(3,1);
% Diskretisering:
Q_beta_g=diag(tilde_Q_beta_g*dt); % Random walk variable
Q_gamma_g=diag((tilde_Q_gamma_g*T__g/2)*(1-exp(-2*dt/T__g)));
Q_beta_a=diag(tilde_Q_beta_a*dt); % Random walk variable
Q_gamma_a=diag((tilde_Q_gamma_a*T__a/2)*(1-exp(-2*dt/T__a)));
Fi__g=expm(F__g*dt);
Qb_g(1:3,1:3)=diag(tilde_Q_beta_g);
Qb_g(4:6,4:6)=diag(tilde_Q_gamma_g);
Ga__g=kp2dpga(F__g,G,Qb_g,dt);
 Fi__a=expm(F__a*dt);
Qb_a(1:3,1:3)=diag(tilde_Q_beta_a);
Qb_a(4:6,4:6)=diag(tilde_Q_gamma_a);
Ga__a=kp2dpga(F__a,G,Qb_a,dt);
 % Simulering av støymodell
% X : [be ga]'
% P_g=[P_beta_g,eye(3);eye(3),P_gamma_g];
% P_a=[P_beta_a,eye(3);eye(3),P_gamma_a];
for k=1:N-1
      P_beta_g=P_beta_g+Q_beta_g;
      P_gamma_g=(1-2*dt/T__g)*P_gamma_g+Q_gamma_g;
      P_beta_a=P_beta_a+Q_beta_a;
       P_gamma_a=(1-2*dt/T__a)*P_gamma_a+Q_gamma_a;
      X__g(:,k+1)=Fi__g*X__g(:,k)+Ga__g*randn(6,1);
      X__a(:,k+1)=Fi__a*X__a(:,k)+Ga__a*randn(6,1);
      S_g(:,k)=sqrt([diag(P_beta_g);diag(P_gamma_g)]);
      S_a(:,k)=sqrt([diag(P_beta_a);diag(P_gamma_a)]);
end
```

```
dw=X__g(1:3,1:N)+X__g(4:6,1:N)+sqrt(R_g*dt)*randn(3,N); % målt vinkelhastighet
df=X__a(1:3,1:N)+X__a(4:6,1:N)+sqrt(R_a*dt)*randn(3,N); % målt spesifikk kraft
%% plott
%plot_støy_beta
figure()
subplot(221)
hold on,
title('Akselerometer støy, v^a(x), ga(x) og be(x)')
plot([X__a(1,:);X__a(4,:);(df(1,:)-X__a(1,:)-X__a(4,:))]')
plot([S_a(1,:);-S_a(1,:)]','b')
plot([S_a(4,:);-S_a(4,:)]','g')
legend('be(x)','ga(x)','v^a(x)','S_abe(x)','','S_aga(x)')
xlabel('tid')
ylabel('m/s^2')
subplot(222)
hold on,
title('Akselerometer støy, v^a(y), ga(y) og be(y)')
plot([X__a(2,:);X__a(5,:);(df(2,:)-X__a(2,:)-X__a(5,:))]')
plot([S_a(2,:);-S_a(2,:)]','b')
plot([S_a(5,:);-S_a(5,:)]','g')
legend('be(y)','ga(y)','v^a(y)','S_abe(y)','','S_aga(y)')
xlabel('tid')
ylabel('m/s^2')
subplot(223)
hold on,
title('Akselerometer støy, v^a(z), ga(z) og be(z)')
plot([X__a(3,:);X__a(6,:);(df(3,:)-X__a(3,:)-X__a(6,:))]')
plot([S_a(3,:);-S_a(3,:)]','b')
plot([S_a(6,:);-S_a(6,:)]','g')
legend('be(z)','ga(z)','v^a(z)','S_abe(z)','','S_aga(z)')
xlabel('tid')
ylabel('m/s^2')
subplot(224)
hold on;
title('Simulert akselerasjon')
plot(df(1,:),'red');
plot(df(2,:),'green');
```

```
    plot(df(3,:),'blue');
    legend('x','y','z');
    xlabel('tid'), ylabel('m/s^2')
    figure()
    subplot(221)
    hold on,
    title('Gyrometer støy, v^g(x), ga(x) og be(x)')
    plot([(180/pi)*X__g(1,:);(180/pi)*X__g(4,:);(180/pi)*(dw(1,:)-X__g(1,:)-X__g(4,:))]')
    plot([(180/pi)*S_g(1,:);-(180/pi)*S_g(1,:)]','b')
    plot([(180/pi)*S_g(4,:);-(180/pi)*S_g(4,:)]','g')
    legend('be(x)','ga(x)','v^g(x)','S_gbe(x)','','S_gga(x)')
    xlabel('k'), ylabel('deg/s')
    subplot(222)
    hold on,
    title('Gyrometer støy, v^g(y), ga(y) og be(y)')
    plot([(180/pi)*X__g(2,:);(180/pi)*X__g(5,:);(180/pi)*(dw(2,:)-X__g(2,:)-X__g(5,:))]')
    plot([(180/pi)*S_g(2,:);-(180/pi)*S_g(2,:)]','b')
    plot([(180/pi)*S_g(5,:);-(180/pi)*S_g(5,:)]','g')
    legend('be(y)','ga(y)','v^g(y)','S_gbe(y)','','S_gga(y)')
    xlabel('k'), ylabel('deg/s')
    subplot(223)
    hold on,
    title('Gyrometer støy, v^g(z), ga(z) og be(z)')
    plot([(180/pi)*X__g(3,:);(180/pi)*X__g(6,:);(180/pi)*(dw(3,:)-X__g(3,:)-X__g(6,:))]')
    plot([(180/pi)*S_g(3,:);-(180/pi)*S_g(3,:)]','b')
    plot([(180/pi)*S_g(6,:);-(180/pi)*S_g(6,:)]','g')
    legend('be(z)','ga(z)','v^g(z)','S_gbe(z)','','S_gga(z)')
    xlabel('k'), ylabel('deg/s')
    subplot(224)
    hold on;
    title('Simulert vinkelhastighet')
    plot((180/pi)*dw(1,:),'red');
    plot((180/pi)*dw(2,:),'green');
    plot((180/pi)*dw(3,:),'blue');
    legend('x','y','z');
    xlabel('tid'), ylabel('deg/s')
end
```

Appendiks D: Errbud.m:

```
function [res] = errbud(F__st,G__st,H__st,K__st,Pe_0,Qb,Rd,dt)
% Feilbudsjett
% v.0.0.1 05.03.2013 (Start dato)
% Av Petter Lefoka
% I et feilbudsjett gjøres en kovaransanalyse for å se totalfeilen til de
% forksjellige tilstande. Totalfeilen til en tilstand bergnes ved og RMS
% summere alle standaravikene til gitt tilstand.
%
% dPp_a/dt=FPe+PF'+GQG'
% Pe_a(th_k)=(I-KH)Pp(I-KH)'+KRK'
%
% feilbudesjetresultat = errbud(F__st,G__st,H__st,K__st,Pe_0,Qb,Rd,dt)
%
% N=[I,zeros(6,size(H__st(:,:,1),2)-6)];
%
% Feilgrupper:
% 1) s(t0)__p
% 2) s(t0)__v
% 3) s(t0)__E
% 4) s(t0)__ga_a+Q_ga_a
% 5) s(t0)__ga_g+Q_ga_g
% 6) s(t0)__be_a+Q_be_a
% 7) s(t0)__be_g+Q_be_g
% 8) Q__a
% 9) Q__g
% 10)R__p
% 11)R__v
%
%
    r2g=180/pi;
    % count=[1 1 1 2 2 2 3 3 3 4 4 4 5 5 5 6 6 6 7 7 7 8 8 8 9 9 9];
     n=size(F__st,3);
     res=zeros(size(Pe_0,1),1,1);
     I_a=eye(size(K__st(:,:,1),1));
```

```
for i = 1:11

    % Minimaliserer bidrag, fordi og sette det til null lager
    % singulæriteter.
     temp=Qb*10^-9; %Minimaliserer bidrag
     Pe_a=dt*Pe_0*10^-9;
     Rb=0;
    te=zeros(size(Pe_0,1),1);
     if i==1
          Pe_a(1:3,:)=dt*Pe_0(1:3,:);
     elseif i==2
          Pe_a(4:6,:)=dt*Pe_0(4:6,:);
     elseif i==3
          Pe_a(7:9,:)=dt*Pe_0(7:9,:);
     elseif i==4
          temp(7:9,:)=Qb(7:9,:);
          Pe_a(10:12,:)=dt*Pe_0(10:12,:);
         % te(i:i+2)=diag(Pe_0(i:i+2,i:i+2));
     elseif i==5
          temp(10:12,:)=Qb(10:12,:);
          Pe_a(13:15,:)=dt*Pe_0(13:15,:);
     elseif i==6
          temp(13:15,:)=Qb(13:15,:);
          Pe_a(16:18,:)=dt*Pe_0(16:18,:);
     elseif i==7
          temp(16:18,:)=Qb(16:18,:);
          Pe_a(19:21,:)=dt*Pe_0(19:21,:);
     elseif i==8
          temp(1:3,:)=Qb(1:3,:);
     elseif i==9
          temp(4:6,:)=Qb(4:6,:);
     elseif i==10
          Rb=Rd;
     else
          Pe_a=dt*Pe_0;
          temp=Qb;
          Rb=Rd;
     end
```

```
    for k = 1:n
        H_a=H__st(:,:,k);
        G_a=G__st(:,:,k);
        F_a=F__st(:,:,k);
        K_a=K__st(:,:,k);
        [Fi_a,Ga_a]=k2dp(dt,F_a,G_a,temp); %Diskretisering
       Pp_a=Fi_a*Pe_a*Fi_a'+Ga_a*Ga_a'; %TO
       Pe_a=(I_a-K_a*H_a)*Pp_a*(I_a-K_a*H_a)'+K_a*Rb*K_a'; %MO

        %Lagrer Feil bidrag.
        res(:,k,i)=sqrt(diag(Pe_a));
        te=te+diag(Pe_a);
    end
     %lagrer
     fbr(:,i)=sqrt(te/k);
end
res(:,:,12)=sqrt(res(:,:,1).^2+res(:,:,2).^2+res(:,:,3).^2+(res(:,:,4).^2)...
+res(:,:,5).^2+res(:,:,6).^2+res(:,:,7).^2+res(:,:,8).^2+...
res(:,:,9).^2+res(:,:,10).^2);
fbr(:,12)=sqrt(fbr(:,1).^2+fbr(:,2).^2+fbr(:,3).^2+fbr(:,4).^2+...
fbr(:,5).^2+fbr(:,6).^2+fbr(:,7).^2+fbr(:,8).^2+fbr(:,9).^2+...
fbr(:,10).^2);
% plott
lt=1:size(K__st,3);
figure()
plot([res(1,lt,1)',res(1,lt,2)',res(1,lt,3)',res(1,lt,4)',res(1,lt,5)',...
 res(1,lt,6)',res(1,lt,7)',res(1,lt,8)',res(1,lt,9)'...
 ,res(1,lt,10)',res(1,lt,11)',res(1,lt,12)'])
title('Bidrag til posisjons feil (dp^n_x)'), xlabel('k'), ylabel('m')
legend('(1)s(t0)^p','(2)s(t0)^v','(3)s(t0)^E','(4)s(t0)^g^a_a+Q_g_a^a',...
 '(5)s(t0)^g^a_g+Q_g_a^g','(6)s(t0)^b^e_a+Q_b_e^a',...
 '(7)s(t0)^b^e_g+Q_b_e^g','(8)Q^a','(9)Q^g','(10)Rd','(11)Total','sum')
figure()
plot([res(4,lt,1)',res(4,lt,2)',res(4,lt,3)',res(4,lt,4)',res(4,lt,5)',...
 res(4,lt,6)',res(4,lt,7)',res(4,lt,8)',res(4,lt,9)',res(4,lt,10)'...
 ,res(4,lt,11)',res(4,lt,12)'])
title('Bidrag til hastighets feil (dv^n_x)'), xlabel('k'), ylabel('m/s')
```

```
    legend('(1)s(t0)^p','(2)s(t0)^v','(3)s(t0)^E','(4)s(t0)^g^a_a+Q_g_a^a',...
     '(5)s(t0)^g^a_g+Q_g_a^g','(6)s(t0)^b^e_a+Q_b_e^a',...
     '(7)s(t0)^b^e_g+Q_b_e^g','(8)Q^a','(9)Q^g','(10)Rd','(11)Total','sum')
    figure()
    plot(r2g*[res(7,lt,1)',res(7,lt,2)',res(7,lt,3)',res(7,lt,4)',res(7,lt,5)',...
     res(7,lt,6)',res(7,lt,7)',res(7,lt,8)',res(7,lt,9)',res(7,lt,10)'...
     ,res(7,lt,11)',res(7,lt,12)'])
    title('Bidrag til skvjestiling (E_x)'), xlabel('k'), ylabel('deg')
    legend('(1)s(t0)^p','(2)s(t0)^v','(3)s(t0)^E','(4)s(t0)^g^a_a+Q_g_a^a',...
     '(5)s(t0)^g^a_g+Q_g_a^g','(6)s(t0)^b^e_a+Q_b_e^a',...
     '(7)s(t0)^b^e_g+Q_b_e^g','(8)Q^a','(9)Q^g','(10)Rd','(11)Total','sum')
    %print error budget result
    fbr
end
```

Appendiks E: plot_banegen.m:

```
% plot_banegen
%figure()
%clf
%hold all,axis equal;
%title('plot av simulert omløp')
%axis([-0.04 0.40 -0.04 0.4])
%xlabel('x-akse')
%ylabel('y-label')
%plot(p__n(1,:),p__n(2,:))
% Simulering av akselerasjon
figure()
clf
subplot(231)
hold on;
title('simulert akselerasjon')
plot(time,a__n(1,1:size(time,2)),'red');
plot(time,a__n(2,1:size(time,2)),'green');
plot(time,a__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('tid')
ylabel('m/s^2')
subplot(232)
hold on;
title(' simulert hastighet')
plot(time,v__n(1,1:size(time,2)),'red');
plot(time,v__n(2,1:size(time,2)),'green');
plot(time,v__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('tid')
ylabel('m/s')
subplot(233)
hold on;
title(' simulert posisjon')
plot(time,p__n(1,1:size(time,2)),'red');
```

```
plot(time,p__n(2,1:size(time,2)),'green');

plot(time,p__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('tid')
ylabel('meter')
subplot(234)
hold on;
title('simulert vinkelhastighet')
plot(time,w_b_nb(1,1:size(time,2)),'red');
plot(time,w_b_nb(2,1:size(time,2)),'green');
plot(time,w_b_nb(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('tid')
ylabel('rad/s')
subplot(235)
hold on;
title(' simulert spesifikk kraft')
plot(time,f__b(1,1:size(time,2)),'red');
plot(time,f__b(2,1:size(time,2)),'green');
plot(time,f__b(3,1:size(time,2)),'blue');
legend('x','y');
legend('z')
legend('x','y','z');
xlabel('tid')
ylabel('m/s^2')
```

Appendiks F: plot_kalman.m:

```
%plot_kalman
figure();
subplot(221)
hold on;
plot(time,s(1,1:size(time,2)),'b',time,-s(1,1:size(time,2)),'r',time,p__n(1,1:size(time,2))-pb__n(1,1:size(time,2
% plot(time,s(1,1:size(time,2)),'b',time,-s(1,1:size(time,2)),'r',time,dxem(1,1:size(time,2)),'g');
legend('standardawik', '-standardawik','Estimert dx')
xlabel('sek')
ylabel('m')
subplot(222)
hold on;
plot(time,s(2,1:size(time,2)),'b',time,-s(2,1:size(time,2)),'r',time,(p__n(2,1:size(time,2))-pb__n(2,1:size(time,
% plot(time,s(2,1:size(time,2)),'b',time,-s(2,1:size(time,2)),'r',time,(dxem(2,1:size(time,2))),'g');
legend('standardawik','-standardawik','estimert dy')
xlabel('sek')
ylabel('m')
subplot(223)
hold on;
plot(time,s(3,1:size(time,2)),'b',time,-s(3,1:size(time,2)),'r',time,(p__n(3,1:size(time,2))-pb__n(3,1:size(time,
% plot(time,s(3,1:size(time,2)),'b',time,-s(3,1:size(time,2)),'r',time,(dxem(3,1:size(time,2))),'g');
legend('standardawik','-standardawik','estimert dz')
xlabel('sek')
ylabel('m')
figure();
subplot(221)
title('Differansen mellom sann og estimert hastighet om z retning')
hold on;
plot(time,s(4,1:size(time,2)),'b',time,-s(4,1:size(time,2)),'r',time,(v__n(1,1:size(time,2))-vb__n(1,1:size(time,
legend('standardawik','-standardawik','estimert dx')
xlabel('sek')
ylabel('m/s')
subplot(222)
title('Differansen mellom sann og estimert hastighet om z retning')
hold on;
```

Appendix F plot_kalman.m:

```
plot(time,s(5,1:size(time,2)),'b',time,-s(5,1:size(time,2)),'r',time,(v__n(2,1:size(time,2))-vb__n(2,1:size(time,2))

legend('standardawik','-standardawik','estimert dy')
xlabel('sek')
ylabel('m/s')
subplot(223)
title('Differansen mellom sann og estimert hastighet om z retning')
hold on;
plot(time,s(6,1:size(time,2)),'b',time,-s(6,1:size(time,2)),'r',time,(v__n(3,1:size(time,2))-vb__n(3,1:size(time,2))
legend('standardawik','-standardawik','estimert dz')
xlabel('sek')
ylabel('m/s')
figure();
subplot(221)
title('Forskjellen mellom estimert og sann skjevstilling om x retning')
hold on;
plot(time,(s(7,1:size(time,2))*r2g),'b',time,(-s(7,1:size(time,2))*r2g),'r',time,(eps(1,1:size(time,2))*r2g-
dxem(7,1:size(time,2))*r2g),'g');
%plot(0:0.01:15.98,(eps(1,1:size(time,2))*r2g),'k')
legend('standardawik','-standardawik','estimert dx','eps_x')
xlabel('sek')
ylabel('deg')
subplot(222)
title('Forskjellen mellom estimert og sann skjevstilling om y retning')
hold on;
plot(time,(s(8,1:size(time,2))*r2g),'b',time,(-s(8,1:size(time,2))*r2g),'r',time,(eps(2,1:size(time,2))*r2g-
dxem(8,1:size(time,2))*r2g),'g');
%plot(0:0.01:15.98,(eps(2,1:size(time,2))*r2g),'k')
legend('standardawik','-standardawik','estimert dy','eps_y')
xlabel('sek')
ylabel('deg')
subplot(223)
title('Forskjellen mellom estimert og sann skjevstilling om z retning')
hold on;
plot(time,(s(9,1:size(time,2))*r2g),'b',time,(-s(9,1:size(time,2))*r2g),'r',time,(eps(3,1:size(time,2))*r2g-
dxem(9,1:size(time,2))*r2g),'g');
%plot(time,(eps(3,1:size(time,2))*r2g),'k')
legend('standardawik','-standardawik','estimert dz','eps_z')
xlabel('sek')
```

```
ylabel('deg')
```

Appendiks G: plot_nav_euler.m:

```
%plot_nav_euler
figure()
title('NAV Euler')
subplot(221)
hold on;
title('posisjon NAV')
plot(time,pb__n(1,1:size(time,2)),'red');
plot(time,pb__n(2,1:size(time,2)),'green');
plot(time,pb__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('sek')
ylabel('meter')
subplot(222)
hold on;
title('hastighet NAV')
plot(time,vb__n(1,1:size(time,2)),'red');
plot(time,vb__n(2,1:size(time,2)),'green');
plot(time,vb__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('sek')
ylabel('m/s')
%-------plot av avvik mellom posisjon p__n og pb__n------
subplot(223)
hold on;
title('avvik mellom posisjon og posisjon NAV')
plot(time,dp__n(1,1:size(time,2)),'red');
plot(time,dp__n(2,1:size(time,2)),'green');
plot(time,dp__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('sek')
ylabel('meter')
%-------plot av avvik mellom hastighet v__n og vb__n------
subplot(224)
hold on;
```

```
title('avvik mellom hastighet og hastighet NAV')

plot(time,dv__n(1,1:size(time,2)),'red');
plot(time,dv__n(2,1:size(time,2)),'green');
plot(time,dv__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('sek')
ylabel('m/s')
```

Appendiks H: plot_nav_heun.m:

```
% plot_nav_heun
figure()
title('NAV Heun')
clf
subplot(221)
hold on;
title('posisjon NAV')
plot(time,pb__n(1,1:size(time,2)),'red');
plot(time,pb__n(2,1:size(time,2)),'green');
plot(time,pb__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('sek')
ylabel('meter')
subplot(222)
hold on;
title('hastighet NAV')
plot(time,vb__n(1,1:size(time,2)),'red');
plot(time,vb__n(2,1:size(time,2)),'green');
plot(time,vb__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('sek')
ylabel('m/s')
%-------plot av avvik mellom posisjon p__n og p_b_n-----
subplot(223)
hold on;
title('avvik mellom posisjon(sann) og posisjon NAV')
plot(time,dp1__n(1,1:size(time,2)),'red');
plot(time,dp1__n(2,1:size(time,2)),'green');
plot(time,dp1__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('sek')
ylabel('meter')
%-------plot av avvik mellom hastighet v__n og vb__n-----
subplot(224)
```

Appendix H plot_nav_heun.m:

```
hold on;

title('avvik mellom hastighet(sann) og hastighet NAV')
plot(time,dv1__n(1,1:size(time,2)),'red');
plot(time,dv1__n(2,1:size(time,2)),'green');
plot(time,dv1__n(3,1:size(time,2)),'blue');
legend('x','y','z');
xlabel('sek')
ylabel('m/s')
```