

**UNIVERSITETET I OSLO**  
**Fysisk Institutt**

**3-D Banegenerator**

**Masteroppgave**

Peshko Majidian

**15. desember 2014**





## Forord

Denne oppgaven er formulert og utgitt av Universitetscenteret på Kjeller (UNIK) høsten 2014 og er avslutningen på mitt masterstudium i Elektronikk og Datateknologi, studieretning Kybernetikk ved Universitetet i Oslo. Jeg vil gjerne takke min veileder Oddvar Hallingstad for veiledning og innspill som har gjort oppgaven veldig lærerik. Jeg vil også takke familie og venner som har vært støttende gjennom perioden.

Peshko Majidian

Kjeller 15. desember-14



## Sammendrag

Denne oppgaven har gått ut på å lage en banegenerator som simulerer banen til et fartøy i 3-D. Den spesifikke kraften og vinkelhastigheten har blitt regnet ut for å simulere akselerometer og gyroskop og ved å implementere et treghetsnavigasjonssystem (TNS), har det vært mulig å beregne banen til fartøyet ved å bruke sensordataene i navigasjonslikningene.

En sirkelbane ble brukt til å verifisere TNS systemet før 3-D banen ble testet uten støy. Deretter skulle det legges til hvit og farget støy på sensorene for å få realistiske målinger og se hvordan TNS systemet blir påvirket av støybelagte målinger.

Resultatet viser integrasjonsrutinen til Heuns bringer TNS systemet nærmere flyets sanne tilstand enn Eulers metode i det støyfrie tilfellet. Men straks det blir lagt på støy vil systemet ikke klare å følge riktig bane uten å ta inn målinger av posisjon vha et kalmanfilter.



# Innholdsfortegnelse

|   |     |
|---|-----|
| Forord .....  | III |
| Sammendrag .....                                      | V   |
| 1 Innledning .....                                    | 1   |
| 1.1 Generelt .....                                    | 1   |
| 1.2 Målsetting for oppgaven .....                     | 1   |
| 1.3 Struktur .....                                    | 1   |
| 2 <i>Matematisk grunnlag</i> .....                    | 2   |
| 2.1 Affint rom.....                                   | 2   |
| 2.2 Koordinatsystemer og rammer .....                 | 3   |
| 2.2.1 <i>Referansesystem og treghetsrom</i> .....     | 3   |
| 2.2.2 <i>Legemeramme</i> .....                        | 4   |
| 2.2.3 <i>Tangentramme</i> .....                       | 4   |
| 2.3 Retningskosinmatrise (RKM) .....                  | 4   |
| 2.4 Eulervinkelrepresentasjon av RKM.....             | 5   |
| 2.4.1 <i>3-2-1 Eulervinkler</i> .....                 | 5   |
| 2.5 Derivasjon av RKM.....                            | 6   |
| 2.6 Diskretisering av et kontinuerlig system .....    | 7   |
| 3 <i>Banegenerator</i> .....                          | 9   |
| 3.1 Baneelementene .....                              | 9   |
| 3.1.1 <i>Rett bane med og uten akselerasjon</i> ..... | 10  |
| 3.1.2 <i>Eulerspiral</i> .....                        | 11  |
| 3.1.3 <i>Sirkelbane</i> .....                         | 13  |
| 3.1.4 <i>Omvendt Eulerspiral</i> .....                | 14  |
| 3.2 Transformasjon mellom koordinatsystemer .....     | 16  |
| 3.3 Spesifikk kraft .....                             | 17  |
| 3.6 Vinkelfart .....                                  | 17  |
| 3.7 Banegeneratoren i Matlab .....                    | 18  |
| 3.8 Rett bane i Matlab .....                          | 20  |
| 3.8.1 <i>Eulerspiral i Matlab</i> .....               | 21  |
| 3.8.2 <i>Sirkelbane i Matlab</i> .....                | 22  |
| 3.8.3 <i>Omvendt Eulerspiral i Matlab</i> .....       | 23  |
| 4 <i>TNS for flat ikke-rotterende jord</i> .....      | 25  |
| 4.1 Integrasjonsrutiner .....                         | 25  |
| 4.1.1 <i>Eulers metode</i> .....                      | 25  |

|       |   |    |
|-------|---|----|
| 4.1.2 | <i>Heuns Metode</i> .....                             | 26 |
| 4.2   | TNS i Matlab.....                                     | 27 |
| 4.3   | Støy og støymodellering.....                          | 29 |
| 4.3.1 | <i>Hvit støy</i> .....                                | 29 |
| 4.3.2 | <i>Farget støy</i> .....                              | 29 |
| 4.4   | Modellering av akselerometer .....                    | 30 |
| 4.5   | Modellering av gyro.....                              | 34 |
| 4.6   | Kalmanfilter .....                                    | 38 |
| 4.6.1 | <i>Linearisert Kalmanfilter</i> .....                 | 38 |
| 5     | <i>Simuleringsresultater</i> .....                    | 42 |
| 5.1   | Banegenerator .....                                   | 42 |
| 5.2   | Simulering av 3-D banen uten støy .....               | 44 |
| 5.2.1 | <i>Resultater Eulers metode uten støy</i> .....       | 44 |
| 5.2.2 | <i>Resultater Heuns metode uten støy</i> .....        | 45 |
| 5.3   | Simulering av 3-D banen med hvit støy.....            | 46 |
| 5.3.1 | <i>Resultater Eulers metode med hvit støy</i> .....   | 46 |
| 5.3.2 | <i>Resultater Heuns metode med hvit støy</i> .....    | 47 |
| 5.4   | Simulering av 3-D bane med farget støy .....          | 48 |
| 5.4.1 | <i>Resultater Eulers metode med Farget støy</i> ..... | 48 |
| 5.4.2 | <i>Resultater Heuns metode med Farget støy</i> .....  | 49 |
| 5.5   | Resultater TNS med hvit og farget støy.....           | 50 |
| 5.5.1 | <i>Eulers metode med hvit og farget støy</i> .....    | 50 |
| 5.5.2 | <i>Heuns metode med hvit og farget støy</i> .....     | 51 |
| 6     | <i>Diskusjon</i> .....                                | 52 |
| 7     | <i>Konklusjon</i> .....                               | 53 |
| 8     | <i>Videre arbeid</i> .....                            | 54 |
|       | <i>Referanser</i> .....                               | 55 |
|       | <i>Vedlegg</i> .....                                  | 56 |
|       | <i>Appendiks A: Pseudokoder</i> .....                 | 56 |
|       | Appendiks A-1: Rett linje med akselerasjon.....       | 56 |
|       | Appendiks A-2: Eulerspiral.....                       | 56 |
|       | Appendiks A-3: Sirkel.....                            | 57 |
|       | Appendiks A-4: Omvendt Eulerspiral.....               | 57 |
|       | Appendiks A-5: TNS (Heuns) .....                      | 58 |
|       | Appendiks A-6: Akselerometer.....                     | 58 |



|  |    |
|--|----|
| Appendiks A-7: Gyro .....                                  | 59 |
| Appendiks A-8: Linearisert Kalmanfilter .....              | 60 |
| <i>Appendiks B: Matlabkoder</i> .....                      | 61 |
| Appendiks B-1: Main.....                                   | 61 |
| Appendiks B-2: Banegenerator.....                          | 62 |
| Appendiks B-3: Flyets stilling .....                       | 65 |
| Appendiks B-4: Vinkelhastighet.....                        | 66 |
| Appendiks B-5: 3-2-1 Eulervinkler .....                    | 66 |
| Appendiks B-6: Navigasjonslikningene .....                 | 67 |
| Appendiks B-7: Skjevsymmetriske matrisen .....             | 67 |
| Appendiks B-8: Støyfunksjonen.....                         | 68 |
| Appendiks B-9: Diskretisering av kontinuerlig system ..... | 69 |
| Appendiks B-10: Eulervinklene invers Euler .....           | 69 |
| Appendiks B-11: Figurer .....                              | 70 |

## Figurliste

|   |    |
|---|----|
| Figur 1: 3-2-1 Eulervinkler rotasjonssekvens .....  | 5  |
| Figur 2: Fire baneelementer utgjør hele banen. Kulene viser start og slutt for hver baneelement ..... | 10 |
| Figur 3: Illustrert Eulerspiral for forskjellige vinkler .....  | 13 |
| Figur 4: Sammenhengen mellom flyets bevegelse i planet og rommet .....                                | 16 |
| Figur 5: RBA funksjonen returnerer fartøyets hastighet, akselerasjon og posisjon .....                | 21 |
| Figur 6: ESP funksjonen returnerer fartøyets hastighet, akselerasjon og posisjon .....                | 22 |
| Figur 7: SBA funksjonen returnerer fartøyets hastighet, akselerasjon og posisjon.....                 | 23 |
| Figur 8: OES funksjonen returnerer fartøyets hastighet, akselerasjon og posisjon .....                | 23 |
| Figur 9: Utgang systemmodellen og navigasjonslikningene .....   | 27 |
| Figur 10: Akkumulert feil i Heuns integrasjonsmetode etter 3600 sekunder .....                        | 28 |
| Figur 11: Velocity Random Walk (VRW) i akselerometeret .....  | 33 |
| Figur 12: 1.ordens Gauss Markov støyprosess i akselerometer .....                                     | 33 |
| Figur 13: Angular Random Walk i gyroen.....   | 37 |
| Figur 14: 1.ordens Gauss Markov støyprosess i gyro.....   | 37 |
| Figur 15: Blokkskjema for linearisert kalmanfilter (LKF). LKF har ingen tilbakekobling.....           | 38 |
| Figur 16: Utvidet TNS blokkskjema med Kalmanfilter .....  | 40 |
| Figur 17: Akselerasjonen og hastighet fra banegenerator.....  | 42 |
| Figur 18: 3-D banen og posisjonen til flyet. Rød sirkel angir startposisjon.....                      | 42 |
| Figur 19: Eulervinklene beregnet av banegeneratoren .....   | 43 |
| Figur 20: Spesifikk kraft og vinkelhastighet fra banegeneratoren.....                                 | 43 |
| Figur 21: Differanse Banegenerator og TNS, Eulers metode .....  | 44 |
| Figur 22: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS.....                      | 44 |
| Figur 23: Differanse Banegenerator og TNS, Heuns metode.....  | 45 |
| Figur 24: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS.....                      | 45 |
| Figur 25: Differanse Banegenerator og TNS, Eulers metode med hvit støy .....                          | 46 |
| Figur 26: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS.....                      | 46 |
| Figur 27: Differanse Banegenerator og TNS, Heuns metode.....  | 47 |
| Figur 28: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS.....                      | 47 |
| Figur 29: Differanse Banegenerator og TNS, Eulers metode .....  | 48 |
| Figur 30: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS.....                      | 48 |
| Figur 31: Differanse Banegenerator og TNS, Heuns metode .....   | 49 |
| Figur 32: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS.....                      | 49 |
| Figur 33: Differanse Banegenerator og TNS, Eulers metode .....  | 50 |
| Figur 34: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS.....                      | 50 |
| Figur 35: Differanse Banegenerator og TNS, Heuns metode .....   | 51 |
| Figur 36: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS.....                      | 51 |

## Tabelliste

|   |    |
|---|----|
| Tabell 1: Notasjon for vektorrom og affint rom.....   | 2  |
| Tabell 2: Symbolbeskrivelse for kalmanfilteret.....   | 39 |
| Tabell 3: Feildefinisjoner brukt i kalmanfilteret[12] .....                                 | 39 |
| Tabell 4: Sammenlikning av feilen til integrasjonsrutinene med forskjellige frekvenser..... | 52 |

# 1 Innledning

## 1.1 Generelt

Denne oppgaven bygger på en tidligere masteroppgave som har analysert en banegenerator for et fartøy i planet. Målet var å finne en algoritme som gir ut sanne verdier av fartøyets akselerasjon, hastighet og posisjon. Det ble laget en bane som var basert på rettlinjet og sirkelbevegelse, med Eulerspiraler som ga overgangen mellom rettlinjet og sirkelbevegelse.

## 1.2 Målsetting for oppgaven

Med denne banen som utgangspunkt er målsettingen for oppgaven følgende:

1. Banegeneratoren utvides til å kunne svinge til både venstre og høyre og tiden i en sving skal bestemmes av ønsket sving i radianer.
2. Deretter skal banen generaliseres til 3-D.
3. Den spesifikke kraften  $\underline{f}^b$  og vinkelhastigheten  $\underline{\omega}_b^{nb}$  for 3-D banen skal beregnes uten og med støy.
4. Det skal implementeres et treghtetsnavigasjonssystem for flat ikke-roterende jord og denne skal testes uten og med støy.

Oppgaven vil hovedsakelig basere seg på fagene UNIK 4540-Matematisk modellering av dynamiske systemer og UNIK 4500 Stokastiske systemer, som begge blir forelest på UNIK.

## 1.3 Struktur

**Kapittel 2:** Mye av den matematiske bakgrunnen som ligger til grunn for denne oppgaven er presentert i dette kapitlet.

**Kapittel 3:** Her blir banegeneratoren presentert i detalj med utledning av baneelementene. Det blir også gitt en beskrivelse av Matlab programmet som genererer banen.

**Kapittel 4:** I dette kapitlet introduseres TNS systemet og de numeriske integrasjonsmetodene brukt i oppgaven. I tillegg kommer støy og modellering av akselerometer og gyro med begrensede støykilder. Til slutt presenteres kalmanfilteret.

**Kapittel 5:** Her blir resultatene lagt fram i form av figurer som viser differansen mellom TNS systemets beregninger og de sanne verdiene

**Kapittel 6:** Diskusjon

**Kapittel 7:** Konklusjon

**Kapittel 8:** Videre arbeid

## 2 Matematisk grunnlag

### 2.1 Affint rom

La  $\mathcal{A}$  være en ikke-tom mengde av punkter, og la  $\mathcal{V}$  være et vektorrom. Anta at for et vilkårlig punkt  $P \in \mathcal{A}$  og  $\vec{a} \in \mathcal{V}$  er det definert en addisjon  $P + \vec{a} \in \mathcal{A}$  som tilfredsstiller følgende betingelser[1]:

1.  $P + \vec{0} = P$ , ( $\vec{0}$  er nullvektoren i  $\mathcal{V}$ )
2.  $(P + \vec{a}) + \vec{b} = P + (\vec{a} + \vec{b})$  for alle  $\forall \vec{a}, \vec{b} \in \mathcal{V}$
3. For enhver  $Q \in \mathcal{A}$  eksisterer en entydig vektor  $\vec{a} \in \mathcal{V}$  slik at  $Q = P + \vec{a}$

Et affint har et innebygd vektorrom som er en ikke-tom mengde av elementer kalt vektorer. Vektoren må da oppfylle et sett med regneregler[2]. En vektor er en fysisk størrelse som både har en retning og en verdi. Vektoren kan beskrives som en geometrisk eller algebraisk vektor. Geometriske vektorer kan ses på som piler i rommet med en lengde og retning, mens algebraiske vektorer kan ses på som n-tupler av tall samlet i en kolonnematrise og som er representert i et basissystem.

Notasjonen for de sentrale begrepene i et vektorrom og affint rom er gitt i tabellen[1]

Tabell 1: Notasjon for vektorrom og affint rom

| Rom                        | Rammer  | Andre navn på rammer                        |
|----------------------------|---|---|
| $\mathcal{V}$ : vektorrom  | $\mathcal{F}_a^{\mathcal{V}} = \{\vec{a}_1, \vec{a}_2, \vec{a}_3\}$ :<br>Ramme a i vektorrom $\mathcal{V}$<br>med basisvektorer $\vec{a}_i$                             | Basis, basissystem, basisvektorsett<br>k.s. |
| $\mathcal{A}$ : affint rom | $\mathcal{F}_a^{\mathcal{A}} = \{O_a, \vec{a}_1, \vec{a}_2, \vec{a}_3\}$<br>Ramme a i det affine rom<br>$\mathcal{A}$ med origo i $O_a$ og<br>basisvektorer $\vec{a}_i$ | k.s.  |

Lengden av en vektor er gitt ved den euklidiske normen

$$\|\vec{v}\| = \sqrt{v_1^2 + v_2^2 + v_3^2} \quad (2.1)$$

og enhetsvektoren er en vektor med enhetslengde

$$\vec{e} = \frac{\vec{v}}{\|\vec{v}\|} \quad (2.2)$$

Kryssproduktet av to vektorer  $\vec{a} \times \vec{b}$  gir en tredje vektor  $\vec{c}$  som står vinkelrett på både  $\vec{a}$  og  $\vec{b}$  og har en lengde lik

$$\|\vec{c}\| = \|\vec{a}\| \|\vec{b}\| \sin \angle \vec{a} \vec{b} \quad (2.3)$$

## 2.2 Koordinatsystemer og rammer

Et affint rom består av punkter og vektorer. Det er ønskelig med en funksjon som tilordner entydige tallverdier til hvert punkt i det affine rommet. En slik funksjon kalles et koordinatsystem[1]. Affine koordinater i et  $n$ -dimensjonalt affint rom representeres slik

$$\mathcal{F}^q = \{0_q; \vec{q}_1, \vec{q}_2, \vec{q}_3, \dots, \vec{q}_n\}$$

Der  $0_q$  er origo og vektorene  $\{\vec{q}_i\}$ , der  $i = 1, 2, 3, \dots, n$  er et sett basisvektorer for det tilhørende vektorrom. Dersom et punkt  $P$  kan representeres i to rammer med ortonormale basisvektorsett i et affint rom vil det punktet kunne assosieres med en posisjonsvektor i hver ramme slik at [1]

$$\begin{aligned}\mathcal{F}^q &= \{0_q; \vec{q}_1, \vec{q}_2, \vec{q}_3, \dots, \vec{q}_n\} \\ \mathcal{F}^p &= \{0_p; \vec{p}_1, \vec{p}_2, \vec{p}_3, \dots, \vec{p}_n\}\end{aligned}$$

Punktet  $P$  kan assosieres med en posisjonsvektor i hver ramme

$$P = O_q + \vec{r} = O_p + \vec{\rho} \quad (2.4)$$

der  $\vec{r}_q$  og  $\vec{r}_p$  er posisjonsvektorer fra origo i  $q$  og  $p$ -rammen til punktet  $P$ . Dette gir

$$\vec{r}_{qp} = O_p - O_q \quad (2.5)$$

som resulterer i

$$\vec{r} = \vec{r}_{qp} + \vec{\rho} \quad (2.6)$$

For geometriske vektorer er sammenhengen

$$\underline{r}^q = \underline{r}_{qp}^q + R_p^q \underline{\rho}^p \quad (2.7)$$

Der matrisen  $R_p^q$  er en retningskosinmatrise som blir presentert senere.

Bevegelsen til et fartøy vil refereres til ulike rammer. I denne oppgaven for en flat ikke-roterende jord er det nok å definere treghetsrammen, tangentrammen og legemerammen

### 2.2.1 Referansesystem og treghetsrom

Et referansesystem er et koordinatsystem som er fast i forhold til et referanselegeme. I fysikken beskriver man en bevegelse relativt til et referansesystem.

Newtons første lov (treghetsloven) tjener til å avgjøre i hvilke referansesystemer alle Newtons lover gjelder. Dersom man har et referansesystem der treghetsloven gjelder, kalles referansesystemet et treghetssystem. I et treghetssystem vil Newtons andre og tredje lov også gjelde [3]. Treghetsrammen har origo i jordas sentrum og basisvektorene refererer til fiksstjernene i verdensrommet og roterer ikke om jorda.

Rammen beskrives slik

$$\mathcal{F}^i = \{O_i; \vec{b}_i, \vec{b}_i, \vec{b}_i\}$$

### 2.2.2 Legemeramme

Legemerammen er et fartøyfast koordinatsystem som forflytter seg med fartøyet og har origo  $O_b$  i massesenteret. Basisvektorene til legemerammen sammenfaller med fartøyets roll, pitch og yaw akser.

$$\mathcal{F}^b = \{O_b; \vec{b}_1, \vec{b}_2, \vec{b}_3\}$$

### 2.2.3 Tangentramme

Tangentrammen består av et koordinatsystem med basisvektorene  $\vec{t}_1, \vec{t}_2$  og  $\vec{t}_3$  som tangerer jordoverflaten. Basisvektorene faller sammen med retning nord, øst og opp. Det er ønskelig å navigere relativt til denne rammen. Origo sammenfaller med et fartøy der den har sin posisjon ved tiden  $t_0$ . Rammen beskrives slik

$$\mathcal{F}^t = \{O_t; \vec{b}_t, \vec{b}_t, \vec{b}_t\}$$

Dersom man antar at gyro og akselerometersensorene som er brukt er av en nøyaktighetsklasse som gjør at vi kan betrakte tangentplanet som et treghetsrom[5], kan vi definere et geografisk koordinatsystem med origo i flyet og aksene samme faller med nord, øst og opp.

$$\mathcal{F}^n = \{O_n; \vec{b}_n, \vec{b}_n, \vec{b}_n\}$$

som nå er den rammen fartøyets navigasjon ønskes referert til.

## 2.3 Retningskosinmatrise (RKM)

For å kunne representere punkter og vektorer i forskjellige basissystemer brukes RKM[1]. For eksempel vil vektoren  $\underline{r}^p$ , en geometrisk vektor representert i p-systemet, og  $\underline{r}^q$ , samme geometrisk vektor representert i q-systemet ha sammenhengen

$$\underline{r}^q = C_p^q \underline{r}^p \text{ hvor } C_p^q = [\langle \vec{p}_j, \vec{q}_i^* \rangle] \quad (2.8)$$

$C_p^q$  er en retningskosinmatrise som her er brukt som en koordinattransformasjonsmatrise (KTM), men RKM kan brukes i mange sammenhenger og får navn deretter. RKM mellom to rammer som har ortonormale basissystemer betegnes med  $R_p^q$ . Denne matrisen er ortogonal slik at

$$(R_p^q)^{-1} = (R_p^q)^T \quad (2.9)$$

$$R_p^q (R_p^q)^T = I \quad (2.10)$$

og kolonnevektorene er ortonormale (her kan  $R_p^q$  også tolkes som en stillingsmatrise)

$$R_p^q = [\underline{p}_1^q, \underline{p}_2^q, \underline{p}_3^q] \quad (2.11)$$

KTM kan samlet rotere et helt basisvektorsett fra en rammen  $\mathcal{F}^p$  til  $\mathcal{F}^q$ . I denne oppgaven blir det mellom de ortonormale rammene navigasjonsrammen  $\mathcal{F}^n$ , legemerammen,  $\mathcal{F}^b$  og hjelperammen  $\mathcal{F}^a$  som blir beskrevet under banegeneratoren.

## 2.4 Eulervinkelrepresentasjon av RKM

Stillingen til et legeme blir ofte representert av eulervinkler. For å finne sammenhengen mellom to rammer, eller stillinger, kan man tenke seg at de opprinnelig var sammenfallende. For eksempel, dersom basisvektorsettene til p og q-systemet opprinnelig var sammenfallende, fåes den endelige stillingen til p-systemet ved å dreie den en vinkel  $\theta$  om  $q_i$  akse. Dette gjøres ved hjelp av de elementære RKM:

$$R_1(\theta_1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\theta_1 & -s\theta_1 \\ 0 & s\theta_1 & c\theta_1 \end{bmatrix}, R_2(\theta_2) = \begin{bmatrix} c\theta_2 & 0 & s\theta_2 \\ 0 & 1 & 0 \\ -s\theta_2 & 0 & c\theta_2 \end{bmatrix}, R_3(\theta_3) = \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 \\ s\theta_3 & c\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Når RKM representeres slik ved hjelp av basisvektorene, kan det gjøres på to måter[1]:

1. Rotasjon om faste koordinataksler

$$R_p^q(\underline{\theta}) = R_1(\theta_1)R_2(\theta_2)R_3(\theta_3) \quad (2.12)$$

2. Rotasjon om nye koordinataksler (Eulervinkler).

$$R_p^q(\underline{\theta}) = R_3(\theta_3)R_2(\theta_2)R_1(\theta_1) \quad (2.13)$$

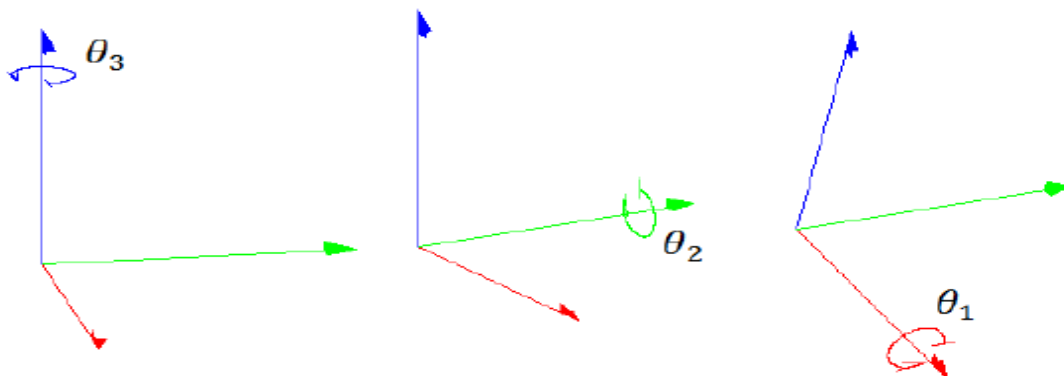
I denne oppgaven benyttes kun Eulervinkler.

### 2.4.1 3-2-1 Eulervinkler

Ved simulering av fly og båter brukes ofte følgende stillingsmatrise

$$R_b^n(\underline{\theta}) = R_3(\theta_3)R_2(\theta_2)R_1(\theta_1) \quad (2.14)$$

Notasjonen 3-2-1 refererer til dreiesekvens, der det først dreies  $\theta_3$  om akse 3, så  $\theta_2$  om akse 2 og til slutt  $\theta_1$  om akse 1. Det finnes flere dreiesekvenser, men i denne oppgaven brukes kun 3-2-1 Eulervinkler.



Figur 1: 3-2-1 Eulervinkler rotasjonssekvens

Koordinattransformasjonsmatrisen kan finnes ved å multiplisere sammen de elementære RKM slik at

$$R_p^q(\underline{\theta}) = R_3(\theta_3)R_2(\theta_2)R_1(\theta_1) = \begin{bmatrix} c\theta_3c\theta_2 & c\theta_3s\theta_2s\theta_1 - s\theta_3c\theta_1 & c\theta_3s\theta_2c\theta_1 + s\theta_3s\theta_1 \\ s\theta_3c\theta_2 & s\theta_3s\theta_2s\theta_1 + c\theta_3c\theta_1 & s\theta_3s\theta_2c\theta_1 - c\theta_3s\theta_1 \\ -s\theta_2 & c\theta_2s\theta_1 & c\theta_2c\theta_1 \end{bmatrix}$$

$$= \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (2.15)$$

Dette er løsningen på det direkte problem, der  $R_b^n$  skal bestemmes gitt vinkler og dreiesekvens.

Det inverse problem er når  $R_b^n$  er gitt og vinklene for en bestemt dreiesekvens skal bestemmes.

Gitt  $R_b^n(\underline{\theta})$  med tallverdier for elementene, vinklene finnes vha atan2 som gir entydige vinkler har løsningen [4]

$$\theta_1 = \text{atan2}\left(\frac{r_{32}}{c\theta_2}, \frac{r_{33}}{c\theta_2}\right) \quad (2.16)$$

$$\theta_2 = \text{atan2}\left(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}\right) \quad (2.17)$$

$$\theta_3 = \text{atan2}\left(\frac{r_{21}}{c\theta_2}, \frac{r_{11}}{c\theta_2}\right) \quad (2.18)$$

Først må  $\theta_2$  løses. For  $\theta_2 = \mp 90^\circ$  kan bare summen av  $\theta_1$  og  $\theta_2$  beregnes[1].

## 2.5 Derivasjon av RKM

Fartøyet i oppgaven er i bevegelse og stillingen endrer seg slik at RKM er tidsvariant. Det er da nødvendig å finne den deriverte av RKM og vinkelhastigheten mellom hver stilling. RKM er en ortogonal matrise som gir[1]

$$RR^T = I \quad (2.19)$$

Der forenklet notasjon  $R = R_b^n$  er brukt, derivasjon av denne gir

$$\dot{R}R^T + R\dot{R}^T = \dot{R}R^T + (\dot{R}R^T)^T = 0 \quad (2.20)$$

Definerer så

$$S = \dot{R}R^T \quad (2.21)$$

S må da være en skjevsymmetrisk matrise fordi

$$S + S^T = 0 \quad (2.22)$$

$$S = \dot{R}R^T = \dot{R}R^{-1} \Leftrightarrow \dot{R} = SR \quad (2.23)$$



S er skjevsymmetrisk og må ha strukturen

$$\begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \quad (2.24)$$

som gir

$$\underline{\omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \underline{\omega}^{nb} = \begin{bmatrix} \omega_x^{nb} \\ \omega_y^{nb} \\ \omega_z^{nb} \end{bmatrix} \quad (2.25)$$

Som er vinkelhastigheten til  $\mathcal{F}^b$  sett fra  $\mathcal{F}^n$  og representert i  $\mathcal{F}^b$ .  
 $\dot{R}$  er regnet ut numerisk.

## 2.6 Diskretisering av et kontinuerlig system

Gitt det kontinuerlige systemet

$$\dot{\underline{x}}(t) = F\underline{x}(t) + L\underline{u}(t) + G\underline{v}(t) \quad (2.26)$$

$$\underline{x}_0 \sim \mathcal{N}(\underline{\bar{x}}_0, \bar{P}_0), \quad \underline{v} \sim \mathcal{N}(\underline{0}, \tilde{Q}\delta(t-\tau)), \quad E\{\underline{x}_0 * \underline{v}^T(t)\} = \underline{0}$$

Systemet ønskes over på formen[10]

$$\underline{x}_{k+1} = \phi \underline{x}_k + \Lambda \underline{u}_k + \Gamma \underline{v}_k \quad (2.27)$$

$$\underline{x}_0 \sim \mathcal{N}(\underline{\bar{x}}_0, \bar{P}_0), \quad \underline{v} \sim \mathcal{N}(\underline{0}, Q), \quad E\{\underline{x}_0 * \underline{v}_k^T\} = \underline{0}$$

slik at

$$\underline{\bar{x}}(t_k) = \underline{\bar{x}}, \quad \bar{P}(t_k) = \bar{P}_k, \quad t_k = k * \Delta t$$

Diskretiseringen gjøres slik, først for systemmatrisen

$$\phi = e^{F*\Delta t} = e^{F(t_{k+1}-t_k)} \quad (2.28)$$

Pådragsmatrisen finnes slik

$$\Lambda \underline{u}_k = \int_{t_k}^{t_{k+1}} e^{F(t_{k+1}-\tau)} L d\tau * \underline{u}_k \quad (2.29)$$

dersom  $\underline{u}(t) = \underline{u}_k, t \in [t_k, t_{k+1}]$ , kan  $\Lambda$  finnes fra

$$\tilde{F} = \begin{bmatrix} F & L \\ 0 & 0 \end{bmatrix} \quad (2.30)$$

$$e^{\tilde{F}(t_{k+1}-t_k)} = \begin{bmatrix} \tilde{\Phi}_{11} & \tilde{\Phi}_{12} \\ 0 & I \end{bmatrix} \quad (2.31)$$

hvor  $\Lambda = \tilde{\phi}_{12}$ ,  $\phi = \tilde{\phi}_{11}$

Og den diskretiserte støymatrisen kan finnes slik

$$\Gamma Q \Gamma^T = \int_{t_k}^{t_{k+1}} \phi(t, \tau) G \tilde{Q} G^T \phi(t, \tau) d\tau \quad (2.32)$$

$$\tilde{F} = \begin{bmatrix} F & G \tilde{Q} G^T \\ 0 & -F^T \end{bmatrix} \quad (2.33)$$

$$\tilde{\phi}(t_{k+1}, t_k) = e^{\tilde{F}(t_{k+1}-t_k)} = \begin{bmatrix} \tilde{\phi}_{11} & \tilde{\phi}_{12} \\ 0 & \tilde{\phi}_{22} \end{bmatrix} \quad (2.34)$$

$$\Gamma Q \Gamma^T = \tilde{\phi}_{12} \tilde{\phi}_{22}^{-1}, \quad \phi = \tilde{\phi}_{11} \quad (2.35)$$

$\Gamma$  kan spaltes ut av  $\Gamma Q \Gamma^T$  ved hjelp av UD faktorisering

$$\Gamma Q \Gamma^T = \left( U D^{\frac{1}{2}} \right) I \left( U D^{\frac{1}{2}} \right)^T \quad (2.36)$$

$$\Gamma = U D^{\frac{1}{2}} \quad (2.37)$$

som gir  $Q = I$  slik at støyen som trekkes av random tallgeneratoren i Matlab er standardnormalfordelte tall

$$\underline{v} \sim \mathcal{N}(\underline{0}, I)$$

### 3 Banegenerator

Banegeneratoren skal simulere banen til et fartøy i 3-D, men siden det har blitt simulert med hastigheter opp til 100 m/s vil det være mer spesifikt å betegne fartøyet som et fly videre i rapporten.

For å generere realistiske målinger til TNS systemet er det ønskelig å bruke en banegenerator. Banegeneratoren er en matematisk modell for et fysisk system som gir ut deterministiske målinger for akselerasjon, fart, posisjon samt flyets orientering og vinkelhastighet[14]. Bevegelsen til et punkt i rommet kan beskrives ved hjelp av tre ortonormale rammer:

$\mathcal{F}^n = \{O_n; \vec{n}_1, \vec{n}_2, \vec{n}_3\}$ , den geografiske rammen som banegeneratoren og TNS systemet skal representeres i.

$\mathcal{F}^a = \{O_a; \vec{a}_1, \vec{a}_2, \vec{a}_3\}$ , er den lokale rammen for baneelementene. Hvert baneelement defineres i en egen  $\mathcal{F}^a$  ramme som blir nummerert  $\mathcal{F}^1$  til  $\mathcal{F}^k$  etter antall baneelementer k som banen består av. Bevegelsen til alle baneelementene starter i  $O_a$  i sine respektive rammer.

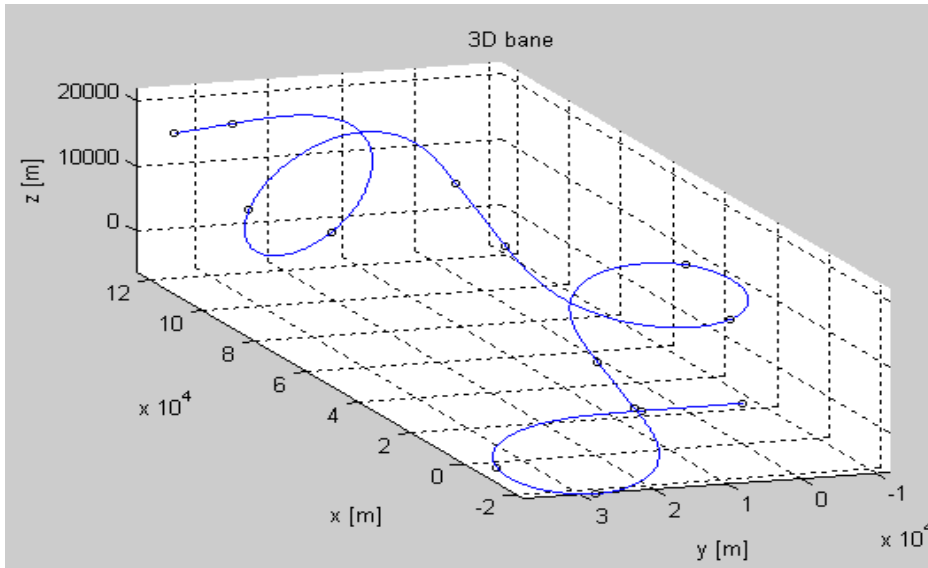
$\mathcal{F}^b = \{O_b; \vec{b}_1, \vec{b}_2, \vec{b}_3\}$ , er legemerammen som er fast i flyet med  $\vec{b}_1$  langs hastighetsvektoren  $\underline{v}_b^n$ ,  $\vec{b}_2$  langs venstre vinge og  $\vec{b}_3$  peker oppover. I hver  $\mathcal{F}^a$  ramme blir akselerasjonen  $\underline{a}_b^a$ , hastigheten  $\underline{v}_b^a$  og posisjonen  $\underline{p}_b^a$  beregnet i planet.

Disse koordinatsystemene er skjøtet sammen i  $\mathcal{F}^n$  slik at banen, og dermed posisjon, hastighet, akselerasjon og tiden, blir kontinuerlig i rommet. Utdata fra banegeneratoren representert i  $\mathcal{F}^n$  er akselerasjon  $\underline{a}_b^n$ , fart  $v_b^n$  og posisjon  $\underline{p}_b^n$ , samt flyets orientering  $\underline{\theta}_b^n$ . Vinkelhastigheten  $\underline{\omega}_b^{nb}$ , og den spesifikke kraften  $\underline{f}_b^b$  beregnes også i banegeneratoren, men representeres i  $\mathcal{F}^b$ .

Banegeneratoren i denne oppgaven bygger på en tidligere masteroppgave som har brukt baneelementene rett linje, Eulerspiral, sirkelbane og omvendt Eulerspiral[11]. I denne oppgaven skal de samme baneelementene, etter å ha blitt utvidet til tredimensjonale, brukes til å lage en kontinuerlig bane i rommet. Det vil også være nødvendig å definere planet som flyet svinger på, dette planet står vinkelrett på flyets vinger.

#### 3.1 Baneelementene

Denne seksjonen viser hvordan baneelementene er definert, samtlige representert i sin lokale  $\mathcal{F}^a$  ramme. Disse skal være kontinuerlige i rom og tid i når de skjøtes sammen i  $\mathcal{F}^n$ . Figur 2 på neste side viser hvordan 3-D banen er tenkt å se ut i  $\mathcal{F}^n$ . Her ser man at banen består av en rekke baneelementer som er skjøtet sammen for å gi en kontinuerlig bane. De hvite sirklene som dukker opp langs banen angir start og slutt punkt for hvert baneelement. Banen består av fire baneelementer som blir presentert i det følgende.



Figur 2: Fire banelementer utgjør hele banen. Sirklene viser start og slutt for hver baneelement

### 3.1.1 Rett bane med og uten akselerasjon

Akselerasjonen langs  $\vec{a}_1$  sett fra  $\mathcal{F}^a$  rammen er gitt ved

$$a_1^a(t) = A(1 - \cos\omega t) \quad (3.1)$$

der  $A = (V_0 - V)/T$  dersom farten øker fra  $V_0$  til  $V$ ,  $T$  er tiden flyet bruker på å komme seg gjennom baneelementet og  $\omega = 2\pi/T$ . Akselerasjonen integreres til å gi hastighet og posisjon, henholdsvis

$$v_1^a(t) = v_1^a(0) + A \left(1 - \frac{1}{\omega} \sin\omega t\right) \quad (3.2)$$

og

$$p_1^a(t) = v_1^a(0)t + \frac{1}{2}At^2 + \frac{1}{\omega^2}(\cos\omega t - 1) \quad (3.3)$$

Disse ligningene gir oss tre vektorer for akselerasjon, hastighet og posisjon som vist nedenfor.

$$\underline{p}^a(t) = \begin{bmatrix} V_0 t + \frac{1}{2}At^2 + \frac{1}{\omega^2}(\cos\omega t - 1) \\ 0 \\ 0 \end{bmatrix} \quad (3.4)$$

$$\underline{v}^a(t) = \begin{bmatrix} V_0 + A \left(1 - \frac{1}{\omega} \sin\omega t\right) \\ 0 \\ 0 \end{bmatrix} \quad (3.5)$$

$$\underline{a}^a(t) = \begin{bmatrix} A(1 - \cos\omega t) \\ 0 \\ 0 \end{bmatrix} \quad (3.6)$$

### 3.1.2 Eulerspiral

Overgangen mellom rettlinjert bevegelse og sirkelbevegelse er gitt av en Eulerspiral. Eulerspiralen krummer lineært og proporsjonalt med kurvens lengde og gir dermed en myk overgang[11]. Akselerasjonenes tangentialkomponent er lik 0 slik at slik at den tangentielle farten gjennom baneelementet er konstant. For å finne posisjonen brukes fresnelintegralene som er Matlab kan løse med *fresnelc* for sinusfunksjonen og *fresnelc* for cosinusfunksjonen. Utledning av Eulerspiralen tar utgangspunkt i sammenhengen mellom hastigheten relativt  $\mathcal{F}^a$  representert i enten  $\mathcal{F}^a$  eller  $\mathcal{F}^b$  som i 2-D tilfellet er[14]

$$\underline{v}^a(t) = R_b^a \underline{v}^{ab}(t), \quad R_b^a(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \quad (3.7)$$

Generelt gir dette for akselerasjon og posisjon

$$\underline{a}^a(t) = \dot{\theta} \begin{bmatrix} -\sin(\theta) & -\cos(\theta) \\ \cos(\theta) & -\sin(\theta) \end{bmatrix} \underline{v}^{ab}(t) + \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \underline{\dot{v}}^{ab}(t) \quad (3.8)$$

$$\underline{p}^a(t) = \underline{p}_0^a + \int_0^t \underline{v}^a(s) ds \quad (3.9)$$

Dersom man antar kun normalakselerasjon og konstant tangentiell fart langs banen kan man skrive

$$\underline{v}^a(t) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} V \\ 0 \end{bmatrix} = V \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \quad (3.10)$$

$$\underline{a}^a(t) = \dot{\theta} V \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix} = a(t) \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix} \quad (3.11)$$

Dersom akselerasjonen øker lineært fra 0 til  $A$  i løpet av  $T$  sekunder

$$a(t) = \frac{At}{T} = \dot{\theta} V, \quad \dot{\theta} = \frac{At}{VT} \quad (3.12)$$

$$\theta(t) = \frac{At^2}{2VT} \quad (3.13)$$

$$\underline{v}^a(t) = V \begin{bmatrix} \cos(\theta(t)) \\ \sin(\theta(t)) \end{bmatrix}, \quad \underline{p}^a(t) = \int_0^t V \begin{bmatrix} \cos\left(\frac{As^2}{2VT}\right) \\ \sin\left(\frac{As^2}{2VT}\right) \end{bmatrix} ds, \quad \underline{a}^a(t) = \frac{At}{T} \begin{bmatrix} -\sin(\theta(t)) \\ \cos(\theta(t)) \end{bmatrix}$$

(3.14)

Det er ønskelig å uttrykke  $\underline{p}^a(t)$  slik for å kunne løse Fresnelintegralene i Matlab. Programmene *fresnelc* og *fresnelc* i Matlab løser Fresnelintegralene når de er definert

slik[14]

$$S(t) = \int_0^t \sin\left(\frac{\pi\tau^2}{2}\right) d\tau, \quad S_1(t) = \sqrt{\frac{2}{\pi}} \int_0^t \sin(\tau^2) d\tau \quad (3.15)$$

$$C(t) = \int_0^t \cos\left(\frac{\pi\tau^2}{2}\right) d\tau, \quad C_1(t) = \sqrt{\frac{2}{\pi}} \int_0^t \cos(\tau^2) d\tau \quad (3.16)$$

I posisjonslikningen må integrasjonsvariabelen  $s$  endres til  $\tau$  for å få fresnelintegraler

$$\sqrt{\frac{\pi}{2}}\tau = s \sqrt{\frac{A}{2VT}}, \quad \sqrt{\frac{\pi}{2}}d\tau = ds \sqrt{\frac{A}{2VT}} \quad (3.17)$$

$$\tau = s \sqrt{\frac{A}{\pi VT}}, \quad ds = d\tau \sqrt{\frac{\pi VT}{A}} \quad (3.18)$$

$$\underline{p}^a(t) = \int_0^t V \begin{bmatrix} \cos\left(\frac{As^2}{2VT}\right) \\ \sin\left(\frac{As^2}{2VT}\right) \end{bmatrix} ds = V \int_0^t \sqrt{\frac{A}{\pi VT}} \begin{bmatrix} \cos\left(\frac{\pi\tau^2}{2}\right) \\ \sin\left(\frac{\pi\tau^2}{2}\right) \end{bmatrix} \sqrt{\frac{\pi VT}{A}} d\tau \quad (3.19)$$

Tiden  $T$  som flyet skal bruke i Eulerspiralen bestemmes av antalladianer flyet skal svinge. Svingen er gitt ved

$$\theta(t) = \frac{At^2}{2VT} \quad (3.20)$$

til en sluttverdi ved tiden  $T$  gitt ved

$$\theta(T) = \frac{AT}{2V} \quad (3.21)$$

og akselerasjonens normalkomponent øker fra 0 til  $A$  iløpet av  $T$ . Argumentet til fresnelintegralene er gitt ved

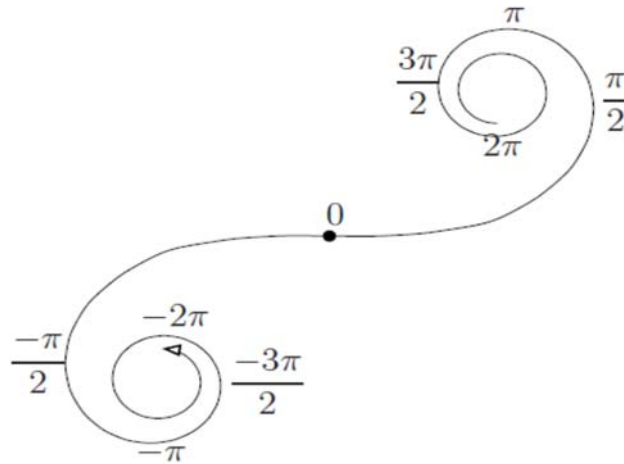
$$\Omega = \sqrt{\frac{A}{\pi VT}} \quad (3.22)$$

Eulerspiralen kan enkelt utvides til 3D ved å utvide posisjonsvektoren, hastighetsvektoren og akselerasjonsvektoren til tre komponenter. Oppsumert kan man skrive for 3D tilfellet

$$\underline{v}_0^a = \begin{bmatrix} V \\ 0 \\ 0 \end{bmatrix}, \text{ og initialverdien til posisjonen settes til } \underline{p}_0^a = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\underline{p}^a(t) = \frac{V}{\Omega} \begin{bmatrix} C(\Omega t) \\ S(\Omega t) \\ 0 \end{bmatrix}, \quad \underline{v}^a(t) = V \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \\ 0 \end{bmatrix}, \quad \underline{a}^a(t) = \frac{At}{T} \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \\ 0 \end{bmatrix} \quad (3.23)$$

Der C er S er fresnelintegralene fresnelc og fresnels. Det kan sees at spiralen fremdeles er på planet. Figur 3 viser hvordan Eulerspiralen krummer for å gi en myk overgang mellom en rett linje og en sving.



Figur 3: Illustrert Eulerspiral for forskjellige vinkler

### 3.1.3 Sirkelbane

Utleddingen av sirkelbanen gjøres for 2-D tilfellet og kan, som for Eulerspiralen, enkelt gjøres om til å gjelde i rommet. Sirkelbevegelsen skal ha konstant banefart  $V$  og normalakselerasjonen

$$A = \frac{V^2}{R} \quad (3.24)$$

Tiden  $T_\omega$  for en  $2\pi$  sving er

$$T_\omega = \frac{2\pi R}{V} \quad (3.25)$$

Vinkelhastigheten

$$\omega = \dot{\theta} = \frac{2\pi}{T_\omega} = \frac{V}{R} = \frac{A}{V} \quad (3.26)$$

Antar at bevegelsen starter langs  $\vec{a}_1$  og svinger mot venstre.

Posisjonen til flyet er

$$\underline{p}^a(t) = \int_0^t V \begin{bmatrix} \cos(\omega\tau) \\ \sin(\omega\tau) \end{bmatrix} d\tau = V \begin{bmatrix} \frac{1}{\omega} \sin(\omega t) \\ -\frac{1}{\omega} \cos(\omega t) \end{bmatrix} = \frac{V}{\omega} \begin{bmatrix} \sin(\omega t) \\ 1 - \cos(\omega t) \end{bmatrix} = R \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix}$$

$$(3.27)$$

Flyets sving i grader er gitt ved

$$\theta(t) = \omega t \quad (3.28)$$

radianer etter  $t$  sekunder. Igjen er den totale tiden i svingen gitt av antall radianer flyet skal svinge, dvs

$$\theta(T) = \frac{AT}{V} \quad (3.29)$$

Oppsumert blir bevegelseslikningene for sirkelbane i rommet

$$\underline{p}^a(t) = R \begin{bmatrix} \sin(\theta) \\ 1 - \cos(\theta) \\ 0 \end{bmatrix}, \quad \underline{v}^a(t) = V \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \\ 0 \end{bmatrix}, \quad \underline{a}^a(t) = \frac{At}{T} \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \\ 0 \end{bmatrix} \quad (3.30)$$

### 3.1.4 Omvendt Eulerspiral

Overgangen mellom sirkelbane og rettlinjet bane er gitt av omvendt Eulerspiral. Akselerasjonen avtar fra  $A$  til 0 og bevegelsen går fra en krum linje til en rett linje. Fra Eulerspiralen ble det vist at dersom akselerasjonsvektoren kun står vinkelrett på hastighetsvektoren slik at den tangentielle farten holdes konstant, får man

$$\underline{v}^a(t) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} V \\ 0 \end{bmatrix} = V \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix}$$

$$\underline{a}^a(t) = \dot{\theta} V \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix} = a(t) \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix}$$

For Eulerspiralen lot man akselerasjonen gå fra 0 til  $A$ . Dersom man istedet lar akselerasjonen gå lineært fra  $A$  til 0 får man

$$a(t) = \frac{A(T-t)}{T} = \dot{\theta} V \quad (3.31)$$

$$\dot{\theta}(t) = \frac{A}{VT}(T-t) \quad (3.32)$$

$$\theta(t) = \int_0^t \frac{A}{VT}(T-\tau)d\tau = \frac{A}{VT} \left( \int_0^t T d\tau - \int_0^t \tau d\tau \right) = \frac{A}{V} \left( t - \frac{t^2}{2T} \right) \quad (3.33)$$

Akselerasjonen og hastigheten blir

$$\underline{v}^a(t) = V \begin{bmatrix} \cos(\theta(t)) \\ \sin(\theta(t)) \end{bmatrix} \quad (3.34)$$

$$\underline{a}^a(t) = \frac{A(T-t)}{T} \begin{bmatrix} -\sin(\theta(t)) \\ \cos(\theta(t)) \end{bmatrix} \quad (3.35)$$



når

$$\theta(t) = \frac{A}{V} \left( t - \frac{t^2}{2T} \right) \quad (3.36)$$

For posisjonen kan man bruke løsningen for Eulerspiralen I likning 3.23 ved å speile banen om  $\vec{b}_2$  vektoren i rammen  $\mathcal{F}^b(T)$  og snu tiden for Eulerspiralen slik at  $t = (T - t)$ . Den nye banen som har blitt funnet i rammen til Eulerspiralen ( $\mathcal{F}^a$ ) kalles for omvendt Eulerspiral[14]. Siden det er ønskelig å kalle rammen til den omvendte Eulerspiralen for  $\mathcal{F}^a$ , må rammen til Eulerspiralen døpes om til feks  $\mathcal{F}^h$ . Man antar at man kan beregne  $\underline{p}^h(t)$  i den nye  $\mathcal{F}^a$  rammen.

$$\Omega = \sqrt{\frac{A}{\pi VT}} \quad (3.37)$$

$$\underline{p}^h(t) = \frac{V}{\Omega} \begin{bmatrix} C(\Omega t) \\ S(\Omega t) \end{bmatrix} \quad (3.38)$$

$$\underline{p}^h(t) = \underline{p}_{Oc}^h + R_h^a \underline{p}_a(t) = \underline{p}^h(T) + R_a^h \underline{p}^a(t) \quad (3.39)$$

$$\underline{p}^a(t) = R_h^a (\underline{p}_h(t) - \underline{p}^h(T)) \quad (3.40)$$

For at banen skal starte i  $O_a$  med retning  $\vec{a}_1$  må  $t$  erstattes med  $(T - t)$  og  $\vec{a}_1$  komponenten må bytte fortegn. Det gir

$$\underline{p}^a(t) = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} R_h^a (\underline{p}^h(T - t) - \underline{p}^h(T)) \quad (3.41)$$

$$= \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} R_3 \left( -\frac{AT}{2V} \right) \frac{V}{\Omega} \begin{bmatrix} C(\Omega(T - t)) - C(\Omega T) \\ S(\Omega(T - t)) - S(\Omega T) \end{bmatrix} \quad (3.42)$$

$$= \frac{V}{\Omega} \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\left(\frac{AT}{2V}\right) & \sin\left(\frac{AT}{2V}\right) \\ -\sin\left(\frac{AT}{2V}\right) & \cos\left(\frac{AT}{2V}\right) \end{bmatrix} \quad (3.43)$$

$$= \frac{V}{\Omega} \begin{bmatrix} -\cos\left(\frac{AT}{2V}\right) & -\sin\left(\frac{AT}{2V}\right) \\ -\sin\left(\frac{AT}{2V}\right) & \cos\left(\frac{AT}{2V}\right) \end{bmatrix} \begin{bmatrix} C(\Omega(T - t)) - C(\Omega T) \\ S(\Omega(T - t)) - S(\Omega T) \end{bmatrix} \quad (3.44)$$

Svingen i radianer er gitt av

$$\theta = \frac{A}{V} \left( t - \frac{t^2}{2T} \right) \quad (3.45)$$

Og den totale svingen ved sluttiden

$$\theta_T = \theta(T) = \frac{AT}{2V} \quad (3.46)$$

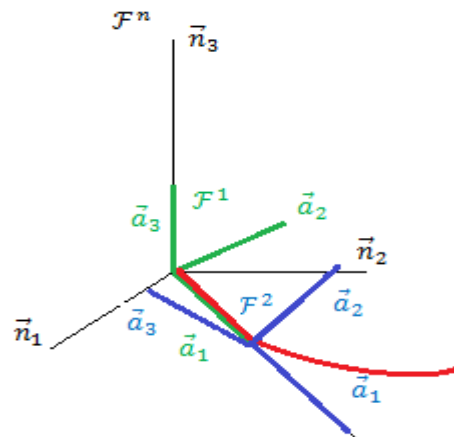
En utvidelse av vektorkomponentene gir omvendt Eulerspiral som kan brukes i rommet  
 Bevegelsesligningene for omvendt Eulerspiral er som følger

$$\underline{p}^a(t) = \frac{v}{\Omega} \begin{bmatrix} -\cos(\theta_T) & -\sin(\theta_T) & 0 \\ -\sin(\theta_T) & \cos(\theta_T) & 0 \\ 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} C(\Omega(T-t)) - C(\Omega T) \\ S(\Omega(T-t)) - S(\Omega T) \\ 0 \end{bmatrix} \quad (3.47)$$

$$\underline{v}^a(t) = V \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \\ 0 \end{bmatrix}, \underline{a}^a(t) = \frac{A(T-t)}{T} \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \\ 0 \end{bmatrix} \quad (3.48)$$

### 3.2 Transformasjon mellom koordinatsystemer

Posisjonen, hastighet og akselerasjon til flyet blir funnet i planet i hvert baneelement. En kontinuerlig bane i  $\mathcal{F}^n$  systemet fåes når de lokale  $\mathcal{F}^a$  rammene blir transformert til  $\mathcal{F}^n$  rammen og dette gjøres ved hjelp av KTM. Sett fra  $\mathcal{F}^n$  rammen får man en kontinuerlig bane ved å skjøte neste baneelement der forrige sluttet. Resultatet er at flyets bevegele, opprinnelig definert i planet, blir transformert til rommet. Dette er illustrert i figur 4.



Figur 4: Sammenhengen mellom flyets bevegelse i planet og rommet

KTM fra  $\mathcal{F}^1$  rammen til  $\mathcal{F}^n$  rammen er gitt ved  $R_1^n$  slik at akselerasjonen og hastigheten i  $\mathcal{F}^1$  sett ifra  $\mathcal{F}^n$  er gitt ved hhv

$$\underline{a}^n = R_1^n \underline{a}^1$$

og

$$\underline{v}^n = R_1^n \underline{v}^1$$

Fra ramme  $\mathcal{F}^2$  til  $\mathcal{F}^n$  er den totale rotasjonen for akselerasjon og hastighet

$$\underline{a}^n = R_1^n R_2^1 \underline{a}^2$$

og

$$\underline{v}^n = R_1^n R_2^1 \underline{v}^2$$

Slik at koordinattransformasjonsmatrisen

$$R_k^n = R_1^n R_2^1 R_3^2 \dots R_k^{k-1} \quad (3.49)$$

Farten og akselerasjonen er like stor sett ifra de forskjellige rammene, men for posisjonen må man ta med en translasjonsvektor fra origo i  $\mathcal{F}^n$  til origo i  $\mathcal{F}^a$ , slik at

$$\underline{p}^n(t) = \underline{p}_{na}^n + R_a^n \underline{p}^a(t^a + T_0) \quad (3.50)$$

Der  $t^a \in [0, T^a]$  og  $T_0$  er starttidspunktet.

For eksempel er posisjonen i  $\mathcal{F}^1$  representert i  $\mathcal{F}^n$  gitt ved

$$\underline{p}^n(t) = \underline{p}_{n1}^n + R_1^n \underline{p}^1(t^1 + T_0) \quad (3.51)$$

### 3.3 Spesifikk kraft

Et akselerometer skiller ikke mellom akselerasjon relatert treghetsrommet og gravitasjonskrefter[9]. En måling fra et akselerometer vil være summen av disse, som kalles for spesifikk kraft. Den spesifikke kraften er altså ikke en kraft, men en akselerasjon. Den har benevnning  $\frac{m}{s^2}$ . Det gjøres en forenkling ved å anta at målingene avleses i  $\mathcal{F}^b$  rammen, men representert i  $\mathcal{F}^n$  ser likningen for den spesifikke kraften slik ut[9].

$$\underline{f}^n = \underline{a}^n + \underline{g}^n \quad (3.52)$$

For å kunne kompensere målingene i et bestemt tidspunkt må man kunne modellere gravitasjonskreftene fordi man ønsker  $\underline{a}^n$  som den målte verdien. Når gravitasjonskreftene er modellert og den spesifikke kraften målt, sitter man igjen med akselerasjonen man ønsker og den kan dobbeltintegreres til å gi posisjonen.

$$\underline{a}^n = \underline{f}^n - \underline{g}^n \quad (3.53)$$

### 3.6 Vinkelfart

Gyroskopet er et hjelpemiddel for å finne stillingen, eller retningen til et fartøy i rommet. Gyroen er en sensor som måler vinkelhastighet sett fra treghetsrommet, men også her forenkler man ved å anta at målingene i avleses i  $\mathcal{F}^b$  rammen[9].

Banegeneratoren sender beregnede verdier av  $\underline{\omega}_b^{nb}$  og  $\underline{f}^b = R_n^b \underline{a}^n + R_n^b \underline{g}^n$  til TNS systemet. For å finne disse er det behov for en sammenheng mellom  $\mathcal{F}^n$  og  $\mathcal{F}^b$ . Det er tidligere vist at  $R_b^n$  kan uttrykkes ved hjelp av basisvektorene til b-systemet slik

$$R_b^n = [\underline{b}_1^n, \underline{b}_2^n, \underline{b}_3^n]$$

Siden flyet flyr «rent», dvs  $\vec{f}$  ligger i  $\vec{b}_1 - \vec{b}_3$  planet og  $\vec{b}_2$  står  $\perp$  på planet for  $\vec{b}_1$  og  $\vec{f}$ , der  $\vec{b}_1$  har samme retning som  $\vec{v}_b^n$ , kan de ortonormale basisvektorene finnes ved hjelp av kryssprodukt.

$$\underline{b}_1^n = \underline{v}_b^n / \|\underline{v}_b^n\| \quad (3.54)$$

$$\underline{b}_2^n = \underline{f}^n \times \underline{b}_1^n / \|\underline{f}^n \times \underline{b}_1^n\| \quad (3.55)$$

$$\underline{b}_3^n = \underline{b}_1^n \times \underline{b}_2^n / \|\underline{b}_1^n \times \underline{b}_2^n\| \quad (3.56)$$

Den tidsvariante retningskosinmatrisen  $R_b^n$  må bestemmes for hvert tidsinkrement, og den deriverte av RKM er i beregnet numerisk

$$\dot{R}_b^n = \lim_{\Delta t \rightarrow 0} \frac{1}{\Delta t} (R_b^n(t + \Delta t) - R_b^n(t)) \quad (3.57)$$

der  $\Delta t = 0.000001$  i Matlab programmet og må ikke forveksles med programmets samplingstid.

Den deriverte av RKM er også gitt ved

$$\dot{R}_b^n = R_n^b S(\underline{\omega}_b^{nb}) \quad (3.58)$$

og det er da mulig å bestemme den skjevsymmetriske matrisen

$$S(\underline{\omega}_b^{nb}) = R_n^b \dot{R}_b^n, \quad R_n^b = R_b^{nT} \quad (3.59)$$

og elementene i den skjevsymmetriske matrisen

$$S(\underline{\omega}_b^{nb}) = \begin{bmatrix} 0 & -\omega_z^{nb} & \omega_y^{nb} \\ \omega_z^{nb} & 0 & \omega_x^{nb} \\ -\omega_y^{nb} & \omega_x^{nb} & 0 \end{bmatrix} \quad (3.60)$$

kan settes sammen til en vektor slik at man har vinkelfarten i hvert tidspunkt

$$\underline{\omega}_b^{nb} = \begin{bmatrix} \omega_x^{nb} \\ \omega_y^{nb} \\ \omega_z^{nb} \end{bmatrix}$$

I første omgang genererer banegeneratoren støyfrie målinger av den spesifikke kraften og vinkelfarten. Dersom prosessmodellen for TNS systemet er korrekt skal de beregnede verdiene for posisjon, hastighet og orientering i TNS systemet samsvare med verdiene fra banegeneratoren.

### 3.7 Banegeneratoren i Matlab

Matlab programmet som skal simulere oppgaven består av et hovedprogram og en rekke funksjoner. Det første brukeren gjør er å velge banetype og type støy som skal legges til i simuleringen. Det er mulig å generere en banegenerator for en sirkel i planet eller en 3D bane i rommet. For støyen er det mulig å velge kun hvit støy, kun farget støy eller hvit + farget støy. Dette gjøres ved hjelp av et array som vist

```
VD=[1 11 12 13;
     2 21 22 23];
```

der verdiene i arrayet svarer til seleksjon av bane og støy som vist

```
%[1 = Sirkelbane 11 = Hvit+Farget 12 = Hvit 13 = Farget]
%[2 = 3D bane 21 = Hvit+Farget 22 = Hvit 23 = Farget]
```

Banegeneratoren består som kjent av sammensatte baneelementer. Egenskapene til banegeneratoren blir satt inn i et array der hver linje i arrayet representerer egenskapene til et baneelement. For å ta det enkle tilfellet med sirkelbanen først, den kan sees på som kun ett langt baneelement og egenskapene er gitt ved en liggende vektor

```
BD=[3 3600 100 10 [0 0 0]]; % sirkelbane
```

som representerer

```
 %[Baneelement nummer, tid i baneelementet, fart, akselerasjon [ $\theta_x$   $\theta_y$   $\theta_z$ ] ]
```

der  $\theta_x$ ,  $\theta_y$  og  $\theta_z$  angir antall grader den lokale rammen  $\mathcal{F}^a$ , der baneelementet først er representert, skal dreies ved hjelp av 3-2-1 Eulervinkler som gir en KTM. For sirkelbane tilfellet er det ingen dreining da det antas at rammene  $\mathcal{F}^n$  og  $\mathcal{F}^a$  sammenfaller.

For 3-D banen derimot, brukes dreiningen slik at neste baneelement kan følge kursendringen i rommet. Denne dreiningen viser seg å være kritisk å få så nøyaktig som mulig for å fjerne/minimere spikere som oppstår i akselerasjonen og vinkelhastigheten, og plutselige hopp ved skjøtepunktene av baneelementene. Her ble det eksperimentert mye for å gjøre banen så kontinuerlig som mulig.

Banegeneratoren for 3-D banen har lik oppsett for hver linje, dvs hvert baneelement, men 3-D banen er sammensatt av en rekke baneelementer. Arrayet til 3-D banen (som kun er en kombinasjon av fire baneelementer) ser slik ut

```

%[Nr, T,          V0,          V/A          [ $\theta_x$   $\theta_y$   $\theta_z$ ] ]
BD=[1 143         [s] 90 [m/s] 100 [m/s]   [-2*0.087 0 pi/2]   [rad]
   2 314.16       [s] 100 [m/s] 1 [m/s^2] [-0.087 0 0]       [rad]
   3 157.08       [s] 100 [m/s] 1 [m/s^2] [0 0 1.5708]     [rad]
   4 314.16       [s] 100 [m/s] 1 [m/s^2] [0 0 1.5708]     [rad]
   5 150          [s] 100 [m/s] 100 [m/s]   [-pi 0 1.5708]     [rad]
   6 418.88       [s] 100 [m/s] 1 [m/s^2] [0 0 0]           [rad]
   7 209.44       [s] 100 [m/s] 1 [m/s^2] [0 0 2.0944]     [rad]
   8 418.88       [s] 100 [m/s] 1 [m/s^2] [0 0 2.0944]     [rad]
   9 200          [s] 100 [m/s] 100 [m/s]   [-1.5*5*pi/6 0 2.0944] [rad]
  10 471.24       [s] 100 [m/s] 1 [m/s^2] [0 0 0]           [rad]
  11 235.62       [s] 100 [m/s] 1 [m/s^2] [0 0 2.3562]     [rad]
  12 471.24       [s] 100 [m/s] 1 [m/s^2] [0 0 2.3562]     [rad]
  13 100          [s] 100 [m/s] 100 [m/s]   [pi 0 2.3562]]    [rad]
```

Brukeren har også mulighet til generere kun et banelement for feilsøking.

Her kan man se at i det første baneelementet, som er en rett linje med akselerasjon, øker startfarten fra 90 m/s til slutfarten 100 m/s i løpet av 143 sekunder. For baneelement nr fem derimot er både start og slutfart lik 100 m/s og det er følgelig ingen akselerasjon.

Kolonne fire er altså slutfart for alle tilfeller av rett bane og akselerasjon for de andre baneelementene.

Dette arrayet bestemmer banen flyet skal følge i rommet, mens akselerasjonen, farten og posisjonen blir regnet ut i egne funksjoner som kalles fra banegeneratoren.

Den første funksjonen hovedprogrammet kaller er banegeneratorfunksjonen «BaneGen» og det første som gjøres her er initialiseringer av blant annet start hastighet, posisjon og stilling. Det denne funksjonen gjør er å trekke ut hver linje fra BD arrayet sekvensielt og ut ifra informasjonen genererer den to baneelementer, et ved tiden  $t$  og et ved tiden  $t + d_d$  der  $d_d = 0.000001$ . Dette gjøres for å kunne beregne vinkelhastigheten numerisk som blir vist senere.

### 3.8 Rett bane i Matlab

Det første baneelementet som skal genereres er rett bane elementet med akselerasjon og det gjøres av funksjonen «RBA»

$$[p\_a, v\_a, a\_a] = RBA(t, T, V, AV)$$

som returnerer akselerasjon, hastighet og posisjon. Av BD kan man se at denne rette linjen er rotert  $\pi/2$  radianer om basisvektoren  $\vec{a}_3$  tilhørende  $\mathcal{F}^a = \mathcal{F}^1$  rammen og  $\pi/18$  radianer om basisvektoren  $\vec{a}_1$  (dette er enklere å se i figur 4 for rotasjon om  $\vec{a}_3$ , i figuren er det dog ingen rotasjon om  $\vec{a}_1$  før flyet når  $\mathcal{F}^2$ ).

Man kan også se for seg den nye orienteringen ved hjelp av høyrehåndsregelen. Koordinattransformasjonsmatrisen  $R_a^n$  fåes også fra disse dreiningene da man tenker at disse var opprinnelig sammenfallende. Da er det enkelt å finne farten

$$\underline{v}^a = R_a^n \underline{v}^n$$

og akselerasjonen

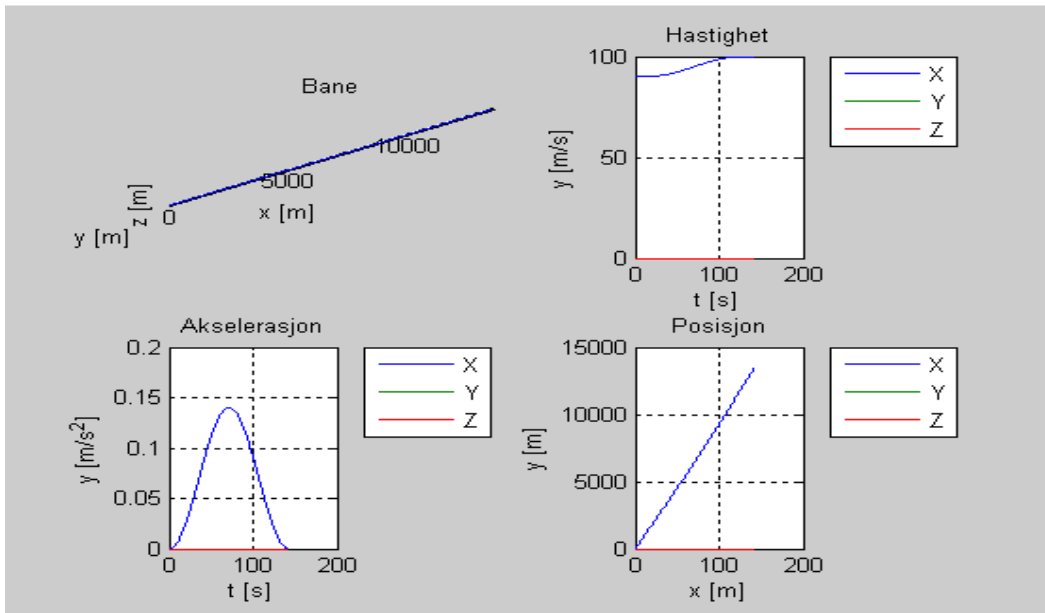
$$\underline{a}^n = R_a^n \underline{a}^a$$

mens posisjonen blir

$$\underline{p}_{na}^n + R_a^n \underline{p}^a$$

Der  $\underline{p}_{na}^n$  er en posisjonsvektor fra  $O_n$  til  $O_a$ .

Figur 5 viser utviklingen av akselerasjon, hastighet, posisjon og en figur av banen. Som figuren viser er det en tangentiell akselerasjon, dette gjelder kun for det første baneelementet.



Figur 5: RBA funksjonen returnerer flyets hastighet, akselerasjon og posisjon

Den spesifikke kraften for baneelementet blir regnet ut som tidligere vist og denne vektoren står vinkelrett på flyets vinger og ligger i  $\underline{b}_1^n - \underline{b}_3^n$  planet.

$$\underline{f}^n = \underline{a}^n + \underline{g}^n$$

Sammen med kunnskap om at  $\underline{v}_b^n$  peker i samme retning som  $\underline{b}_1^n$  og  $\underline{b}_2^n \perp \underline{b}_1^n - \underline{b}_3^n$  planet, finner man basisvektorene til legeme systemet som tidligere vist

$$R_b^n = [\underline{p}_1^n, \underline{p}_2^n, \underline{p}_3^n]$$

Eulervinklene blir regnet utifra stillingsmatrisen i funksjonen  $EV$ , som løser det inverse problem for 3-2-1 Eulervinkler. Det er i overgangen rett bane-eulerspiral og eulerspiral-rett bane at vinkelhastigheten får et hopp i verdien  $\omega_x^{nb}$

### 3.8.1 Eulerspiral i Matlab

Så skal flyet svinge og overgangen mellom rettlinjett bevegelse og sving er gitt ved Eulerspiral. Som tidligere nevnt er tiden i baneelementet bestemt av hvor stor svingen er, slik at

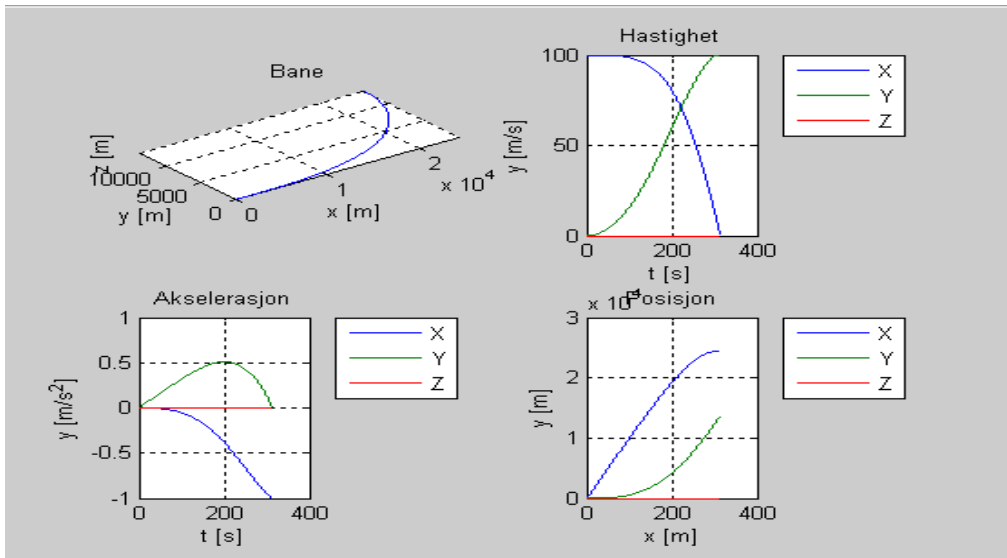
$$\theta(T) = \frac{AT}{2V}, \quad T = \frac{2V\theta(T)}{A}$$

Hvis brukeren velger en sving på  $\theta(T) = \pi/2$  radianer tar det 314.16 sekunder med en konstant tangentiell hastighet på 100 m/s og akselerasjonen 1 m/s<sup>2</sup> (normalakselerasjon), slik det er definert i BD arrayet.

Akselerasjon, hastighet og posisjon blir regnet ut i ESP funksjonen

$$[p\_a, v\_a, a\_a] = \text{ESP}(t, T, V, AV)$$

og representeres i  $\mathcal{F}^n$ . Når det gjelede  $\underline{f}^n$  og  $R_b^n$  blir disse funnet i funksjonen RKM. Figur 6 viser hvordan Eulerspiralen utvikler seg med tiden.



Figur 6: ESP funksjonen returnerer flyets hastighet, akselerasjon og posisjon

### 3.8.2 Sirkelbane i Matlab

Flyet har i Eulerspiralen svingt  $\pi/2$  radianer om  $\underline{a}_3^a$  akse iløpet av 314.16 sekunder. Siden sirkelbaneelementet skal ha origo der Eulerspiralen sluttet må  $\mathcal{F}^a$  rammen til sirkelbaneelementet roteres  $\pi/2$  radianer om  $\underline{a}_3^a$  for at sluttpunktet og startpunktet skal sammenfalle. Dette er lagt inn i BD arrayet.

Flyet skal foreta enda en sving på  $\pi/2$  radianer. Denne svingen er definert til å bruke 157.08 sekunder med den tangentielle hastigheten  $100 \text{ m/s}$  og normalakselerasjonen  $1 \text{ m/s}^2$ .

$$\theta(T) = \frac{AT}{V}, \quad T = \frac{V\theta(T)}{A}$$

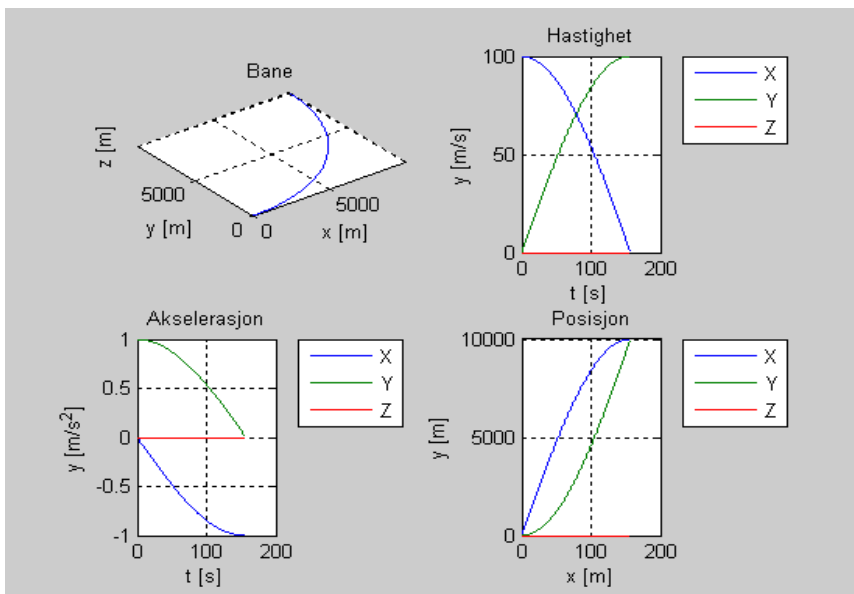
Akselerasjon, hastighet og posisjon blir regnet ut i SBA funksjonen

$$[p\_a, v\_a, a\_a] = \text{SBA}(t, T, V, AV);$$

og representeres i  $\mathcal{F}^n$ . Den spesifikke kraften  $\underline{f}^n$  og  $R_b^n$  blir funnet på samme måte som for forrige banelement. Eulervinklene blir funnet av funksjonen EV.

Det er særlig i overgangen sirkelbane/omvendt Euler at det dukker opp en spiker i vinkelhastigheten.





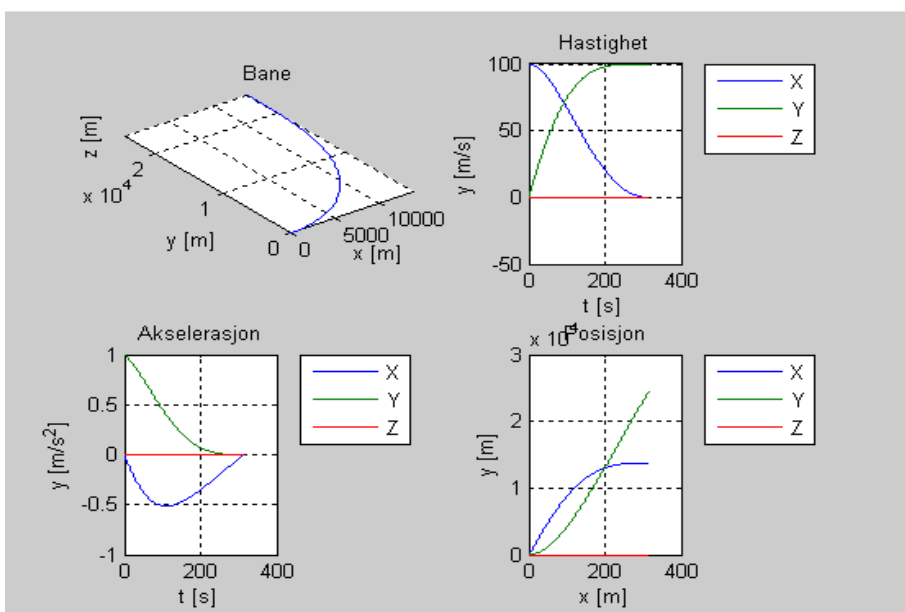
Figur 7: SBA funksjonen returnerer flyets hastighet, akselerasjon og posisjon

### 3.8.3 Omvendt Eulerspiral i Matlab

Flyet skal nå ut av svingen og banen den følger er en omvendt Eulerspiral der normalakselerasjonen minker fra en verdi  $A$  til 0. Flyet brukte 157.08 sekunder på svingen i sirkelbanen som betyr at omvendt Eulerspiral rammen må roteres  $\pi/2$  radianer om  $\underline{a}_3^a$  vektoren for at  $O_a$  i Eulerspiral rammen skal sammenfalle med slutt punktet til sirkelbanen. Akselerasjon, hastighet og posisjon blir regnet ut i funksjonen  $OES$

$$[p\_a, v\_a, a\_a] = OES(t, T, V, AV);$$

og representeres i  $\mathcal{F}^n$ . Den spesifikke kraften  $\underline{f}^n$ ,  $R_b^n$  og Eulervinklene blir funnet på samme måte som for forrige banelement.



Figur 8: OES funksjonen returnerer flyets hastighet, akselerasjon og posisjon

Når alle baneelementene er funnet og banen er skjøttet sammen kan vinkelhastighetene bestemmes ved hjelp av stillingsmatrisene. Som tidligere vist er stillingsmatrisen funnet når man har basisvektorene til b-systemet sett ifra n-systemet

$$R_b^n = [\underline{b}_1^n, \underline{b}_2^n, \underline{b}_3^n]$$

For hvert tidsinkrement ble det funnet to stillingsmatriser, ved tiden  $t$  og  $t + \Delta t$ . Dermed er det mulig å bestemme

$$\dot{R}_b^n = \lim_{\Delta t \rightarrow 0} \frac{1}{\Delta t} (R_b^n(t + \Delta t) - R_b^n(t))$$

Den deriverte av stillingsmatrisen brukes til å finne den skjevsymmetriske matrisen som er gitt ved

$$S(\underline{\omega}_b^{nb}) = R_n^b \dot{R}_b^n, \quad R_n^b = R_b^{nT}$$

$$S(\underline{\omega}_b^{nb}) = \begin{bmatrix} 0 & -\omega_z^{nb} & \omega_y^{nb} \\ \omega_z^{nb} & 0 & \omega_x^{nb} \\ -\omega_y^{nb} & \omega_x^{nb} & 0 \end{bmatrix}$$

De positive vinkelhastighetene plukkes ut slik at man har vinkelfarten for hver tidsinkrement langs hele banen.

$$\underline{\omega}_b^{nb} = \begin{bmatrix} \omega_x^{nb} \\ \omega_y^{nb} \\ \omega_z^{nb} \end{bmatrix}$$

I Matlab gjøres dette i en egen funksjon  $R\_b\_nd$ . Banegeneratoren har nå funnet den spesifikke kraften  $\underline{f}^b$  og vinkelhastigheten  $\underline{\omega}_b^{nb}$  for hele banen. Utdata fra banegeneratoren representert i  $\mathcal{F}^n$  er akselerasjon  $\underline{a}_b^n$ , fart  $\underline{v}_b^n$  og posisjon  $\underline{p}_b^n$ , samt flyets orientering  $\underline{\theta}_b^n$ , vinkelhastighet  $\underline{\omega}_b^{nb}$ , og spesifikke kraft  $\underline{f}^b$ .

Disse blir satt sammen i en matrise for gjøre programmet mer strukturert

$$A = \begin{bmatrix} \underline{p}_b^n \\ \underline{v}_b^n \\ \underline{a}_b^n \\ \underline{f}^b \\ \underline{\omega}_b^{nb} \\ \underline{E}_y \\ t_p \end{bmatrix}$$

Der  $E_y$  er Eulervinklene til hele banen og  $t_p$  er tiden

Disse verdiene blir sendt videre til TNS systemet som først tester  $\underline{f}^b$  og  $\underline{\omega}_b^{nb}$  på en sirkelbane uten støy, for å verifisere den matematiske modellen for TNS systemet. Dersom den eneste feilen er integrasjonsfeilen til den numeriske integrasjonsrutinen, kan vi si at modellen er korrekt.

#### 4 TNS for flat ikke-roterende jord

I treghetsnavigasjon navigerer man ved hjelp av akselerometer og gyroskop. Det betyr at når starttilstanden til flyet er kjent, akselerasjonen er kjent og retningen til akselerasjonen er kjent, kan man til enhver tid integrere seg opp til flyets hastighet og posisjon. Denne oppgaven er begrenset til å navigere på en flat ikke-roterende jord. Navigasjonslikningene er gitt ved

$$\dot{\underline{p}}^n = \underline{\tilde{v}}^n \quad (4.1)$$

$$\dot{\underline{v}}^n = \tilde{\underline{R}}^n \underline{\tilde{f}}^b - \underline{g}^n \quad (4.2)$$

$$\dot{\tilde{\underline{R}}}_b^n = \tilde{\underline{R}}_b^n S(\tilde{\underline{\omega}}_b^{nb}) \quad (4.3)$$

der « $\sim$ » og « $\tilde{\sim}$ » representerer hhv beregnet og målt verdi

Navigasjonslikningene integreres for å finne  $\underline{\tilde{p}}_b^n$ ,  $\underline{\tilde{v}}_b^n$  og  $\tilde{\theta}$ . I denne oppgaven brukes to integrasjonsmetoder, Eulers metode og Heuns metode.

##### 4.1 Integrasjonsrutiner

Opgaven presenterer to to numeriske metoder for integrering av navigasjonslinkningene for et TNS:

- Eulers metode
- Heuns metode

###### 4.1.1 Eulers metode

Eulers metode er en 1.ordens numerisk metode for å løse ordinære differensiallikninger med angitte initialverdier[6]. Gitt en generell skalar differensiallikning, med en gitt initialverdi

$$\dot{x} = f(t, x(t)), \quad x_0 = x(t_0) \quad (4.4)$$

Diskretisert gir dette den generelle skalare løsningen for Eulers metode

$$x_{k+1} = x_k + h * f(t_k, x_k), \text{ der } h \text{ er steplengden} \quad (4.5)$$

Navigasjonslikningene skrevet på standardform gir

$$\dot{\underline{x}} = f(\underline{u}(t), \underline{x}(t), \tilde{\underline{R}}_b^n(t), t), \text{ der } \underline{x}_0 = \underline{x}(t_0) \quad (4.6)$$

$$\dot{\tilde{\underline{R}}}_b^n = f(\underline{u}(t), \tilde{\underline{R}}_b^n(t), t), R_{b,0}^n \text{ gitt} \quad (4.7)$$

$$\text{der } \underline{u}(t) = \begin{bmatrix} \underline{\tilde{f}}^b \\ \tilde{\underline{\omega}}_b^{nb} \end{bmatrix}, \quad \underline{x}(t) = \begin{bmatrix} \underline{\tilde{p}}_b^n \\ \underline{\tilde{v}}_b^n \end{bmatrix}$$

og

$$f(\underline{u}(t), \underline{x}(t), \tilde{R}_b^n(t), t) = \left[ \begin{array}{c} \tilde{v}_b^n(t) \\ \tilde{R}_b^n(t) * \tilde{f}^b(t) - \underline{g}^n \end{array} \right] \quad (4.8)$$

$$f(\underline{u}(t), \tilde{R}_b^n(t), t) = [\tilde{R}_b^n(t) * S(\tilde{\omega}_b^{nb}(t))] \quad (4.9)$$

Diskretiseringen gir

$$\underline{x}_{k+1} = \underline{x}_k + h * f(\underline{u}_k, \underline{x}_k, \tilde{R}_{b,k}^n, t_k), \text{ der } \underline{x}_0 = \underline{x}(t_0) \quad (4.10)$$

$$\tilde{R}_{b,k+1}^n = R_{b,k}^n + h * f(\underline{u}_k, \tilde{R}_{b,k}^n, t_k), R_{b,0}^n \text{ gitt} \quad (4.11)$$

#### 4.1.2 Heuns Metode

Heuns metode er en utvidelse av Eulers metode som gir et mer nøyaktig resultat. Prosedyren for å beregne initialverdi problemet med Heuns metode tar utgangspunkt i Eulers metode[7]. For det skalare tilfellet er Heuns metode gitt ved

Eulers metode

$$x_{k+1} = x_k + h * f(t_k, x_k)$$

Utvidet til Heuns metode

$$x_{k+1} = x_k + \frac{h}{2} * (f(t_k, x_k) + f(t_{k+1}, x_{k+1})) \quad (4.12)$$

Navigasjonslikningene skrevet på standardform gir

$$\dot{\underline{x}} = f(\underline{u}(t), \underline{x}(t), \tilde{R}_b^n(t), t), \text{ der } \underline{x}_0 = \underline{x}(t_0) \quad (4.13)$$

$$\dot{\tilde{R}}_b^n = f(\underline{u}(t), \tilde{R}_b^n(t), t), \text{ der } R_{b,0}^n = R_b^n(t_0) \quad (4.14)$$

$$\text{der } \underline{u}(t) = \left[ \begin{array}{c} \tilde{f}^b \\ \tilde{\omega}_b^{nb} \end{array} \right], \quad \underline{x}(t) = \left[ \begin{array}{c} \tilde{p}_b^n \\ \tilde{v}_b^n \end{array} \right]$$

$$f(\underline{u}(t), \underline{x}(t), \tilde{R}_b^n(t), t) = \left[ \begin{array}{c} \tilde{v}_b^n(t) \\ \tilde{R}_b^n(t) * \tilde{f}^b(t) - \underline{g}^n \end{array} \right] \quad (4.15)$$

$$f(\underline{u}(t), \tilde{R}_b^n(t), t) = [\tilde{R}_b^n(t) * S(\tilde{\omega}_b^{nb}(t))] \quad (4.16)$$

Som etter diskretisering gir

$$\underline{x}_{k+1} = \underline{x}_k + \frac{h}{2} * (f(\underline{u}_k, \underline{x}_k, \tilde{R}_{b,k}^n, t_k) + f(\underline{u}_{k+1}, \underline{x}_{k+1}, \tilde{R}_{b,k+1}^n, t_{k+1})) \quad (4.17)$$

$$\tilde{R}_{b,k+1}^n = R_{b,k}^n + \frac{h}{2} * (f(\underline{u}_k, \tilde{R}_{b,k}^n, t_k) + f(\underline{u}_{k+1}, \tilde{R}_{b,k+1}^n, t_{k+1})) \quad (4.18)$$

Siden Eulermetoden er en 1.ordens metode, vil feilen etter hver tidsskritt være proporsjonal med kvadratet av steplengden  $h$ . Den akkumulerte feilen etter en vilkårlig tid vil da være proporsjonal med  $h$ , dvs integreringsfeilen synker lineært når frekvensen øker. Den akkumulerte feilen vil også gi et avvik fra sann verdi i Heuns tilfellet, derimot vil integreringsfeilen her reduseres kvadratisk når frekvensen øker. Det er da klart at simuleringer med Heuns metode er å foretrekke.

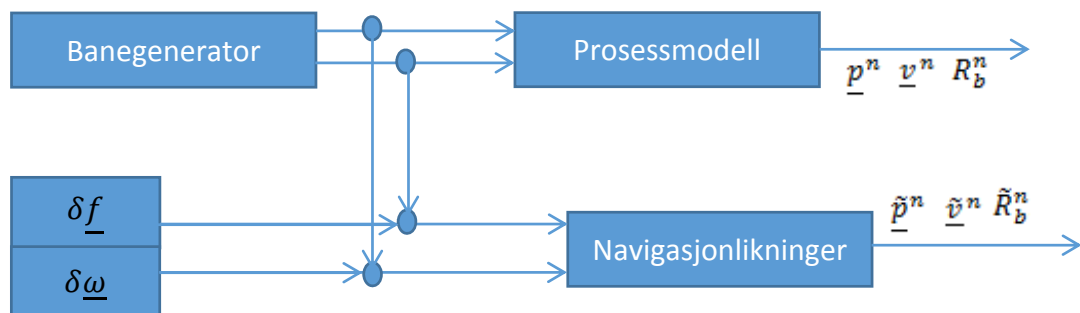
## 4.2 TNS i Matlab

For å teste prosessmodellen til TNS systemet, dvs navigasjonslikningene uten støy, kan det simuleres med støyfrie målinger av vinkelfarten og den spesifikke kraften. Dersom modellen er korrekt skal feilen mellom TNS systemets beregninger og banegeneratoren skyldes integrasjonsrutinen. TNS systemet krever samme initialverdier som banegeneratoren for å fungere riktig. Startverdier for posisjon, hastighet og stilling må initialiseres før integrasjonsrutinene iverksettes. Matlab funksjonen «NAV» integrerer med både Euler og Heuns metode, uten og med støy. Funksjonen returnerer først de støyfrie beregningene av posisjon, hastighet og stilling (attitude)

$$x_{euler} = \begin{bmatrix} \underline{p}^n \\ \underline{v}^n \end{bmatrix}, \quad x_{heuns} = \begin{bmatrix} \underline{p}^n \\ \underline{v}^n \end{bmatrix}$$

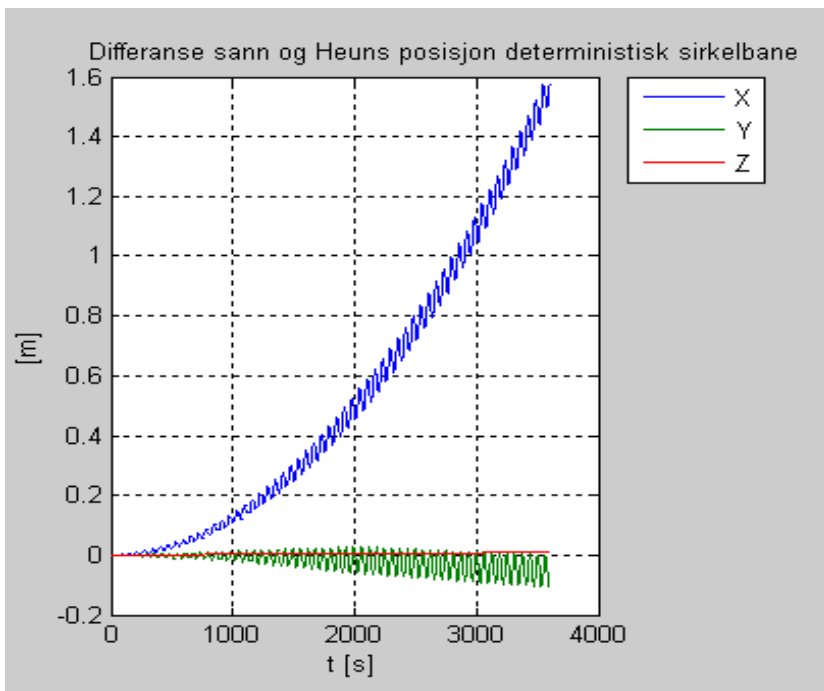
$$att = \begin{bmatrix} \theta_{euler}^n \\ \theta_{euler}^n \end{bmatrix}$$

Figuren under viser støyfrie verdier av  $\underline{f}^b$  og  $\underline{\omega}_n^{nb}$  fra banegeneratoren til prosessmodellen for TNS, og realistiske støybelagte målinger til navigasjonslikningene.



Figur 9: Utgang systemmodellen og navigasjonslikningene

Det ble brukt en sirkelbane i planet for å sammenligne prosessmodellen til TNS'et med banegeneratoren for å teste om TNS likningene i modellen er riktig satt opp. Sirkelbanen har radius 1000 m og konstant tangentiell hastighet på 100 m/s. Heuns integrasjonen viser at integrasjonsfeilen på det meste er 1.6 m i løpet av en time. Neste skritt er å legge støy på sensorene.



Figur 10: Akumulert feil i Heuns integrasjonsmetode etter 3600 sekunder

Det gjøres av funksjonen «Stoy» i Matlab som genererer støy etter modellene vist i neste seksjon. De støybelagte  $\tilde{f}^b$  og  $\tilde{\omega}_b^{nb}$  sendes igjen til NAV funksjonen som returnerer

$$xm_{euler} = \begin{bmatrix} \tilde{p}^n \\ \tilde{v}^n \end{bmatrix}, \quad xm_{heuns} = \begin{bmatrix} \tilde{p}^n \\ \tilde{v}^n \end{bmatrix}$$

$$attm = \begin{bmatrix} \tilde{\theta}_{euler}^n \\ \tilde{\theta}_{heuns}^n \end{bmatrix}$$

Der  $xm_{euler}$  er beregnet posisjon og hastighet med Eulers metode og  $xm_{heuns}$  er beregnet posisjon og hastighet med Heuns metode. Eulervinklene er samlet i attm vektoren, der  $\tilde{\theta}_{euler}^n$  er beregnet med Eulers metode og  $\tilde{\theta}_{heuns}^n$  er beregnet med Heuns.

### 4.3 Støy og støymodellering

Det antas at den spesifikke kraften og akselerasjonen måles av sensorer på flyet. Disse målingene, slik som alle andre målinger gjort av sensorer, vil være befyngt av støy. For å kunne legge til støy på banegenerator verdiene for  $\underline{f}^b$  og  $\underline{\omega}_b^{nb}$  og få realistiske målinger i simuleringen, må støyprosessen kunne modelleres. I denne oppgaven brukes typiske verdier oppgitt for hvit støy og farget støy for akselerometer og gyro. Flyet har tre akselerometere og tre gyroer, en langs hver av flyets tre akser. Støyen som påvirker disse skal være statistisk uavhengige.

#### 4.3.1 Hvit støy

Hvit støy er definert som en stasjonær stokastisk prosess med forventningsverdi lik null og konstant spektraltetthet for alle frekvenser[15]. Autokorrelasjonen til hvit støy kan skrives

$$R_{XX}(t, \tau) = E\{X(t), X(\tau)\} = R_{XX}(\tau) = \frac{1}{2} \pi \int_{-\infty}^{\infty} S_{XX}(\omega) e^{-j\omega\tau} d\omega \quad (4.19)$$

$$E\{\underline{v}(t)\underline{v}^t(\tau)\} = \tilde{Q}\delta(t - \tau) \quad (4.20)$$

og er korrelert kun når  $t = \tau$  da integralet over Dirac pulsen  $\delta$  er lik 1.  $\tilde{Q}$  er spektraltettheten. Spektraltettheten kan skrives

$$S_{XX}(\omega) = \int_{-\infty}^{\infty} R_{XX}(\tau) e^{-j\omega\tau} d\tau \quad (4.21)$$

Men i denne oppgaven er spektraltetthetene oppgitt og  $\tilde{Q}$  brukes som symbol for disse. Det at prosessen er stasjonær betyr at de statistiske egenskapene til signalet ikke vil forandre seg over tid[15]. Siden spektraltettheten er konstant med uendelig båndbredde vil variansen være uendelig stor slik at dersom man kjenner signalet ved et tidspunkt, vil det være umulig å si noe om signalet ved neste tidspunkt. Denne typen støy er vanskelig å modellere og gaussisk hvit støy blir ofte brukt som en tilnærming.

#### 4.3.2 Farget støy

Den fargede støyen beskrives av en 1.ordens Gauss-Markov prosess. En Markov prosess har den egenskapen at sannsynlighetstetthetsfunksjonen (stf) ved et gitt tidspunkt kun avhenger av STF ved forrige tidspunkt[10]. Funksjonen er da gitt ved

$$p[x(t_n)|p(x(t_{n-1}), \dots, x(t_1))] = p[x(t_n)|x(t_{n-1})] \quad (4.22)$$

Hvis STF  $p[x(t_n)|x(t_{n-1})]$  i tillegg er normalfordelt kan vi kalle prosessen en Gauss Markov prosess. Dersom man antar at prosessen startet ved  $t = -\infty$  har den ved  $t = 0$  nådd en stasjonærverdi.

#### 4.4 Modellering av akselerometer

Det antas at støyen i akselerometeret er begrenset til Velocity Random Walk (VRW) og farget støy. VRW er hvit støy som kommer inn på akselerasjonsnivå og kan beskrives med spektraltettheten  $\tilde{Q}$ . Når akselerasjonen, og den ukorrelerte støyen, integreres gjør støyen seg gjeldende som en random walk på hastighetsnivå[13].

VRW kan modelleres slik

$$\dot{\underline{x}} = \underline{v}^a \quad (4.23)$$

$$\underline{x}_0 \sim \mathcal{N}(\underline{\bar{x}}_0, \bar{P}_0), \quad v \sim \mathcal{N}(\underline{0}, \tilde{Q}\delta(t - \tau))$$

og kovariansen

$$\dot{\bar{P}} = F\bar{P} + \bar{P}F^T + G\tilde{Q}G^T = \tilde{Q} \quad (4.24)$$

$$\bar{P} = \tilde{Q}t \quad (4.25)$$

Den fargede støyen beskrives av en 1.ordens Gauss-Markov prosess. Den kan modelleres med den kontinuerlige likningen

$$\dot{\underline{x}} = -\frac{1}{T}\underline{x} + \underline{v} \quad (4.26)$$

der T er en tidskonstant,  $\underline{x}_0 \sim \mathcal{N}(\underline{0}, \bar{P}_0)$ ,  $v \sim \mathcal{N}(\underline{0}, \tilde{Q}\delta(t - \tau))$

$$\dot{\bar{P}} = F\bar{P} + \bar{P}F^T + G\tilde{Q}G^T = -\frac{2}{T}\bar{P} + \tilde{Q} \quad (4.27)$$

der

$$\bar{P}(t) = e^{-\frac{2}{T}t} \bar{P}_0 + \int_0^t e^{-\frac{2}{T}(t-\tau)} \tilde{Q} d\tau$$

som gir

$$\bar{P}(t) = e^{-\frac{2}{T}t} \bar{P}_0 + \frac{\tilde{Q}T}{2} \left(1 - e^{-\frac{2}{T}t}\right) \quad (4.28)$$

og når  $t \rightarrow \infty$  går eksponentialfunksjonene mot 0 og vi sitter igjen med stasjonærverdien

$$\bar{P}_\infty = \frac{\tilde{Q}T}{2} \quad (4.29)$$

$$\tilde{Q} = \frac{2}{T} \bar{P}_\infty \quad (4.30)$$

Der  $\tilde{Q}$  er spektraltettheten til det fargede støyelementet. Støylikningen for akselerometeret er da

$$\delta \underline{\tilde{f}}^a = \underline{v}^a + \underline{\mu}^a \quad (4.31)$$



Ved å bruke likningene ovenfor kan det lages to prosessmodeller for å modellere støyen i et akselerometer. Modellene har formen

$$\dot{\underline{x}}(t) = F\underline{x}(t) + G\underline{v}(t) \quad (4.32)$$

$$\dot{\bar{P}}(t) = F\bar{P}(t) + \bar{P}(t)F^T + G\tilde{Q}G^T \quad (4.33)$$

Setter vi  $\underline{x}^a = \begin{bmatrix} \underline{c}^a \\ \underline{\mu}^a \end{bmatrix}$ ,  $\underline{v}^a = \begin{bmatrix} \underline{v}^a \\ \underline{\nu}^a \end{bmatrix}$ , der c er velocity random walk.

Systemmatrisa er da

$$F^a = \begin{bmatrix} \text{zeros}(3,3) & \text{zeros}(3,3) \\ \text{zeros}(3,3) & -\frac{1}{T^a} * I \end{bmatrix} \quad (4.34)$$

og støymatrisa

$$G^a = \begin{bmatrix} I & \text{zeros}(3,3) \\ I\text{zeros}(3,3) & I \end{bmatrix} \quad (4.35)$$

Vektordifferensiallikningen kan skrives

$$\begin{bmatrix} \dot{\underline{c}}^a \\ \dot{\underline{\mu}}^a \end{bmatrix} = \begin{bmatrix} \text{zeros}(3,3) & \text{zeros}(3,3) \\ \text{zeros}(3,3) & -\frac{1}{T^a} * I \end{bmatrix} \begin{bmatrix} \underline{c}^a \\ \underline{\mu}^a \end{bmatrix} + \begin{bmatrix} I & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I \end{bmatrix} \begin{bmatrix} \underline{v}^a \\ \underline{\nu}^a \end{bmatrix} \quad (4.36)$$

$$\underline{x}_0^a \sim \mathcal{N}(\underline{\bar{x}}_0^a, \bar{P}_0^a), \quad \underline{v}^a \sim \mathcal{N}(\underline{0}, \tilde{Q}^a \delta(t - \tau))$$

og matrisen  $I$  er en 3x3 identitetsmatrise

Verdier som er oppgitt for akselerometeret er

- Tidskonstanten  $T^a = 1200 \text{ s}$
- VRW:  $\tilde{Q}_c = \left(1.47 * \frac{10^{-4}m}{s^{\frac{3}{2}}}\right)^2$  slik at  $[\tilde{Q}] = \left[\frac{m^2}{s^3}\right]$

Siden  $\bar{P}_c = \tilde{Q}_c t$  vil  $\bar{P}(0)$  ved tiden  $t = 0$  være 0 for den hvite støyen og  $[\bar{P}] = \left[\frac{m^2}{s^2}\right]$

- Farget støy:  $\bar{P}_{\mu,\infty} = \frac{\tilde{Q}T}{2} = \left(2.45 * \frac{10^{-4}m^2}{s^2}\right)^2$  som kan rokkeres om til  $\tilde{Q}_\mu = \bar{P}_{\mu,\infty} * \frac{2}{T}$

Kovariansen går direkte inn i stasjonærverdi slik at  $\bar{P}_{\mu,\infty} = \bar{P}_\mu(0)$ .

Spektralitetthetene kan skrives

$$\tilde{Q}^a = \begin{bmatrix} I * \tilde{Q}_c & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I * \tilde{Q}_\mu \end{bmatrix} = \begin{bmatrix} I * \tilde{Q}_c & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I * \bar{P}_{\mu,\infty} * \frac{2}{T} \end{bmatrix} \quad (4.37)$$

Initiell kovarians er

$$P^a(0) = \begin{bmatrix} \text{zeros}(3,3) & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I * \bar{P}_\mu(0) \end{bmatrix} \quad (4.38)$$

Diskretisering av spektralitetthetene gjøres slik

$$Q_c = \tilde{Q}_c * \Delta t \quad (4.39)$$

$$Q_\mu = \frac{\tilde{Q}_\mu T}{2} (1 - e^{(-\frac{2}{T} * \Delta t)}) \quad (4.40)$$

som settes sammen til matrisen

$$Q^a = \begin{bmatrix} I * Q_c & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I * Q_\mu \end{bmatrix} \quad (4.41)$$

$Q_c$  og  $Q_\mu$  beskriver kovariansen til de diskrete hvitstøyelementene

$$\underline{v}^a \sim \mathcal{N}(\underline{0}, Q^a)$$

Etter å ha funnet støymatrisen og systemmatrisen som tidligere vist kan det diskrete systemet skrives

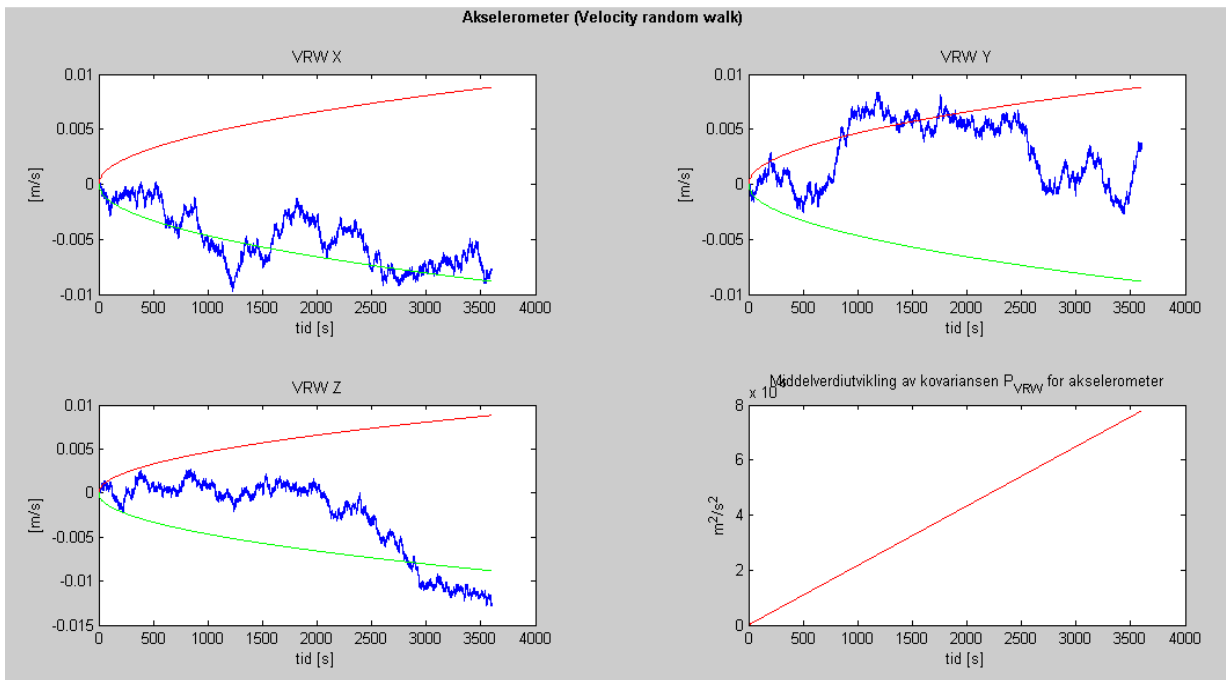
$$\underline{x}_{k+1}^a = \phi^a \underline{x}_k^a + \Gamma^a \underline{v}_k^a \quad (4.42)$$

$$\bar{P}_{k+1}^a = \phi^a \bar{P}_k^a \phi^T + \Gamma^a Q^a \Gamma^{aT} \quad (4.43)$$

Den totale støyen for akselerometeret er da summen av den hvite støyen og den fargede støyen. Men den hvite støyen har nå blitt integrert og gir en random walk på hastighetsnivå. Den totale målefeilen er da gitt ved

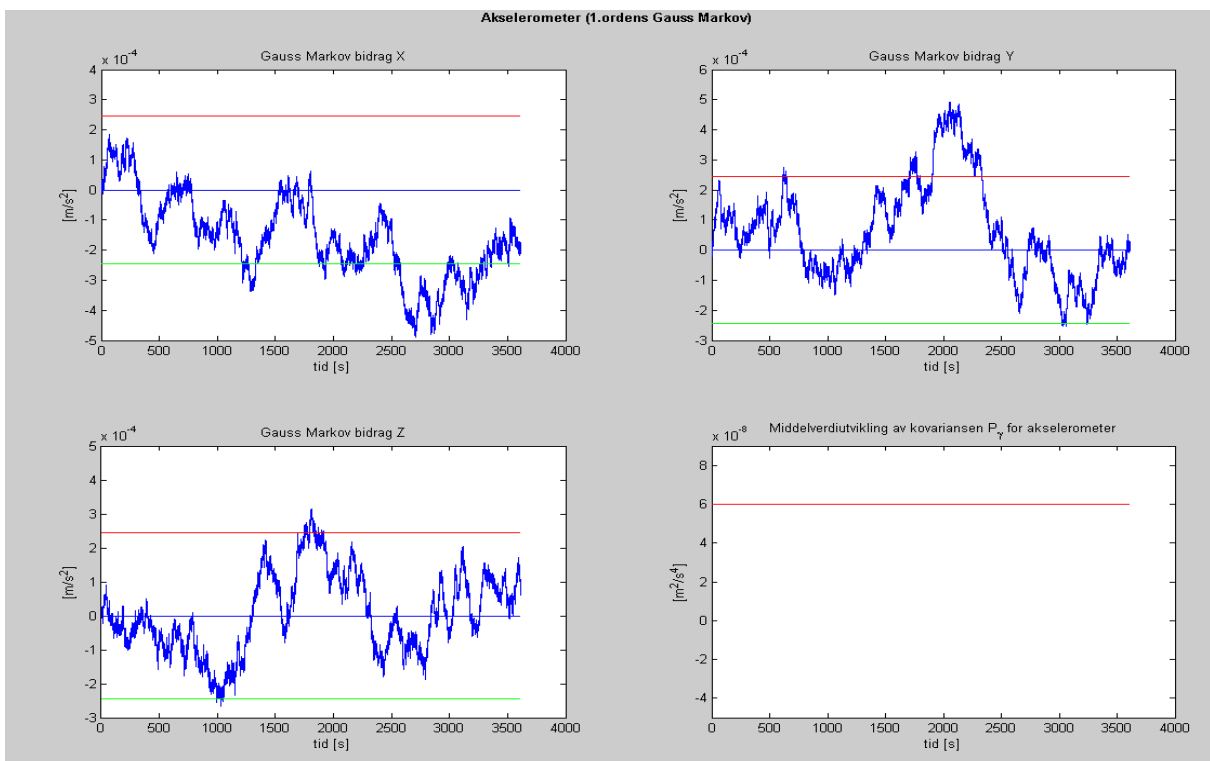
$$\delta \underline{f}_k^a = \frac{(\underline{c}_{k+1}^a - \underline{c}_k^a)}{\Delta t} + \underline{\mu}_k^a \quad (4.44)$$

Det er ønskelig å se hvordan den hvite og fargede støyprosessen påvirker systemet, hver for seg og tilsammen. I Matlab programmet kan brukeren velge å trekke kun hvit støy, kun farget støy eller hvit og farget støy.



Figur 11: Velocity Random Walk (VRW) i akselerometeret

Figur 11 viser den hvite støyen føre til en random walk på hastighetsnivå som er innenfor  $\mp\sigma = 2.45 \times 10^{-4} \text{ m/s}^2$  for det meste av tiden (over  $\frac{2}{3}$  av tiden), og kovariansen øker lineært med tiden.



Figur 12: 1.ordens Gauss Markov støyprosess i akselerometer

Figur 12 viser den fargede støyen som oppstår i akselerometeret. Det kan sees av figuren at middelverdien er 0 og støyen holder seg innenfor  $\mp\sigma = 2.45 \times 10^{-4} \text{ m/s}^2$  det meste av tiden (over  $\frac{2}{3}$  av tiden). Kovariansen går direkte inn i stasjonærverdi

$$P_0 = P_\infty = \left(2.45 \cdot 10^{-4} \frac{m}{s^2}\right)^2 = 6.0025 \cdot 10^{-8} \frac{m^2}{s^4}$$

#### 4.5 Modellering av gyro

Det antas at målingene til gyroskopet er begrenset til Angular Random Walk (ARW) og farget støy. ARW er hvit støy som kommer inn på vinkelhastighetsnivå, blir integrert sammen med vinkelhastigheten og gir en random walk på vinkelnivå. Modelleringen av gyroen har samme fremgangsmåte som for akselerometeret. Men spektraltettheten på den hvite og fargede støyen vil være forskjellig.

ARW kan modelleres slik

$$\dot{\underline{x}} = \underline{v} \quad (4.45)$$

$$\underline{x}_0 \sim \mathcal{N}(\underline{\bar{x}}_0, \bar{P}_0), v \sim \mathcal{N}(\underline{0}, \tilde{Q}\delta(t - \tau))$$

og kovariansen

$$\dot{\bar{P}} = F\bar{P} + \bar{P}F^T + G\tilde{Q}G^T = \tilde{Q} \quad (4.46)$$

der

$$\bar{P} = \tilde{Q}t \quad (4.47)$$

Den fargede støyen beskrives igjen av en 1.ordens Gauss-Markov prosess, som er beskrevet av

$$\dot{\underline{x}} = -\frac{1}{T}\underline{x} + \underline{v} \quad (4.48)$$

der T er en tidskonstant,  $\underline{x}_0 \sim \mathcal{N}(\underline{0}, \bar{P}_0), v \sim \mathcal{N}(\underline{0}, \tilde{Q}\delta(t - \tau))$

$$\dot{\bar{P}} = F\bar{P} + \bar{P}F^T + G\tilde{Q}G^T = -\frac{2}{T}\bar{P} + \tilde{Q} \quad (4.49)$$

Der

$$\bar{P}(t) = e^{-\frac{2}{T}t} \bar{P}_0 + \int_0^t e^{-\frac{2}{T}(t-\tau)} \tilde{Q} d\tau \quad (4.50)$$

som gir

$$\bar{P}(t) = e^{-\frac{2}{T}t} \bar{P}_0 + \frac{\tilde{Q}T}{2} \left(1 - e^{-\frac{2}{T}t}\right) \quad (4.51)$$

og når  $t \rightarrow \infty$  går eksponentialfunksjonene mot 0 og vi sitter igjen med stasjonærverdien

$$\bar{P}_\infty = \frac{\tilde{Q}T}{2} \quad (4.52)$$

$$\tilde{Q} = \frac{2}{T} \bar{P}_\infty \quad (4.53)$$

Der  $\tilde{Q}$  er spektraltettheten til farget støyelementet. Støylikningen for gyroen er da

$$\delta \underline{\tilde{\omega}}^a = \underline{v}^g + \underline{\mu}^g \quad (4.54)$$

Igjen brukes to kontinuerlige prosessmodeller for å modellere støyen i gyroen

$$\dot{\underline{x}}(t) = F\underline{x}(t) + G\underline{v}(t)$$

$$\dot{\underline{P}}(t) = F\underline{P}(t) + \underline{P}(t)F^T + G\underline{Q}G^T$$

Setter vi  $\underline{x}^g = \begin{bmatrix} \underline{\theta}^g \\ \underline{\mu}^g \end{bmatrix}$ ,  $\underline{v}^a = \begin{bmatrix} \underline{v}^g \\ \underline{v}_\mu^g \end{bmatrix}$ , der  $\underline{\theta}$  er angular random walk.

Systemmatrisa er da

$$F^g = \begin{bmatrix} \text{zeros}(3,3) & \text{zeros}(3,3) \\ \text{zeros}(3,3) & -\frac{1}{T^g} * I \end{bmatrix} \quad (4.55)$$

og støymatrisa

$$G^g = \begin{bmatrix} I & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I \end{bmatrix} \quad (4.56)$$

Differensiallikningene på vektorform kan nå skrives

$$\begin{bmatrix} \dot{\underline{\theta}}^g \\ \dot{\underline{\mu}}^g \end{bmatrix} = \begin{bmatrix} \text{zeros}(3,3) & \text{zeros}(3,3) \\ \text{zeros}(3,3) & -\frac{1}{T^g} * I \end{bmatrix} \begin{bmatrix} \underline{\theta}^g \\ \underline{\mu}^g \end{bmatrix} + \begin{bmatrix} I & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I \end{bmatrix} \begin{bmatrix} \underline{v}^g \\ \underline{v}_\mu^g \end{bmatrix} \quad (4.57)$$

$$\underline{x}_0^g \sim \mathcal{N}(\underline{\bar{x}}_0^g, \underline{\bar{P}}_0^g), \quad \underline{v}^g \sim \mathcal{N}(\underline{0}, \underline{Q}^g \delta(t - \tau))$$

Verdier som er oppgitt for gyroskopet er

- Tidskonstanten  $T = 600s$
- ARW:  $\tilde{Q}_\theta = \left(1.76 * \frac{10^{-6} rad}{\sqrt{s}}\right)^2$  slik at  $[\tilde{Q}] = \left[\frac{rad^2}{s}\right]$

Siden  $\underline{\bar{P}}_\theta = \tilde{Q}_\theta t$  vil  $\underline{\bar{P}}_\theta(0)$  ved tiden  $t = 0$  være 0 for den hvite støyen og  $[\underline{\bar{P}}] = [rad^2]$

- Farget støy:  $\underline{\bar{P}}_{\mu,\infty} = \frac{\tilde{Q}T}{2} = \left(1.45 * 10^{-8} \frac{rad}{s}\right)^2$  som kan rokkeres om til  $\tilde{Q}_\mu = \underline{\bar{P}}_{\mu,\infty} * \frac{2}{T}$

Kovariansen går direkte inn i stasjonærverdi slik at  $\underline{\bar{P}}_{\mu,\infty} = \underline{\bar{P}}_\mu(0)$ .

Spektraltetthetene kan skrives

$$\tilde{Q}^g = \begin{bmatrix} I * \tilde{Q}_\theta & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I * \tilde{Q}_\mu \end{bmatrix} = \begin{bmatrix} I * \tilde{Q}_\theta & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I * \bar{P}_{\mu,\infty} * \frac{2}{T} \end{bmatrix} \quad (4.58)$$

Initiell kovarians er

$$P^g(0) = \begin{bmatrix} \text{zeros}(3,3) & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I * \bar{P}_\mu(0) \end{bmatrix} \quad (4.59)$$

Diskretisering av spektraltetthetene gjøres slik

$$Q_\theta = \tilde{Q}_\theta * \Delta t \quad (4.60)$$

$$Q_\mu = \frac{\tilde{Q}_\mu T}{2} (1 - e^{(-\frac{2}{T} * \Delta t)}) \quad (4.61)$$

som settes sammen til

$$Q^g = \begin{bmatrix} I * Q_\theta & \text{zeros}(3,3) \\ \text{zeros}(3,3) & I * Q_\mu \end{bmatrix} \quad (4.62)$$

$Q_\theta$  og  $Q_\mu$  beskriver kovariansen til de diskrete hvitstøyelementene

$$\underline{v}^g \sim \mathcal{N}(\underline{0}, Q^g)$$

Etter å ha funnet støymatrisen og systemmatrisen som tidligere vist kan det diskrete systemet skrives

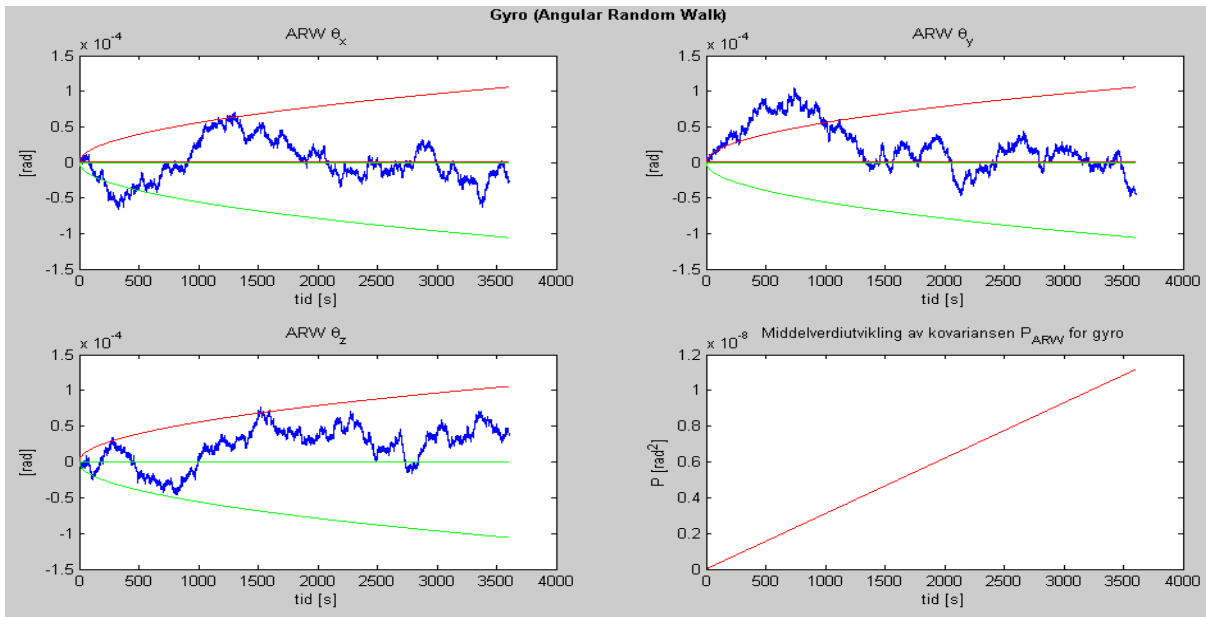
$$\underline{x}_{k+1}^g = \phi^g \underline{x}_k^g + \Gamma^g \underline{v}_k^g \quad (4.63)$$

$$\bar{P}_{k+1}^g = \phi^g \bar{P}_k^g \phi^T + \Gamma^g Q^g \Gamma^{gT} \quad (4.64)$$

Den totale støyen for akselerometeret er da summen av den hvite støyen og den fargede støyen. Men den hvite støyen har nå blitt integrert og gir en random walk på hastighetsnivå. Den totale målefeilen er da gitt ved

$$\delta \tilde{\omega}_k^g = \frac{(\theta_{k+1}^g - \theta_k^g)}{\Delta t} + \underline{\mu}_k^g \quad (4.65)$$

Den hvite støyen i gyroen er vist i figur 13, der den fremstår som en random walk på vinkelnivå, og kovariansen øker lineært.



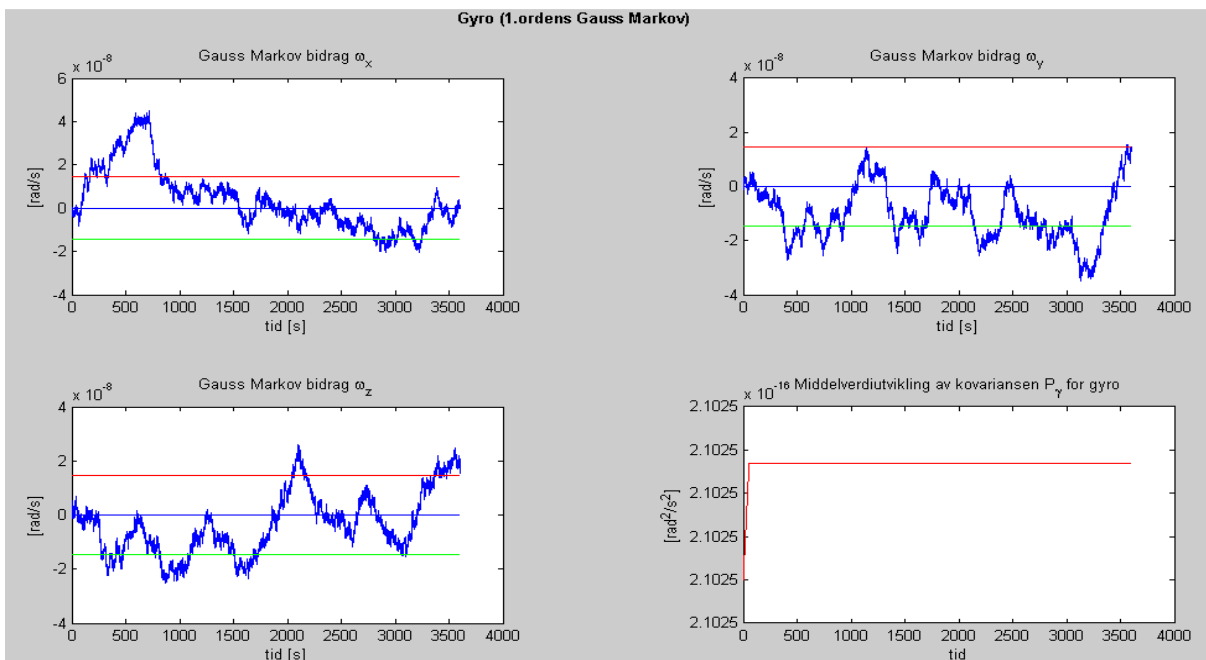
Figur 13: Angular Random Walk i gyroen

Figur 14 viser den fargede støyen som oppstår i gyroskopet. Det kan sees av figuren at middelerdien er 0 og støyen holder seg innenfor

$$\mp \sigma = \mp 1.45 * 10^{-8} \frac{rad}{s}$$

mesteparten av tiden og kovariansen går direkte inn i stasjonærverdi

$$P_0 = P_\infty = \left(1.45 * 10^{-8} \frac{rad}{s}\right)^2 = 2.1025 * 10^{(-16)} \frac{rad^2}{s^2}$$



Figur 14: 1.ordens Gauss Markov støyprosess i gyro

Det er nå mulig å teste TNS systemet med støybelagte målinger for å se hvilken påvirkning støyen har.

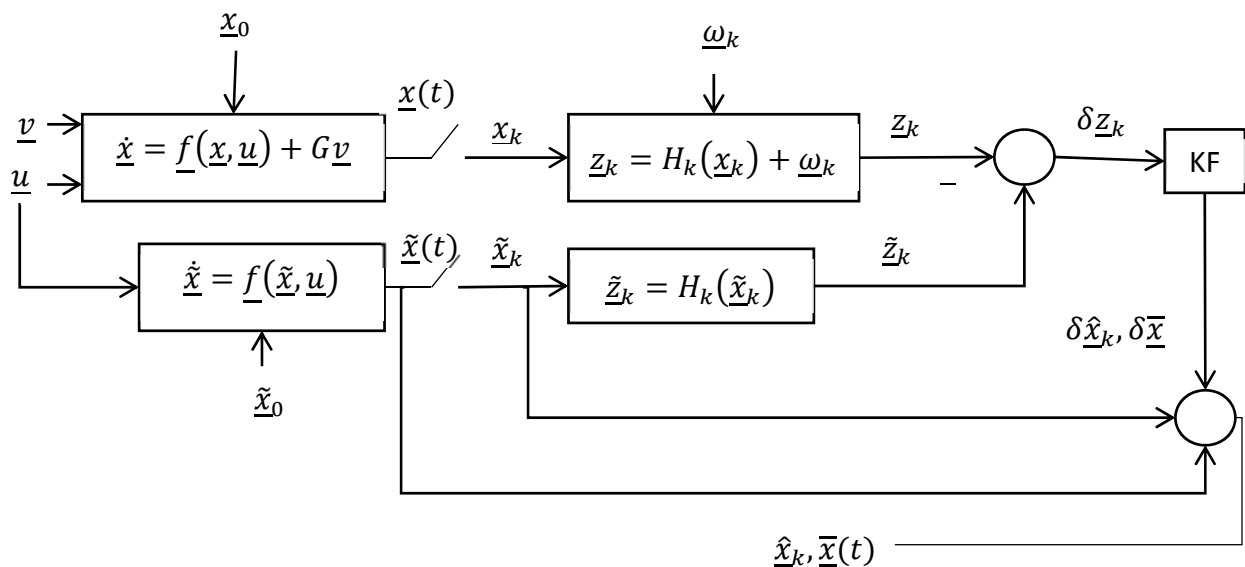
## 4.6 Kalmanfilter

Et kalmanfilter er en optimal tilstandsestimator som på basis av kunnskap om prosessmodell og målinger gir et minimumvarians estimat av tilstanden[10]. Prosessen er beskrevet av ordinære differensiallikninger og målingene beskrevet av algebraiske likninger. Med andre ord er kalmanfilteret en algoritme som implementeres i en datamaskin for å hente ut signaler fra støyfylte målinger og gi et optimalt estimat av tilstanden[8].

Kalmanfilteret kan brukes til å gi et forbedret estimat av tilstanden i fortid, sanntid eller å prediktere tilstanden i fremtid. Metoden ble først introdusert av ungarenen Rudolf Emil Kalman på 1960 tallet. Det var da tenkt at metoden skulle bli brukt på lineære systemer, men et treghetsnavigasjonssystem er ulineært. Selv om kalmanfilteret kan tilpasses til å håndtere ulineære systemer er det ønskelig å bruke et linearisert kalmanfilter. TNS systemet må da lineariseres.

### 4.6.1 Linearisert Kalmanfilter

Utgangspunktet til det lineariserte kalman filteret er et ulineært stokastisk kontinuerlig-diskret systemmodell. Først må man finne en nominell løsning for differensiallikningen for det virkelige systemet, dvs løsning for differensiallikningen uten støy. Deretter lager man en parallell deterministisk (uten støy) modell av systemet som får samme pådraget som det virkelige systemet. Når støyen som kommer inn på det virkelige systemet er satt til null, skal differansen mellom utgangene til det virkelige systemet og det deterministiske systemet være lik null. Dersom vi antar at prosesstøyen er liten, målestøyen er liten og feilen på initialverdier er liten kan man anta at feilen mellom utgangene kan rekkeutvikles og beskrives av 1.ordens ledd i rekkeutviklingen. Dette systemet er en god approksimasjon til et lineært system[10].



Figur 15: Blokkskjema for linearisert kalmanfilter (LKF). LKF har ingen tilbakekobling

Den matematiske modellen for et lineært kontinuerlig-diskret kalmanfilter kan deles inn en tidsoppdatering TO, og en måleoppdatering MO. TO fasen er en prediksjonsfase, der



kalmanfilteret predikterer hva tilstandene vil være, mens MO fasen er en korrigeringsfase der eventuelle prediksjonsfeil blir korrigert.

Tabell 2: Symbolbeskrivelse for kalmanfilteret

| Kontinuerlig Kalmanfilter |  | Diskret Kalmanfilter    |  |
|---------------------------|--|-------------------------|--|
| Symbol                    | Beskrivelse                              | Symbol                  | Beskrivelse                                |
| $\hat{\underline{x}}(t)$  | Estimert tilstandsvektor                 | $\bar{\underline{x}}_k$ | Prediktert (a priori) tilstandsvektor      |
| $\hat{P}(t)$              | Kovariansen til estimert tilstandsvektor | $\hat{\underline{x}}_k$ | Estimert (a posteriori) tilstandsvektor    |
| $K(t)$                    | Kalmanfilterforsterkning                 | $\bar{P}_k$             | Kovariansen til prediktert tilstandsvektor |
|                           |  | $\hat{P}_k$             | Kovariansen til estimert tilstandsvektor   |
|                           |  | $K_k$                   | Kalmanfilterforsterkning                   |

Likningene for tidsoppdatering er som følger

$$\dot{\underline{x}} = \underline{f}(\underline{x}, \underline{u}), \quad \underline{x}(\hat{t}_k) = \underline{x}(\bar{t}_k), \quad t \in [\hat{t}_k, \bar{t}_{k+1}] \quad (4.66)$$

$$\delta \dot{\underline{x}}(t) = F(t)\delta \underline{x}(t), \quad \delta \underline{x}(\hat{t}_k) = \delta \hat{\underline{x}}_k \quad (4.67)$$

$$\bar{\underline{x}}(t) = \hat{\underline{x}}(t) + \delta \underline{x}(t) \quad (4.68)$$

$$\dot{\bar{P}}(t) = F(t)\bar{P}(t) + \bar{P}(t)F^T(t) + G(t)\tilde{Q}(t)G^T(t), \quad \bar{P}(t_k) := \hat{P}_k, \quad \hat{P}_0 \text{ gitt} \quad (4.69)$$

Linkningene for MO er gitt ved

$$\delta \underline{z}_k = \underline{z}_k - \tilde{\underline{z}}_k \quad (4.70)$$

$$\delta \hat{\underline{x}}_k = \delta \bar{\underline{x}}_k + K_k(\delta \underline{z}_k - H_k \delta \bar{\underline{x}}_k) \quad (4.71)$$

$$\hat{\underline{x}}_k = \bar{\underline{x}}_k + \delta \hat{\underline{x}}_k \quad (4.72)$$

$$K_k = \bar{P}_k H_k^T (H_k \bar{P}_k H_k^T + R_k)^{-1}, \quad \bar{P}_k := \bar{P}(t_k) \quad (4.73)$$

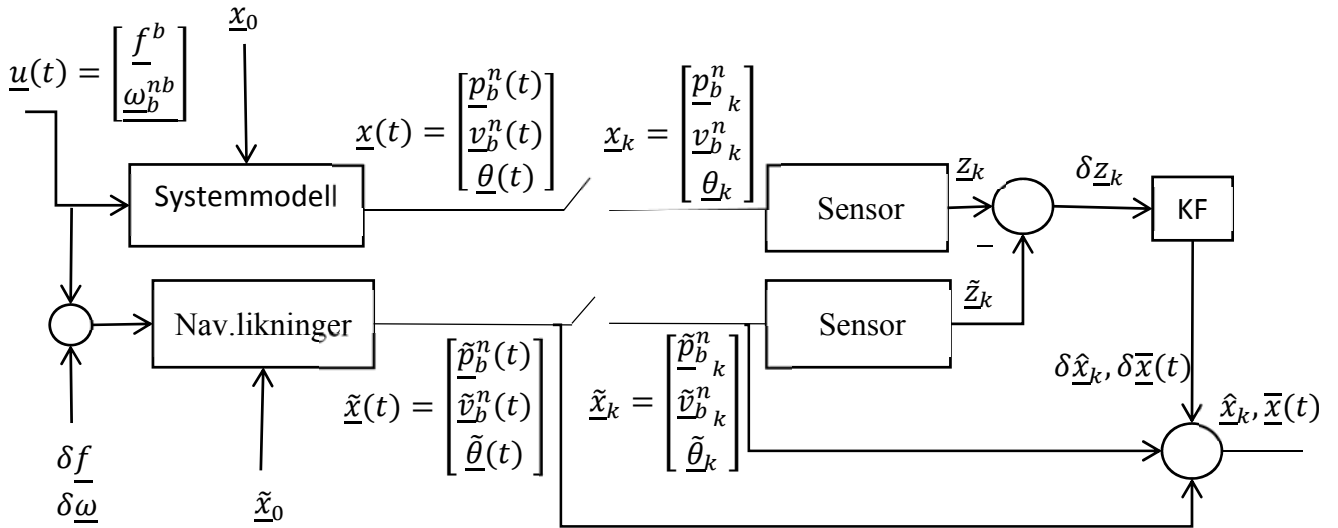
$$\hat{P}_k = (I - K_k H_k) \bar{P}_k, \quad \hat{P}_0 \text{ gitt} \quad (4.74)$$

Modell av TNS med kalmanfilter

Figur 16 viser systemmodellen, navigasjonslikningene og differansen (feilen) mellom modellene som det lineære kalmanfilteret skal estimere ved å finne avviket som gir minst mulig varians for hvert tidsskritt[8]. Feildefinisjonene er gitt i tabellen under.

Tabell 3: Feildefinisjoner brukt i kalmanfilteret[12]

| Feildefinisjoner  |  |   |
|---|--|---|
| $\delta \underline{f} = \underline{f}^b - \tilde{\underline{f}}^b$              | $\delta \underline{x}^n = \underline{x}^n - \tilde{\underline{x}}^n$ | $R_b^n = R(\underline{\epsilon}) - \tilde{R}_b^n$       |
| $\delta \underline{\omega} = \underline{\omega}_b^{nb} - \tilde{\omega}_b^{bn}$ | $\delta \underline{v}^n = \underline{v}^n - \tilde{\underline{v}}^n$ | $R(\underline{\epsilon}) = I + S(\underline{\epsilon})$ |



Figur 16: Utvidet TNS blokkskjema med Kalmanfilter

Navigasjonslikningene er fra før kjent

$$\begin{aligned}\dot{\underline{p}}^n &= \underline{v}^n \\ \dot{\underline{v}}^n &= \underline{\tilde{R}}^n \underline{\tilde{f}}^b - \underline{g}^n \\ \dot{\underline{R}}_b^n &= \underline{\tilde{R}}_b^n S(\underline{\tilde{\omega}}_b^{nb})\end{aligned}$$

Systemmodellen har likningene

$$\begin{aligned}\dot{p}^n &= v^n \\ \dot{v}^n &= \underline{R}^n \underline{f}^b - \underline{g}^n \\ \dot{R}_b^n &= \underline{R}_b^n S(\underline{\omega}_b^{nb})\end{aligned}$$

Her er  $\underline{f}^b$  og  $\underline{\omega}_b^{nb}$  de sanne verdiene som er ukjente. Men fra feildefinisjonen kan man sette inn

$$\underline{f}^b = \underline{\tilde{f}}^b + \underline{\delta f} \quad (4.75)$$

og

$$\underline{\omega}_b^{nb} = \underline{\tilde{\omega}}_b^{nb} + \underline{\delta \omega} \quad (4.76)$$

De nye systemlikningene er

$$\dot{\underline{p}}^n = \underline{v}^n \quad (4.77)$$

$$\dot{\underline{v}}^n = \underline{R}^n (\underline{\tilde{f}}^b + \underline{\delta f}) - \underline{g}^n \quad (4.78)$$

$$\dot{\underline{R}}_b^n = \underline{R}_b^n S(\underline{\tilde{\omega}}_b^{nb} + \underline{\delta \omega}) \quad (4.79)$$

Differansen mellom navigasjonslikningene og de nye systemlikningene er gitt ved

$$\begin{aligned}\delta \underline{\dot{p}} &= \underline{\dot{p}}^n - \dot{\underline{\tilde{p}}}^n = \delta \underline{v} \\ \delta \underline{\dot{v}} &= \underline{\dot{v}}^n - \dot{\underline{\tilde{v}}}^n = -S(\underline{\tilde{R}}_b^n \underline{\tilde{f}}) \underline{\epsilon} + \underline{\tilde{R}}_b^n \underline{\delta f} \\ \underline{\dot{\epsilon}} &= \underline{\tilde{R}}_b^n \underline{\delta \omega}\end{aligned}$$

Feillikningene til kalmanfileret er da

$$\delta \underline{\dot{p}} = \underline{\dot{p}}^n - \underline{\dot{p}}^n = \delta \underline{v} \quad (4.80)$$

$$\delta \underline{\dot{v}} = \underline{\dot{v}}^n - \underline{\dot{v}}^n = -S(\tilde{R}_b^n \underline{\tilde{f}}) \underline{\epsilon} + \tilde{R}_b^n \delta \underline{f} \quad (4.81)$$

$$\underline{\dot{\epsilon}} = \tilde{R}_b^n \delta \underline{\omega} \quad (4.82)$$

Kalmanfiltermodellen kan settes opp slik

$$\delta \underline{\dot{x}} = F(t) \delta \underline{x} + G \underline{v} \quad (4.83)$$

$$\delta \underline{z}_k = H_k \delta \underline{x} + \underline{\omega}_k \quad (4.84)$$

Feilbidraget fra sensorene blir lagt til i tilstandsvektoren slik at

$$\delta \underline{\dot{x}} = [\delta \underline{\dot{p}}^n \quad \delta \underline{\dot{v}}^n \quad \underline{\dot{\epsilon}} \quad \underline{\dot{c}}^a \quad \underline{\dot{\mu}}^a \quad \underline{\dot{c}}^g \quad \underline{\dot{\mu}}^g]$$

Da blir systemmatrisen

$$F = \begin{bmatrix} 0 & I & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -S(\tilde{R}_b^n \underline{\tilde{f}}^b) & \tilde{R}_b^n & \tilde{R}_b^n & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \tilde{R}_b^n & \tilde{R}_b^n \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{T^a} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T^g} \end{bmatrix}$$

Støyvektoren blir

$$[0 \quad 0 \quad 0 \quad \underline{v}^a \quad \underline{v}_\mu^a \quad \underline{v}^g \quad \underline{v}_\mu^g]$$

Og støymatrisen G blir slik

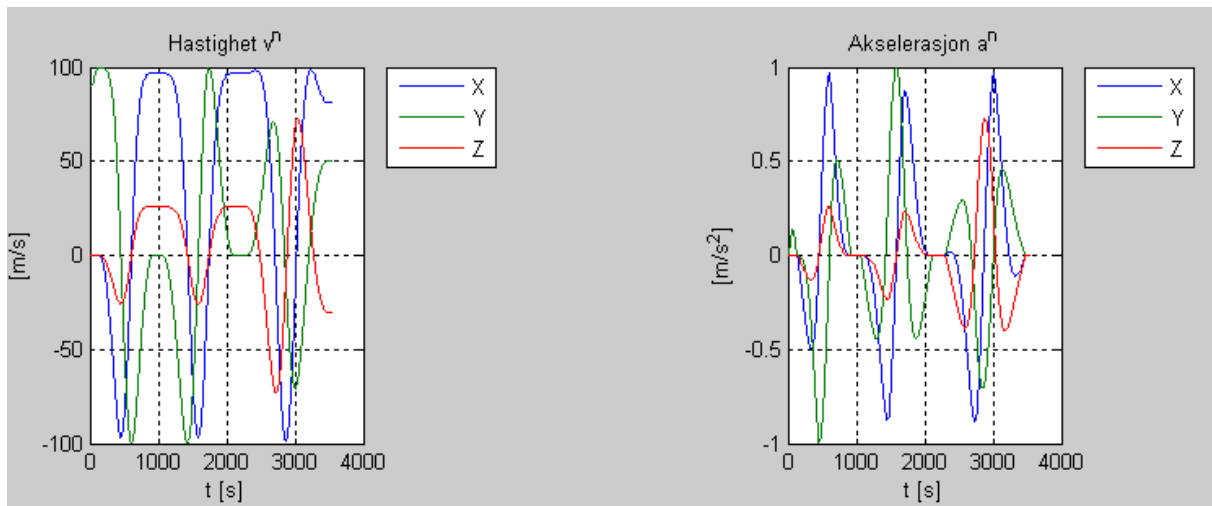
$$G = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \tilde{R}_b^n & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \tilde{R}_b^n & 0 \\ 0 & 0 & 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I \end{bmatrix}$$

## 5 Simuleringsresultater

Alle simuleringer er utført med en frekvens på 100 Hz og en  $\Delta t=0.000001$  (likning 3.57). Aller først vises resultatet for banegeneratoren som genererer sann banedata fra flyet. Deretter simuleres TNS systemet uten støy med integrasjonsrutinene Euler og Heuns. Så kommer simuleringsresultatene som viser hvordan TNS systemet reagerer på kun hvit støy, kun farget støy og til slutt legges det på både hvit og farget støy. Alle simuleringene er gjennomført uten filtrering av støyen.

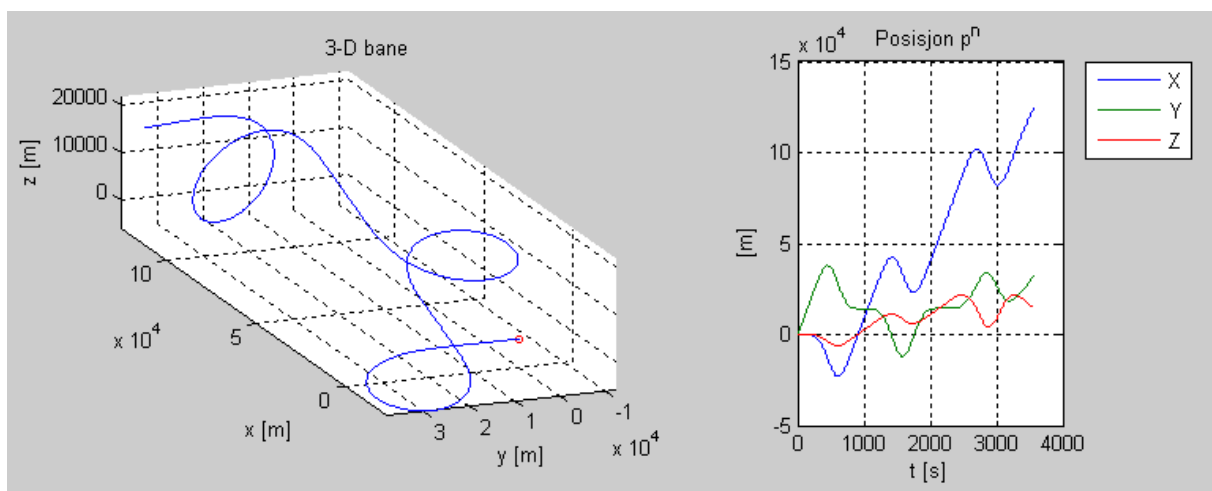
### 5.1 Banegenerator

I tråd med målsettingen er banegeneratoren utvidet til å simulere en 3-D bane. Banegeneratoren simulerer et fly som flyr langs en bane i rommet, representert i  $\mathcal{F}^n$ . Initialbetingelsene, akselerasjonen, farten og stillingen til flyet har generert resultatene for banegeneratoren vist i figurene under. Posisjonen til flyet ved  $t_0$  er i origo i  $\{n\}$ -rammen. Simuleringen kjører i 3600 sekunder.



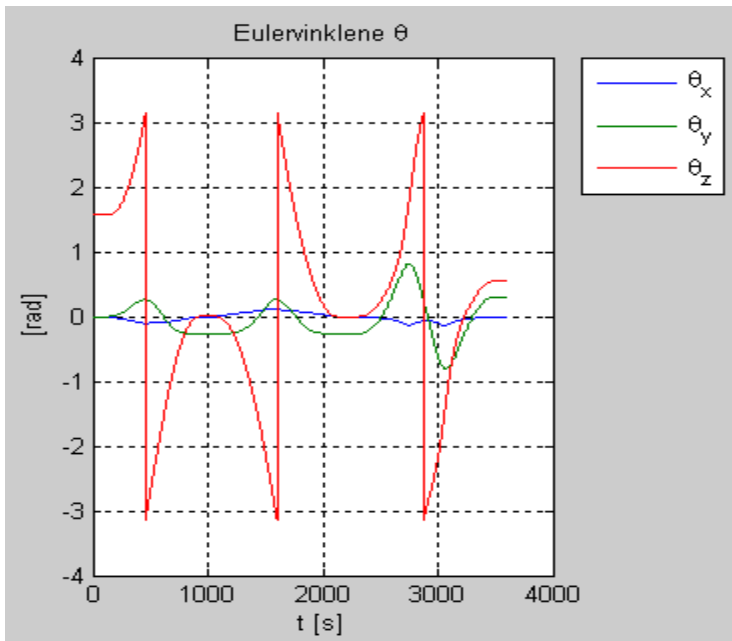
Figur 17: Akselerasjonen og hastighet fra banegenerator

Den tre dimensjonale banen er vist i figur 18 sammen med posisjonen langs hver av aksene som funksjon av tiden



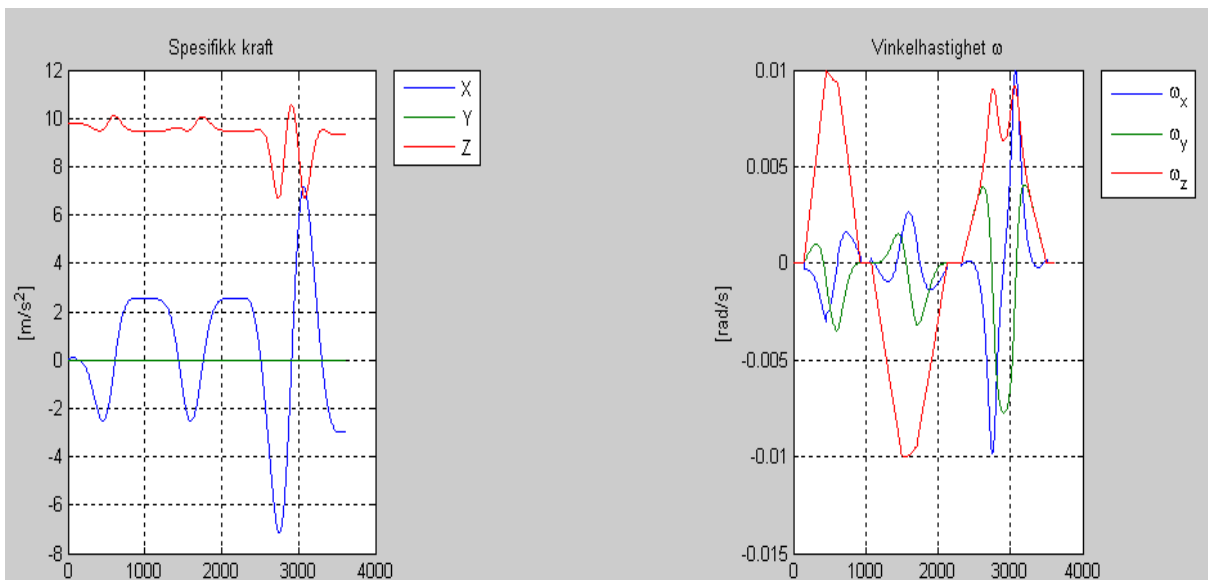
Figur 18: 3-D banen og posisjonen til flyet. Rød sirkel angir startposisjon

Banegeneratoren beregner også Eulervinklene  $\underline{\theta}$ , eller stillingen, til flyet slik figur 19 viser. Figuren viser at flyet roterer mest om z-aksen ( $\vec{n}_3$ -aksen i {n}-systemet) som man også kan se av 3-D banen.



Figur 19: Eulervinklene beregnet av banegeneratoren

De støyfrie beregningene av den spesifikke kraften  $\underline{f}^b$  og vinkelhastigheten  $\underline{\omega}^{nb}$  er vist i figur 20. TNS systemet bruker disse beregningene, med og uten støy, i navigasjonslikningene.



Figur 20: Spesifikk kraft og vinkelhastighet fra banegeneratoren

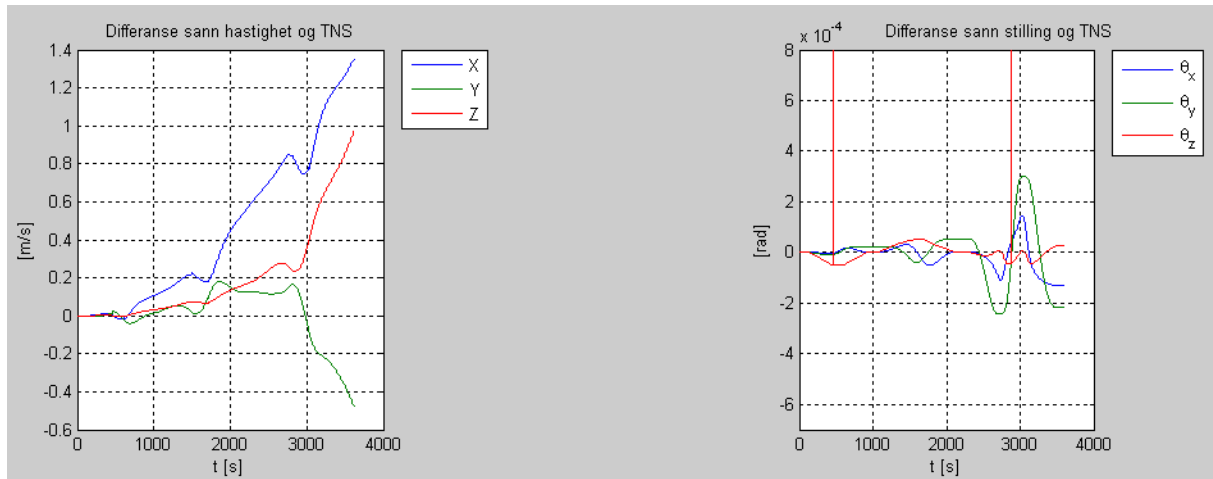
Dette er altså sanne verdier fra banegeneratoren. Når den spesifikke kraften og vinkelhastigheten fra banegeneratoren blir gitt som et pådrag til TNS systemet ønsker man at banen som som TNS systemet beregner er mest mulig lik denne.

## 5.2 Simulering av 3-D banen uten støy

Treghtsnavigasjonssystemet ble tidligere verifisert med test av en sirkelbane uten støy. Testen viste at Eulers metode bommet på posisjonen med ca 195 m etter 3600 sekunder. Heuns metode derimot bommet med ca 1.6 m. Sammenlikning av disse integrasjonsrutinene på sirkelbanen og 3-D banen kan gjøres i det støyfrie tilfellet.

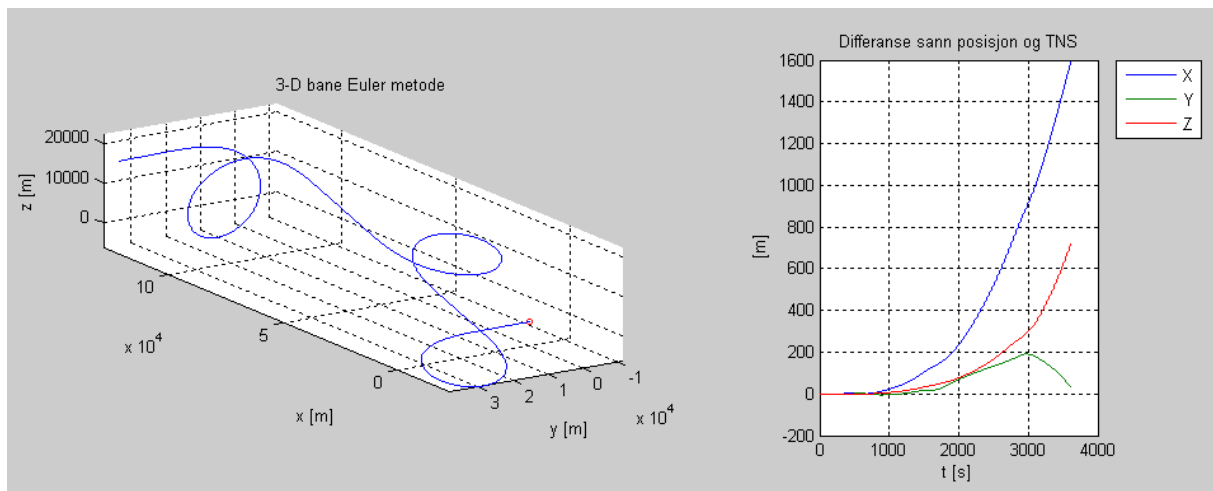
### 5.2.1 Resultater Eulers metode uten støy

Etter å ha initialisert starttilstanden til TNS systemet til samme tilstand som banegeneratoren brukes Eulers metode for å integrere TNS systemet



Figur 21: Differanse Banegenerator og TNS, Eulers metode

Figur 21 viser at differansen mellom sann og beregnet hastighet øker med tiden og bidrar følgelig til feil i posisjonen. Differansen mellom stillingen er mindre relativt til differansen i hastigheten. Spikerne som dukker opp kan forklares med at de sanne og beregnede eulervinklene er forskjøvet i tid i forhold til hverandre, slik at når den ene har gått  $2\pi$  radianer og er tilbake til utgangspunktet, har den andre gått ett tidskritt mindre og differansen mellom disse blir  $2\pi$ , som er verdien til spikerne.

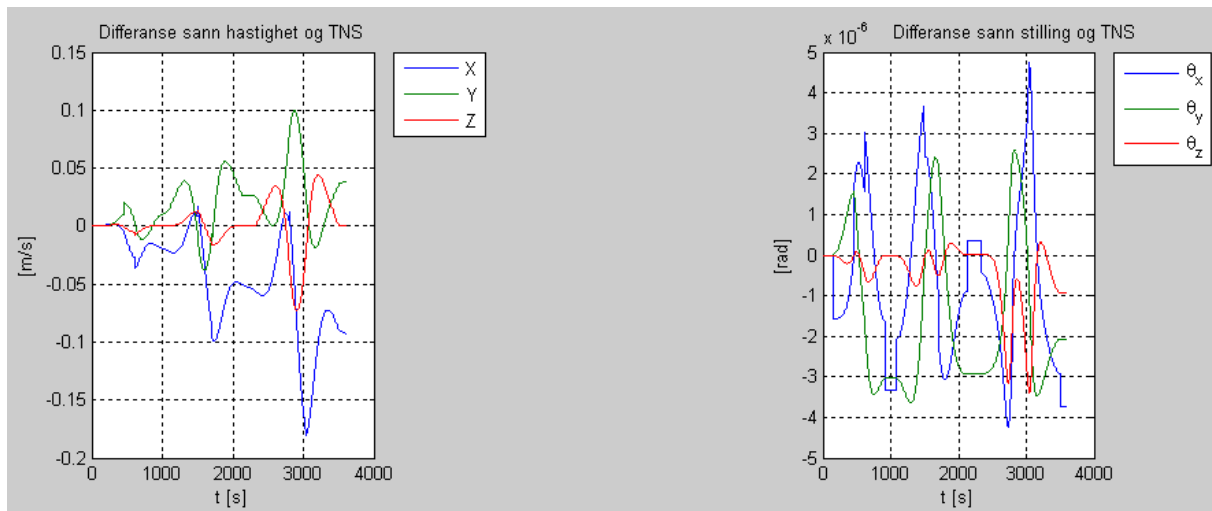


Figur 22: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS

I figur 22 kan man se at banen gitt av navigasjonssystemet med Euler integrasjon virker tilsynelatende lik den sanne banen. Men differansen viser at etter en time er feilen i posisjonen ca 1.78 km.

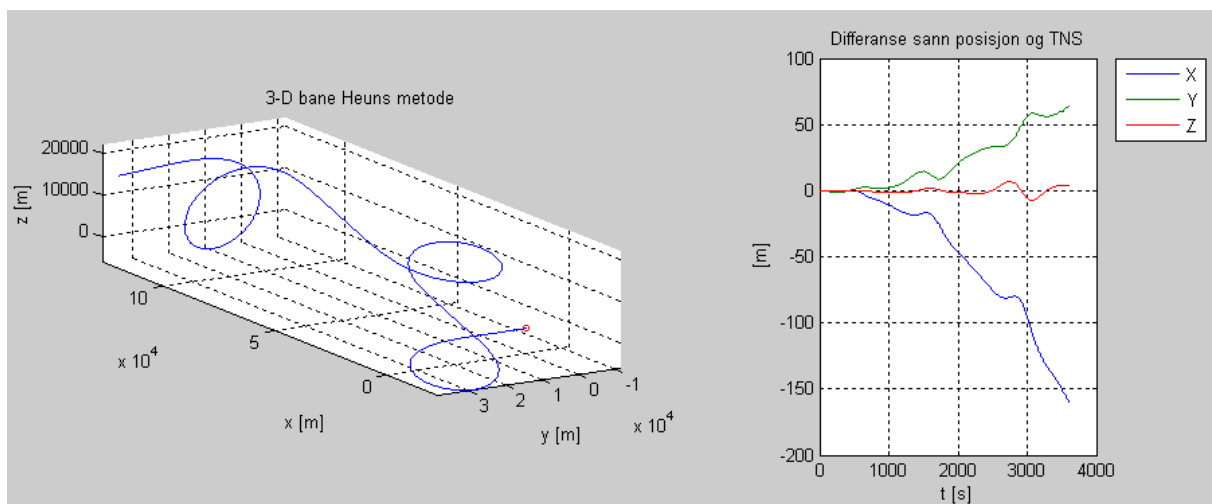
### 5.2.2 Resultater Heuns metode uten støy

Sammenlignet med Euler er integrasjonrutinen til Heuns mer nøyaktig. Som man kan se av figur 23 er feilen i hastigheten og stillingen mindre med Heuns metode.



Figur 23: Differanse Banegenerator og TNS, Heuns metode

Feilen i posisjon er følgelig mindre etter endt simulering og ligger på ca 172 m. Dette viser at i det støyfrie tilfellet er Heuns integrasjonrutine mer nøyaktig enn Heuns på 3-D banen.



Figur 24: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS

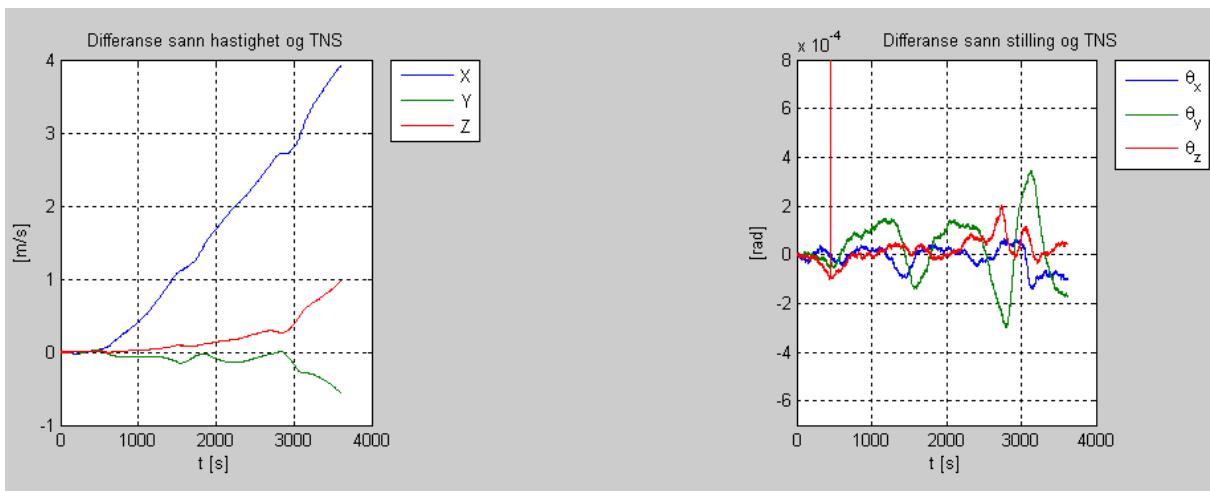
Resultatet for den støyfrie delen av 3-D banen er dermed presentert. Neste seksjon viser hvordan støyprosessene påvirker TNS systemet og de numeriske integrasjonsrutinene.

### 5.3 Simulering av 3-D banen med hvit støy

Ved å legge til støy blir simuleringen mer realistisk og resultatene under viser hvordan TNS systemet reagerer på støyfylte målinger uten algoritmer som tar høyde for nærvær av støy. I denne seksjonen legges det kun hvit støy på akselerometer og gyro målingene

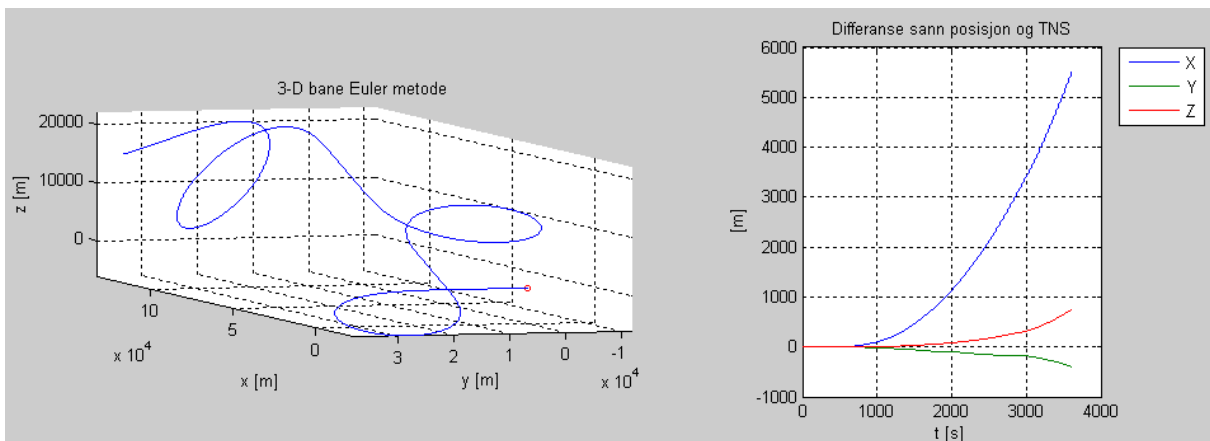
#### 5.3.1 Resultater Eulers metode med hvit støy

Eulers metode med kun hvit støy lagt på målingene fører til enda større feil i hastigheten. Figur 25 viser at TNS systemet feilaktig beregner hastigheten til å øke i x-retningen. Støyen fører også til større feil i stillingen, der spikeren har samme forklaring som tidligere og har verdien  $2\pi$ .



Figur 25: Differanse Banegenerator og TNS, Eulers metode med hvit støy

Feilen i hastigheten og stillingen fører til at feilen i posisjonen øker. Etter å ha simulert i en time bommer TNS systemet på posisjonen med ca 5.6 km. Av figurene kan man se at den største feilen blir gjort i x-retning.

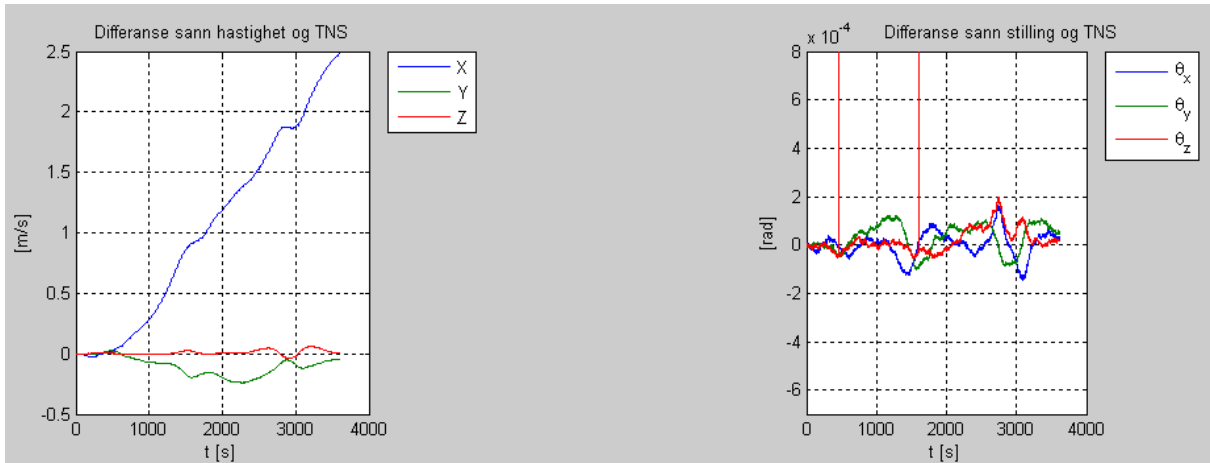


Figur 26: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS



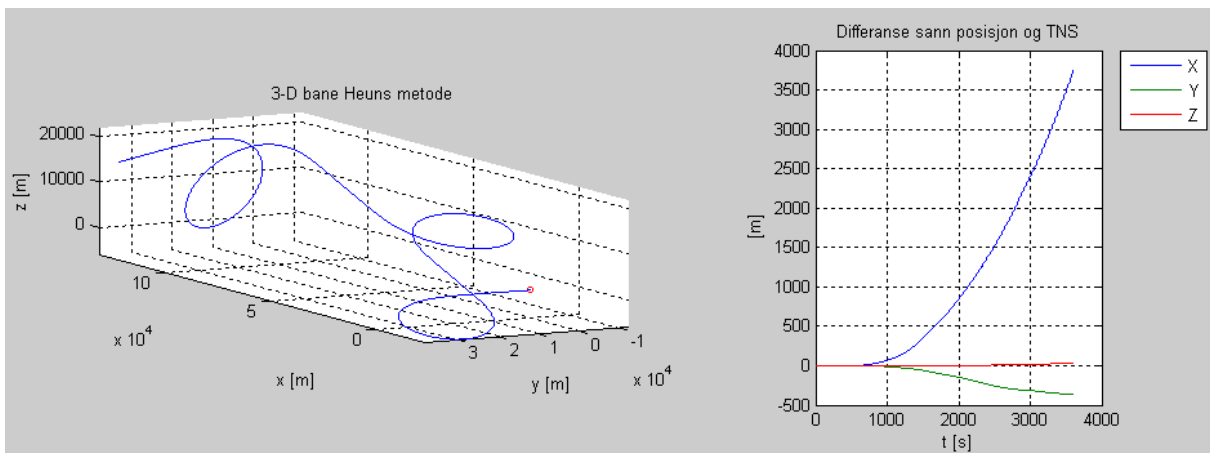
### 5.3.2 Resultater Heuns metode med hvit støy

Selv om Heuns metode ga et langt bedre resultat i det støyfrie tilfellet, viser det seg at også denne metoden er svært påvirket av støy. Den hvite støyen påvirker Heuns rutinen og ut ifra figur 27 øker feilen i x-retningen. Differansen i stillingen øker også sammenlignet med det støyfrie tilfellet.



Figur 27: Differanse Banegenerator og TNS, Heuns metode

Feilen i hastigheten i, og spesielt i x-retningen, og i stillingen fører posisjonen vekk fra den sanne banen. Av TNS beregningen vist i figur 28 ser man at etter en time med simulering bommer TNS systemet med ca 3.7 km.



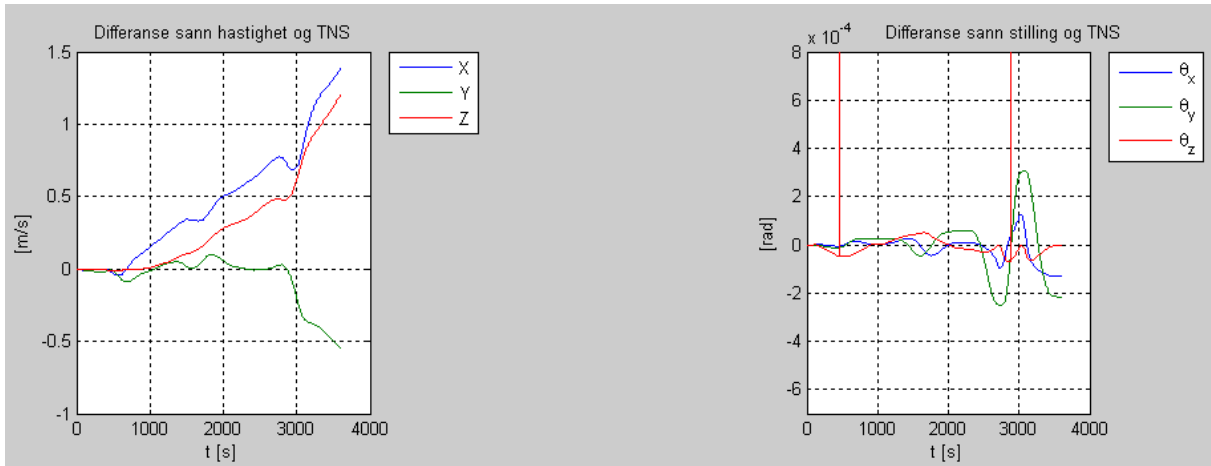
Figur 28: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS

## 5.4 Simulering av 3-D bane med farget støy

I denne seksjonen legges det kun til farget støy for å se hvordan TNS systemet reagerer på sensormålingene.

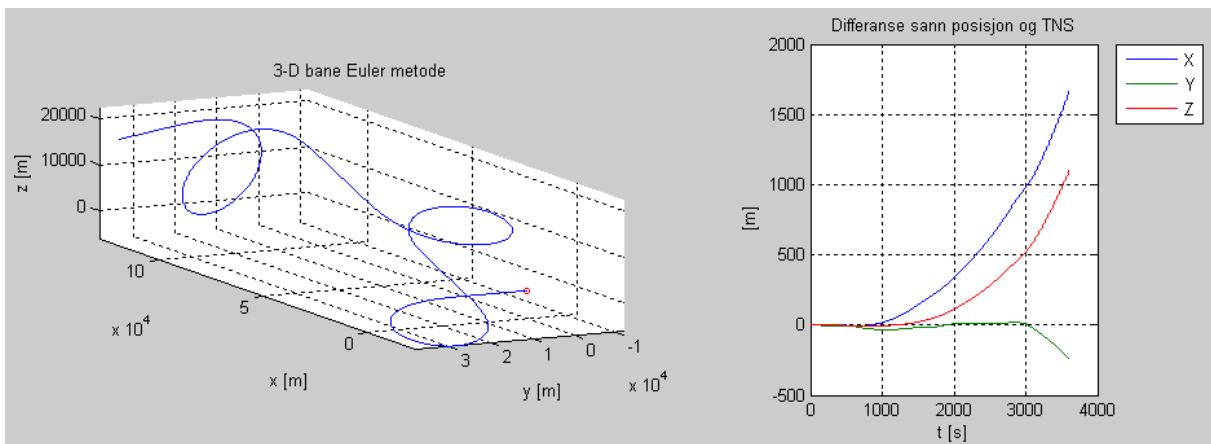
### 5.4.1 Resultater Eulers metode med Farget støy

Eulers metode med farget støy gir at feilen i stillingen varierer som vist i figur 29. Spikeren kan forklares med differansen som oppstår pga tidsforskyvningen som tidligere. Feilen i hastigheten utvikler seg med tiden som vist i figur 29.



Figur 29: Differanse Banegenerator og TNS, Eulers metode

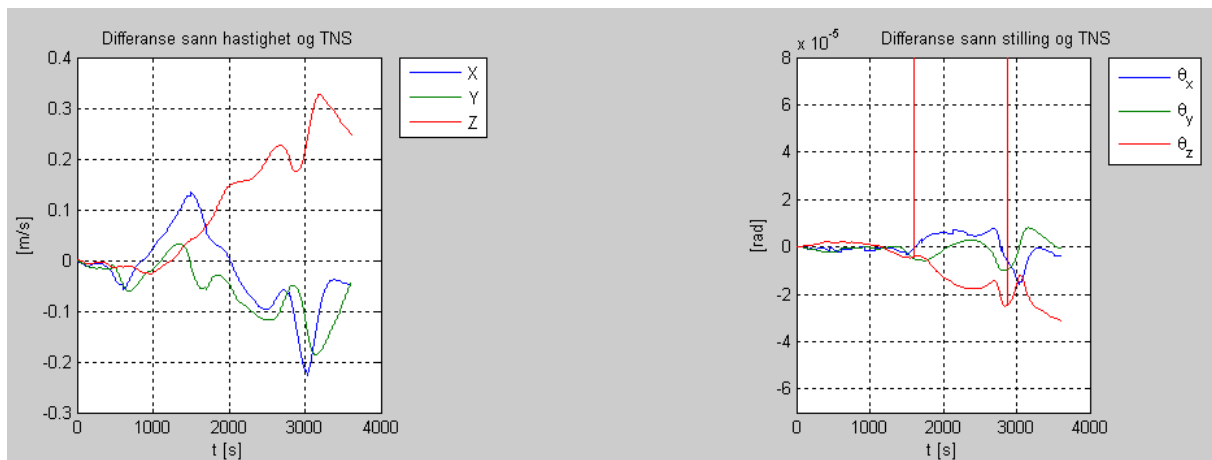
Ettersom feilen i hastigheten forplanter seg blir feilen i posisjon også større og større. I denne simuleringen er den største feilen i hastigheten langs x og z-retning, dermed blir også posisjonen mest påvirket i den retningen. Feilen i posisjon etter en time er ca 2 km.



Figur 30: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS

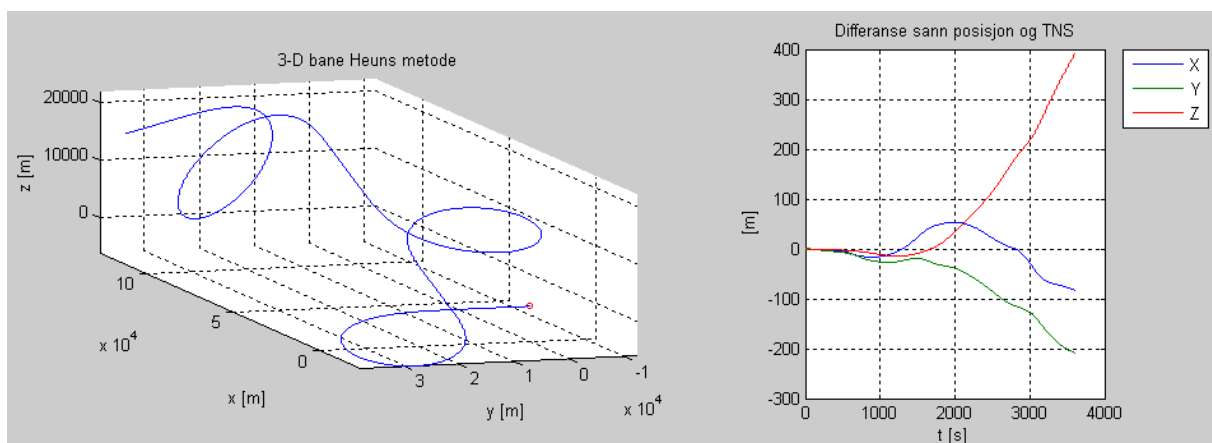
### 5.4.2 Resultater Heuns metode med Farget støy

Den fargede støyen driver de beregnede hastighetskomponentene og stillingen vekk fra sann verdi som figur 31 viser. Spikerne i stillingen kan forklares som tidligere.



Figur 31: Differanse Banegenerator og TNS, Heuns metode

Den økte feilen i hastigheten får TNS systemet til å feilaktig beregne posisjonen til å være 452 meter unna den sanne. Den største feilen gjøres i x-retning som vist i figurene.



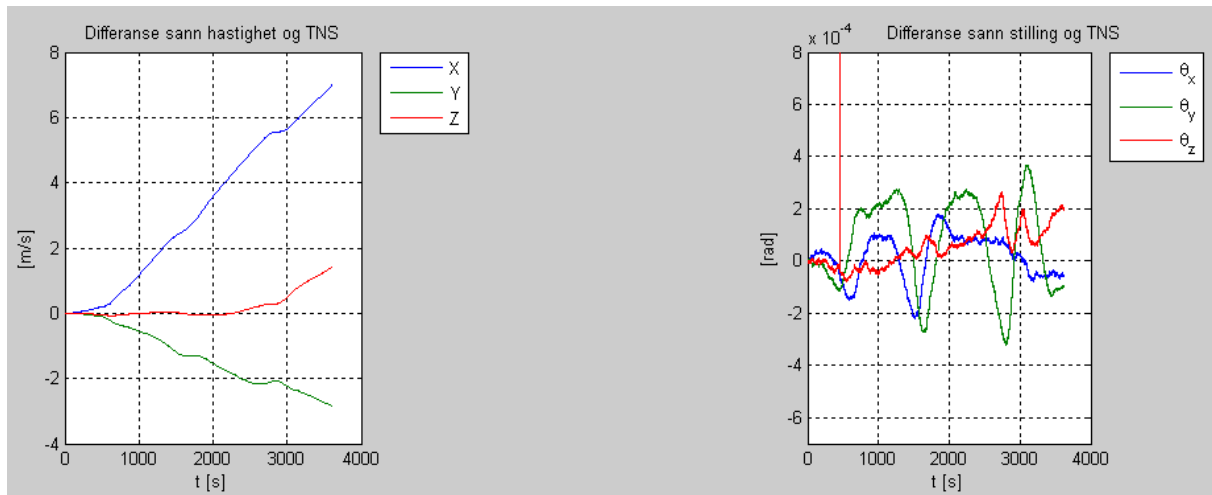
Figur 32: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS

## 5.5 Resultater TNS med hvit og farget støy

Selv om støykildene i en sensor vil være flere er denne oppgaven begrenset til hvit og farget støy. Den hvite og fargede støyens virkning på TNS systemet presenteres i denne seksjonen.

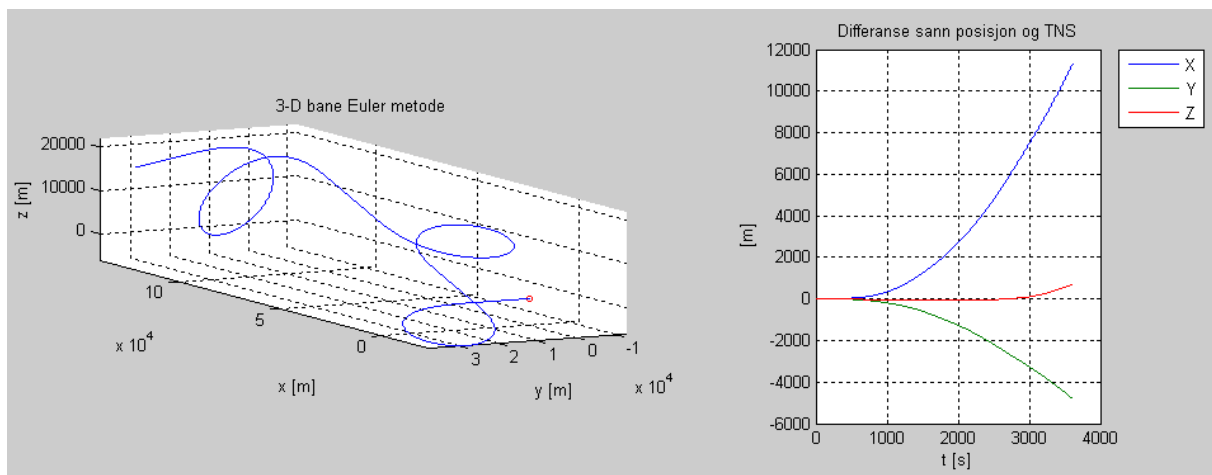
### 5.5.1 Eulers metode med hvit og farget støy

Resultatet med hvit og farget støy viser at feilen i farten og stillingen har økt ytterligere i forhold til det støyfrie tilfellet. Spikeren som dukker opp i figuren til stillingen kan forklares som tidligere.



Figur 33: Differanse Banegenerator og TNS, Eulers metode

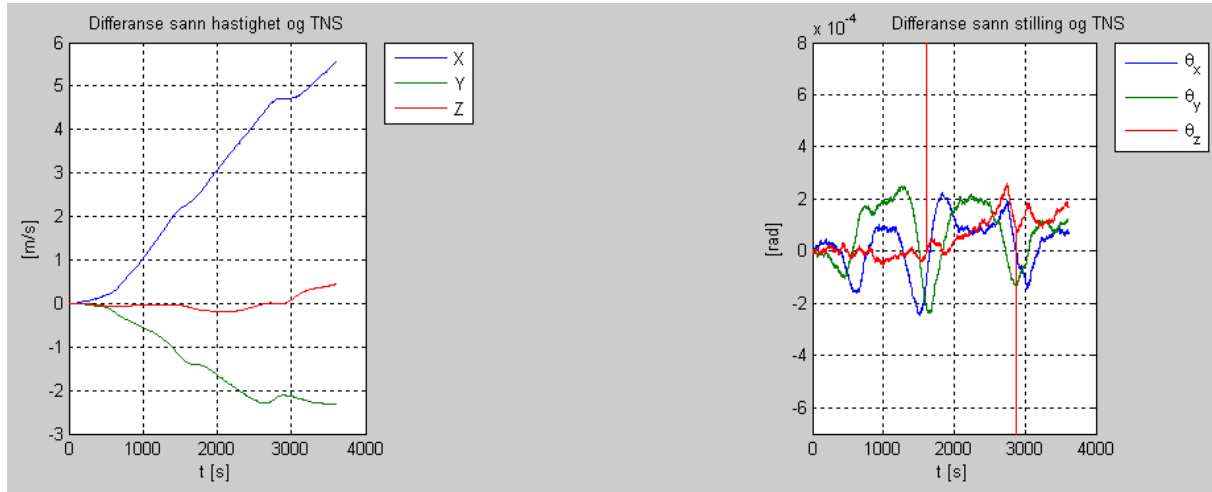
Denne forskjellen mellom beregnet og sann hastighet og stilling fører til at navigasjonssystemet bommer på posisjonen til flyet med 12,3 km etter en time.



Figur 34: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS

### 5.5.2 Heuns metode med hvit og farget støy

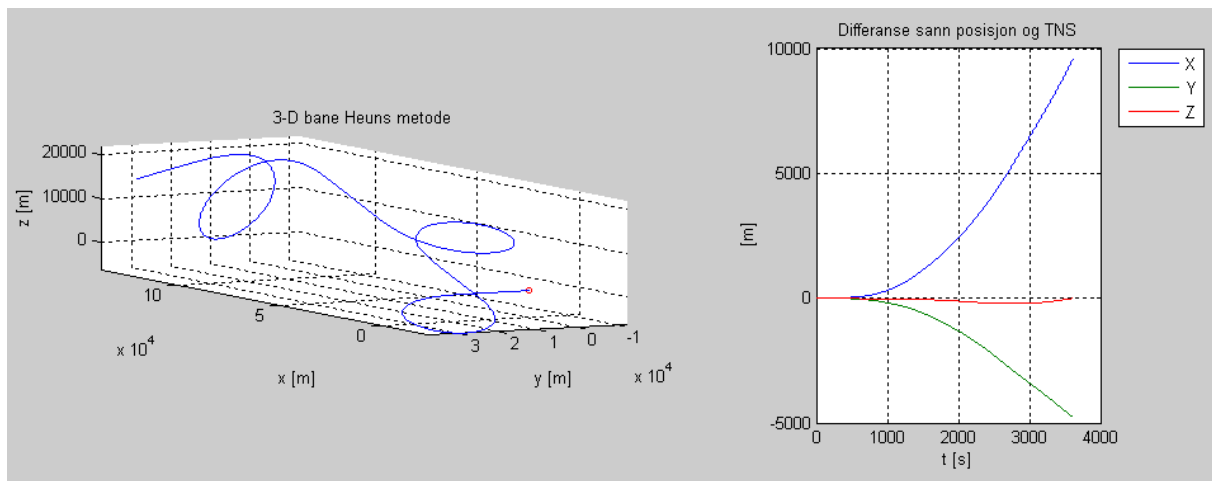
Figur 35 viser at også hastigheten beregnet med Heuns metode har en relativt stor differanse, spesielt i x-retning. Spikeren som oppstår i deifferansen til Eulervinklene skyldes igjen tidsforskyvningen og de der begge lik  $2\pi$ .



Figur 35: Differanse Banegenerator og TNS, Heuns metode

Feilen TNS systemet gjør på hastigheten i x-retning driver den beregnede posisjonen langt unna den sanne posisjonen. Etter en time har TNS systemet feilaktig beregnet posisjonen til å være 10.7 km.

Det er ganske tydelig at i alle tilfeller har støyen hatt en negativ påvirkning på TNS systemets evne til å beregne seg fram til riktig bane.



Figur 36: 3-D banen og differansen i posisjon mellom banegeneratoren og TNS

## 6 Diskusjon

Det er som nevnt ønskelig at den støyfrie simuleringen av TNS systemet skal gi en bane som er mest mulig lik banegeneratoren. Men de numeriske integrasjonsrutinene Euler og Heuns metode gir også opphav til feil og en sammenlikning er på sin plass.

Etter en times simulering av en sirkelbane i planet var feilen mellom Eulers metode og den deterministiske banen 195 m, mens Huens bommet med kun 1.6 m. Feilen til Euler er altså større med omtrent en faktor 100.

For 3-D banen ga integrasjonsrutinen til Huens en feil på 172 m, derimot var feilen til Euler ca 1.72 km. Altså, her er feilen til Euler større med omtrent en faktor 10. Tabellen under viser feilen integrasjonsrutinene gjør ved forskjellige frekvenser uten støy.

Tabell 4: Sammenlikning av feilen til integrasjonsrutinene med forskjellige frekvenser

| Banetype/Integrasjonsrutine | Frekvens |        |         |
|-----------------------------|----------|--------|---------|
|                             | 10 Hz    | 100 Hz | 1000 Hz |
| Sirkel/Euler                | 5046 m   | 195 m  | 17 m    |
| Sirkel/Heuns                | 6.63 m   | 1.6 m  | 1.6 m   |
| 3-D/Euler                   | 18752 m  | 1720 m | 175 m   |
| 3-D/Heuns                   | 2473 m   | 172 m  | 17 m    |

Dette viser at integrasjonsrutinen til Euler har problemer med å følge sirkelbaner sammenlignet med Heuns. Skal man få ned feilen med Eulers metode må frekvensen økes, følgelig mye lengre simuleringstid. Tabell 4 viser at Eulers metode ved 1000 Hz fremdeles gir en større feil enn Heuns metode ved 100 Hz. En annen observasjon er at feilen til Heuns rutinen fremstår som uendret fra 100 Hz til 1000 Hz.

3-D banen har færre sirkelbaner og flere rette strekninger. For denne banetypen er forskjellen i feilen mindre mellom Euler og Heuns, selv om Huens rutinen fremdeles er nærmere sann posisjon.

I tillegg til integrasjonsfeilen kommer feilen som skyldes støy i sensormålingene. Uten et filter vil TNS systemet drives ytterligere vekk fra den sanne verdien som resultatene i kapittel fem viste. Et kalmanfilter kan redusere støyens påvirkning av TNS systemet ved å redusere variansen til feilen og ved hjelp av en kovariansanalyse er det mulig å isolere den hvite og fargede støyen for å undersøke feilbidraget fra hver støyprosess. Prosjektet fikk dessverre ikke tid til å implementere disse forbedringene i oppgaven.

Generering av banegeneratoren for 3-D banen viste seg å bli en tidkrevende prosess i Matlab. Det var to mulige fremgangsmåter, sette sammen banen av et stort antall banelementer med kort varighet, eller sette sammen banen av færre banelementer med lengre varighet. Den siste fremgangsmåten ble valgt for at koden skulle bli mer oversiktlig, men dette førte til at simuleringen med 100 Hz tok ca 25 minutter som er den mest tidkrevende delen av prosesseringen. Det viser seg at tiden for 3-D banen ikke blir nøyaktig 3600 sekunder da tiden i banelementene har desimaler. For 10 Hz legges det til 140 tidsskritt og ved 100 Hz legges det til 156 tidsskritt.

## 7 Konklusjon

Oppgaven har gått ut på å lage en banegenerator i 3-D og beregne vinkelhastighet og spesifikk kraft med og uten støy. Disse verdiene skal så testes i et treghetsnavigasjonssystem for flat ikke-roterende jord. For å verifisere at dataene fra banegeneratoren var riktige og at integrasjonsrutinene i TNS systemet var riktig satt opp ble en sirkelbane simulert i 3600 sekunder uten støy for å se på integrasjonsfeilen. Da integrasjonsfeilen kom ned på det nivået man forventet, ble 3-D banen testet uten og med støy.

Ut ifra resultatene kan man konkludere med at Heuns integrasjonsmetode er bedre egnet enn Eulers metode til å beregne både sirkel og 3-D banen. For begge banetyperne var feilen til Euler større ved 1000 Hz enn Heuns ved 100 Hz.

Men med en gang støyen kommer inn i bildet blir driften i TNS systemet stor. Det er klart at dersom disse metodene skal kunne fungere i praksis må man ta høyde for støyen og implementere en estimeringsmetode som et kalmanfilter som tar i bruk eksterne målinger av flyets posisjon. Ved å gjøre dette er det mulig å sammenlikne sann og beregnet posisjon og dermed si noe om hvordan støyen driver TNS systemet og kalmanfilteret kan da prediktere den optimale tilstanden for flyet. Måleoppdateringene kan også tjene til å modellere feilen integrasjonsrutinene gjør som også vil oppleves som støy for systemet.

Det kan også være interessant å undersøke støyprosessenes feilbidrag til systemet vha en kovariansanalyse.

En annen konklusjon er at generering av banegeneratoren kan ta lang tid. Avhengig av steplengden man velger og banen man ønsker å generere kan tiden fort legge en begrensning på banen man ønsker seg. Det kan være lurt å lagre og laste banegeneratordata når har funnet en bane man er fornøyd med.

## 8 Videre arbeid

Videre arbeid med denne oppgaven kan være å finne en analytisk løsning for  $\dot{R}$  i likning 3.57 og dermed få en analytisk løsning av vinkelhastigheten fra banegeneratoren. Det kan også være interessant å inkludere flere støyprosesser som feks bias for å gjøre støyen mer realistisk. For å gjøre navigasjonssystemet komplett er kan det legges på et lineært kalmanfilter og utføre en kovariansanalyse for å bedømme støyens påvirkning av TNS systemet. Simulering av programmet i Matlab med 100 Hz tar ca 25 minutter og med en simulering med 1000 Hz tar nesten 2 timer. Dersom det er mulig å redusere simuleringstiden eller å simulere med flere frekvenser samtidig ved å bruke parallellprogrammering kan man spare tid i lengden.



## Referanser

- [1] O.Hallingstad(2012). Matematisk modellering av dynamiske systemer. Forelesning
- [2] C.T.J Dodson, T.Poston(1991). *Tensor geometry*. Springer
- [3] Hallseth, Haugan,Hjelmen, Isnes (2002). *Klassisk mekanikk*.NKI
- [4] John J. Craig(2005). *Introduction to Robotics Mechanics and Control* Third edition. Pearson Prentice Hall
- [5] O.Hallingstad(2011). *TNS for flat ikke roterende jord*. Notat
- [6] [http://en.wikipedia.org/wiki/Euler\\_method](http://en.wikipedia.org/wiki/Euler_method).
- [7] [http://en.wikipedia.org/wiki/Heun%27s\\_method](http://en.wikipedia.org/wiki/Heun%27s_method).
- [8] Halvorsen, Hans Petter(2012). *Kalmanfilter*. Forelesning HiT
- [9] O. Hallingstad(2013). Matematisk modellering for dynamiske systemer. Forelesning UNIK
- [10] O. Hallingstad(2012). Stokastiske systemer. Forelesning UNIK
- [11] Hossain, Alya (2014). *Analyse av banegenerator*. Masteroppgave fysisk institutt, UiO.
- [12] O. Hallingstad(2004). *Eksempler på TNS modeller*.
- [13] Gelb, Arthur(1974). *Applied Optimal Estimation*.
- [14] O.Hallingstad(2014). *Banegenerator*. Notat
- [15]Brown, Hwang(2012) *Introduction to Random signals and Applied Kalman filtering*. Wiley

## Vedlegg

### Appendiks A: Pseudokoder

#### Appendiks A-1: Rett linje med akselerasjon

$T = 143$ ;  $t = 0$ :  $dt: T$ ;  $V_0 = 90$ ;  $V = 100$ ;  $dt = 0.01$ ;

$\underline{p}^a(t), \underline{v}^a(t), \underline{a}^a(t) = RBA(t, T, V_0, V)$ ;

function  $\underline{p}^a, \underline{v}^a, \underline{a}^a = RBA(t, T, V_0, V)$

$A = \frac{V_1 - V_0}{T}$ ;  $\omega = \frac{2\pi}{T}$

$$\underline{p}^a(t) = \begin{bmatrix} V_0 t + \frac{1}{2} A t^2 - \frac{A}{\omega^2} (1 - \cos(\omega t)) \\ 0 \\ 0 \end{bmatrix};$$

$$\underline{v}^a(t) = \begin{bmatrix} V_0 + A(t - \frac{1}{\omega} \sin(\omega t)) \\ 0 \\ 0 \end{bmatrix};$$

$$\underline{a}^a(t) = \begin{bmatrix} A(1 - \cos(\omega t)) \\ 0 \\ 0 \end{bmatrix};$$

end

#### Appendiks A-2: Eulerspiral

$T = 314.16$ ;  $t = 0$ :  $dt: T$ ;  $A = 1$ ;  $V = 100$ ;  $dt = 0.01$ ;

$\underline{p}^a(t), \underline{v}^a(t), \underline{a}^a(t) = ESP(t, T, V, A)$ ;

$\theta(T) = \frac{AT}{2V}$ ;

function  $\underline{p}^a(t), \underline{v}^a(t), \underline{a}^a(t) = ESP(t, T, V, A)$

$$\theta(t) = \frac{At^2}{2VT}; \quad \Omega = \sqrt{\frac{A}{\pi VT}};$$

$$\underline{p}^a(t) = \frac{V}{\Omega} \begin{bmatrix} C(\Omega t) \\ S(\Omega t) \\ 0 \end{bmatrix};$$

$$\underline{v}^a(t) = \begin{bmatrix} \cos(\theta(t)) \\ \sin(\theta(t)) \\ 0 \end{bmatrix};$$

$$\underline{a}^a(t) = \begin{bmatrix} -\sin(\theta(t)) \\ \cos(\theta(t)) \\ 0 \end{bmatrix};$$

### Appendiks A-3: Sirkel

$$T = 157.08; \quad t = 0:dt:T \quad A = 1; \quad V = 100; \quad dt = 0.01;$$

$$\left[ \underline{p}^a(t), \underline{v}^a(t), \underline{a}^a(t) \right] = SBA(t, T, V, A)$$

$$\theta(T) = \frac{AT}{V};$$

$$\text{function } \left[ \underline{p}^a(t), \underline{v}^a(t), \underline{a}^a(t) \right] = SBA(t, T, V, A)$$

$$\omega = \frac{AT}{V}; \quad \theta(t) = \omega t;$$

$$\underline{p}^a(t) = \frac{V}{\Omega} \begin{bmatrix} C(\Omega t) \\ S(\Omega t) \\ 0 \end{bmatrix};$$

$$\underline{v}^a(t) = \begin{bmatrix} \cos(\theta(t)) \\ \sin(\theta(t)) \\ 0 \end{bmatrix};$$

$$\underline{a}^a(t) = \begin{bmatrix} -\sin(\theta(t)) \\ \cos(\theta(t)) \\ 0 \end{bmatrix};$$

### Appendiks A-4: Omvendt Eulerspiral

$$T = 314.16; \quad t = 0:dt:T; \quad A = 1; \quad V = 100; \quad dt = 0.01;$$

$$\theta(t) = \frac{A}{V} \left( t - \frac{t^2}{2T} \right); \quad \theta(T) = \frac{AT}{2V}; \quad \Omega = \sqrt{\frac{A}{\pi VT}};$$

$$\underline{p}^a(t) = \frac{V}{\Omega} \begin{bmatrix} -\cos(\theta_T) & -\sin(\theta_T) & 0 \\ -\sin(\theta_T) & \cos(\theta_T) & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} C(\Omega(T-t)) - C(\Omega T) \\ S(\Omega(T-t)) - S(\Omega T) \\ 0 \end{bmatrix};$$

$$\underline{v}^a(t) = V \begin{bmatrix} \cos(\theta(t)) \\ \sin(\theta(t)) \\ 0 \end{bmatrix};$$

$$\underline{a}^a(t) = \frac{A(T-t)}{T} \begin{bmatrix} -\sin(\theta(t)) \\ \cos(\theta(t)) \\ 0 \end{bmatrix};$$

Verdiene for T er avhengig av svingen og vil derfor være banespesifikk. Her er T verdiene til de fire første baneelementene i BD arrayet brukt brukt.

### Appendiks A-5: TNS (Heuns)

```

N = size(t, 2);
p_b^n(:, 1) = p_b^{n'}(:, 1) = p^n(:, 1);
v_b^n(:, 1) = v_b^{n'}(:, 1) = v^n(:, 1);
R_b^n(0) = R_b^{n'}(0) = inversE(theta(:, 1));
u = [f^b; omega_b^{nb}];
For k=1:N-1
p_b^{n'} = p_b^n + dt * v_b^n;
v_b^{n'} = v_b^n + dt * (R_b^n * f_k^b - g^n);
R_b^{n'} = R_b^n + dt * (R_b^n * Skew(omega_b^{nb}));
p_b^n = p_b^n + dt/2 * (v_b^n + v_b^{n'});
v_b^n = v_b^n + dt/2 * (R_b^n * f_k^b - g^n + R_b^{n'} * f_{k+1}^b - g^n);
R_b^{n'} = R_b^n + dt/2 * (R_b^n * Skew(omega_b^{nb}) + R_b^{n'} * Skew(omega_b^{nb}));
theta_{k+1} = inversE(R_b^{n'});
end

```

### Appendiks A-6: Akselerometer

```

I = eye(3);
T^a = 1200;
sigma_mu = 2.45 * 10^-4;
Q_mu^a = I * 2/T^a * sigma_mu^2;
Q_v^a = (1.47 * 10^-4)^2;

```

$$\tilde{Q}^a = \begin{bmatrix} I * \tilde{Q}_v^a & I * 0 \\ I * 0 & I * \tilde{Q}_\mu^a \end{bmatrix};$$

$$P_0^a = \begin{bmatrix} I * 0 & I * 0 \\ I * 0 & I * \sigma_\mu^2 \end{bmatrix};$$

$$F^a = \begin{bmatrix} I * 0 & I * 0 \\ I * 0 & -\frac{1}{T} * I \end{bmatrix};$$

$G^a = \text{eye}(6);$

Diskretisering

$$[\Gamma, \Gamma Q \Gamma^T] = \text{kp2dpGa}(F^a, G, \tilde{Q}^a, \Delta t);$$

$$Q_v^a = \Delta t * \tilde{Q}_v^a;$$

$$Q_\mu^a = \tilde{Q}_\mu^a * \frac{T}{2} (1 - e^{-2 * \frac{\Delta t}{T}});$$

$$Q^a = \begin{bmatrix} Q_v^a & I * 0 \\ I * 0 & Q_\mu^a \end{bmatrix};$$

$$\underline{v}^a \sim \mathcal{N}(\underline{0}, Q^a);$$

Systemet diskretiseret

$$\underline{x}^a(0) = \sqrt{Q^a} \text{randn}(6,1);$$

$$\sigma_0^a = \sqrt{\text{diag}(P_0^a)};$$

for  $k = 1:N - 1$

$$\underline{x}_{k+1}^a = \phi^a \underline{x}_k^a + \sqrt{Q^a} \text{randn}(6,1);$$

$$\overline{P}_{k+1}^a = \phi^a \overline{P}_k^a \phi^T + \Gamma Q \Gamma^T;$$

$$\sigma_{k+1}^a = \sqrt{\text{diag}(P_{k+1}^a)};$$

end

$$\delta \underline{f}_k = \underline{x}_k^a(4:6) + \underline{x}_k^a(1:3);$$

### Appendiks A-7: Gyro

$$I = \text{eye}(3);$$

$$T^g = 600;$$

$$\sigma_\mu = 1.45 * 10^{-8};$$

$$\tilde{Q}_\mu^g = I * \frac{2}{T^a} * \sigma_\mu^2;$$

$$\tilde{Q}_v^g = (1.76 * 10^{-6})^2;$$

$$\tilde{Q}^g = \begin{bmatrix} I * \tilde{Q}_v^g & I * 0 \\ I * 0 & I * \tilde{Q}_\mu^g \end{bmatrix};$$

$$P_0^g = \begin{bmatrix} I * 0 & I * 0 \\ I * 0 & I * \sigma_\mu^2 \end{bmatrix};$$

$$F^g = \begin{bmatrix} I * 0 & I * 0 \\ I * 0 & -\frac{1}{T} * I \end{bmatrix};$$

$$G^g = \text{eye}(6);$$

Diskretisering

$$[\Gamma, \Gamma Q \Gamma^T] = \text{kp2dpGa}(F^g, G, \tilde{Q}^g, \Delta t);$$

$$Q_v^g = \Delta t * \tilde{Q}_v^g;$$

$$Q_\mu^g = \tilde{Q}_\mu^g * \frac{T}{2} (1 - e^{-2 * \frac{\Delta t}{T}});$$

$$Q^g = \begin{bmatrix} Q_v^g & I * 0 \\ I * 0 & Q_\mu^g \end{bmatrix};$$

$$\underline{v}^g \sim \mathcal{N}(\underline{0}, Q^g);$$

Systemet diskretisert

$$\underline{x}^g(0) = \sqrt{Q^g} \text{randn}(6,1);$$

$$\sigma_0^g = \sqrt{\text{diag}(P_0^g)};$$

for  $k = 1:N - 1$

$$\underline{x}_{k+1}^g = \phi^g \underline{x}_k^g + \sqrt{Q^g} \text{randn}(6,1);$$

$$\overline{P}_{k+1}^g = \phi^g \overline{P}_k^g \phi^T + \Gamma Q \Gamma^T;$$

$$\sigma_{k+1}^g = \sqrt{\text{diag}(P_{k+1}^g)};$$

end

$$\delta \underline{f}_k = \underline{x}_k^g(4:6) + \underline{x}_k^g(1:3);$$

### Appendiks A-8: Linearisert Kalmanfilter

Tidsoppdatering (TO)

$$F = \begin{bmatrix} 0 & I & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -S(\tilde{R}_b^n \tilde{f}^b) & \tilde{R}_b^n & \tilde{R}_b^n & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \tilde{R}_b^n & \tilde{R}_b^n \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{T^a} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T^g} \end{bmatrix}$$

$$G = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \tilde{R}_b^n & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \tilde{R}_b^n & 0 \\ 0 & 0 & 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & I \end{bmatrix}$$

$$\Phi = \text{expm}(F \Delta_t);$$

$$[\Gamma, \Gamma Q \Gamma^T] = \text{kp2dpGa}(F, G, \tilde{Q}, \Delta_t);$$

$$\delta \underline{\bar{x}}_{k+1} = \Phi \delta \underline{\bar{x}}_k;$$

$$\overline{P}_k = \Phi \hat{P}_k \Phi^T + \Gamma Q \Gamma^T;$$

$$\underline{\bar{x}}_k = \tilde{\underline{x}}_k + \delta \underline{\bar{x}}_k;$$

$$\overline{R}_b^n = \tilde{R}_b^n + (I + S(\epsilon)) \tilde{R}_b^n;$$

## Appendiks B: Matlabkoder

### Appendiks B-1: Main

```
%-----3-D Banegenerator-----
%-----Main-----
close all
clear all
clc

%-----Velg banetype-----

    VD=[1 11 12 13; %[Sirkelbane Hvit+Farget Hvit Farget] velg bane og støy
        2 21 22 23]; %[3D bane Hvit+Farget Hvit Farget] velg bane og støy
velgbane=[1 11]; %[Banetype støytpe]

if velgbane(1)==1
isw=3;
elseif velgbane(1)==2
isw=5;
end

if isw==1 % Rett bane med akselerasjon
    BD=[1 30 90 100 [0 0 0]];
elseif isw==2 % ES
    BD=[2 69.8 100 1 [0 0 0]];
elseif isw==3 % Sirkel
    BD=[3 3600 100 10 [0 0 0]]; %%30 grader om z
elseif isw==4 % Omvendt ES
    BD=[4 5.23 5 1 [0 0 0]];
elseif isw==5 % S sammensatt bane

    BD=[1 143 90 100 [-2*0.087 0 2*pi/4]; % Rett bane med akselerasjon
        2 314.16 100 1 [-0.087 0 0]; % ES
        3 157.08 100 1 [0 0 1.5708]; % Sirkel
        4 314.16 100 1 [0 0 1.5708]; % Omvendt ES
        5 150 100 100 [-pi 0 1.5708]; % Rett bane u/ akselerasjon
        6 418.88 100 1 [0 0 0]; % ES
        7 209.44 100 1 [0 0 2.0944]; % Sirkel
        8 418.88 100 1 [0 0 2.0944]; % Omvendt ES
        9 200 100 100 [-1.5*5*pi/6 0 2.0944]; % Rett bane u/ akselerasjon
        10 471.24 100 1 [0 0 0]; % ES
        11 235.62 100 1 [0 0 2.3562]; % sirkel
        12 471.24 100 1 [0 0 2.3562]; % Omvendt ES
        13 112 100 100 [0 0 2.3562]]; % Rett bane u/ akselerasjon
end

%-----initialiseringer banegenerator
dt=1;
d_d=0.000001;

%-----initialiseringer støy akselerometer

T__a=1200; %korrelasjonstid farget støy akselerometer
std_ag=2.45*10^(-4); %standardavvik farget støy akselerometer
Qb_a_g=(2/T__a)*std_ag^2; %spektraltetthet farget støy akselerometer
Qb_a_v=(1.47*10^(-4))^2; %spektraltetthet hvit støy akselerometer
Qb_a=diag([Qb_a_v Qb_a_v Qb_a_v Qb_a_g Qb_a_g Qb_a_g]);
P0__a=diag([0 0 0 std_ag^2 std_ag^2 std_ag^2]); %initiell kovarians
F_a=[zeros(3,3) zeros(3,3);zeros(3,3) (-1/T__a)*eye(3)];%Systemmatrise

%-----initialiseringer støy gyro

T__g=600; %korrelasjonstid farget støy gyro
std_gg=1.45*10^(-8); %standardavvik farget støy gyro
Qb_g_g=(2/T__g)*std_gg^2; %spektraltetthet farget støy gyro
Qb_g_v=(1.76*10^(-6))^2; %spektraltetthet hvit støy gyro
Qb_g=diag([Qb_g_v Qb_g_v Qb_g_v Qb_g_g Qb_g_g Qb_g_g]);
```

```

P0__g=diag([0 0 0 std_gg^2 std_gg^2 std_gg^2]); %Initiell kovarians
F_g=[zeros(3,3) zeros(3,3);zeros(3,3) (-1/T__g)*eye(3)]; %Systemmatrise

%-----Banegenerator numerisk-----
[f__n,A,R,twp,Awp,Vwp,Pwp]=BaneGen(dt,d_d,BD); %Generer banen
u=A(10:15,:); %u=[f__b;w__b_nb]

% % %-----Prosessmodell-----

[x,H,att]=NAV(dt,u,A); %Navigasjonslikninger uten støy
x_euler=x; %Mellomlagring tilstander Euler
x_heuns=H; %Mellomlagring tilstander Heuns
Eu=att; %Mellomlagring stilling H&E
% %-----Legger til støy på gyro-----
value=1; %Test verdi 3-D bane eller sirkel

[hvit,farget,error,s_g]=Stoy(A(19,:),dt,T__g,Qb_g,P0__g,F_g,value);

if velgbane==[1 11]|velgbane==[2 21] %Valg av støytype
u(4:6,1:end-1)=u(4:6,1:end-1)+error; %Valg av støytype
elseif velgbane==[1 12]|velgbane==[2 22] %Valg av støytype
u(4:6,1:end-1)=u(4:6,1:end-1)+hvit; %Valg av støytype
elseif velgbane==[1 13]|velgbane==[2 23] %Valg av støytype
u(4:6,1:end-1)=u(4:6,1:end-1)+farget; %Valg av støytype
end

% %-----Legger til støy på akselerometer-----%
value=2; %Test verdi 3-D bane eller sirkel

[hvit,farget,error,s_g]=Stoy(A(19,:),dt,T__a,Qb_a,P0__a,F_a,value);

if velgbane==[1 11]|velgbane==[2 21] %Valg av støytype
u(1:3,1:end-1)=u(1:3,1:end-1)+error; %Valg av støytype
elseif velgbane==[1 12]|velgbane==[2 22] %Valg av støytype
u(1:3,1:end-1)=u(1:3,1:end-1)+hvit; %Valg av støytype
elseif velgbane==[1 13]|velgbane==[2 23] %Valg av støytype
u(1:3,1:end-1)=u(1:3,1:end-1)+farget; %Valg av støytype
end
end
% % % % %-----Navigasjonslikninger-----

[x,H,att]=NAV(dt,u,A); %beregner navigasjonlikningene
xm_euler=x; %Mellomlagring tilstander målt Euler
xm_heuns=H; %Mellomlagring tilstander målt Heuns
attm=att; %Mellomlagring stilling målt E&H
% % % % %-----Figurer-----

figur(twp,Pwp,Vwp,Awp,xm_heuns,xm_euler,A,x_euler,x_heuns,velgbane);

```

## Appendiks B-2: Banegenerator

```
function [f__n,A,R,twp,Awp,Vwp,Pwp]=BaneGen(dt,d_d,BD)
```

```

%-----Initialisering Banegenerator-----
NB=size(BD,1);
p_k_n=zeros(3,NB+1);
p_k_n2=zeros(3,NB+1);
p_k_n(:,1)=[0;0;0];
p_k_n2(:,1)=[0;0;0];
th=zeros(1,NB+1);
th(1)=0;
g=9.81;
g__n=[0; 0; g];
R_k_n=eye(3);

%-----Genererer Banelement-----
for k=1:NB

```



```

T=BD(k,2);V=BD(k,3); AV=BD(k,4);
t=0:dt:T; NT=size(t,2);

if BD(k,1)==1
    [p__a,v__a,a__a]=RBA(t,T,V,AV);           %Rett bane
    [p2__a,v2__a,a2__a]=RBA(t+d_d,T,V,AV);   %Rett bane
    th_v=0;

elseif BD(k,1)==2
    [p__a,v__a,a__a]=ESP(t,T,V,AV);           %Euler spiral
    [p2__a,v2__a,a2__a]=ESP(t+d_d,T,V,AV);   %Euler spiral
    th_v=AV*T/(2*V);

elseif BD(k,1)==3
    [p__a,v__a,a__a]=SBA(t,T,V,AV);           %Sirkelbane
    [p2__a,v2__a,a2__a]=SBA(t+d_d,T,V,AV);   %Sirkelbane
    th_v=AV*T/V;

elseif BD(k,1)==4
    [p__a,v__a,a__a]=OES(t,T,V,AV);           %Omvendt Eulerspiral
    [p2__a,v2__a,a2__a]=OES(t+d_d,T,V,AV);   %Omvendt Eulerspiral
    th_v=AV*T/(2*V);

elseif BD(k,1)==5
    [p__a,v__a,a__a]=RBA(t,T,V,AV);           %Rett bane
    [p2__a,v2__a,a2__a]=RBA(t+d_d,T,V,AV);   %Rett bane
    th_v=0;

elseif BD(k,1)==6
    [p__a,v__a,a__a]=ESP(t,T,V,AV);           %Euler spiral
    [p2__a,v2__a,a2__a]=ESP(t+d_d,T,V,AV);   %Euler spiral
    th_v=AV*T/(2*V);

elseif BD(k,1)==7
    [p__a,v__a,a__a]=SBA(t,T,V,AV);           %Sirkelbane
    [p2__a,v2__a,a2__a]=SBA(t+d_d,T,V,AV);   %Sirkelbane
    th_v=AV*T/V;

elseif BD(k,1)==8
    [p__a,v__a,a__a]=OES(t,T,V,AV);           %Omvendt Eulerspiral
    [p2__a,v2__a,a2__a]=OES(t+d_d,T,V,AV);   %Omvendt Eulerspiral
    th_v=AV*T/(2*V);

elseif BD(k,1)==9
    [p__a,v__a,a__a]=RBA(t,T,V,AV);           %Rett bane
    [p2__a,v2__a,a2__a]=RBA(t+d_d,T,V,AV);   %Rett bane
    th_v=0;

elseif BD(k,1)==10
    [p__a,v__a,a__a]=ESP(t,T,V,AV);           %Euler spiral
    [p2__a,v2__a,a2__a]=ESP(t+d_d,T,V,AV);   %Euler spiral
    th_v=AV*T/(2*V);

elseif BD(k,1)==11
    [p__a,v__a,a__a]=SBA(t,T,V,AV);           %Sirkelbane
    [p2__a,v2__a,a2__a]=SBA(t+d_d,T,V,AV);   %Sirkelbane
    th_v=AV*T/V;

elseif BD(k,1)==12
    [p__a,v__a,a__a]=OES(t,T,V,AV);           %Omvendt Eulerspiral
    [p2__a,v2__a,a2__a]=OES(t+d_d,T,V,AV);   %Omvendt Eulerspiral
    th_v=AV*T/(2*V);

elseif BD(k,1)==13
    [p__a,v__a,a__a]=RBA(t,T,V,AV);           %Rett bane

```

```

    [p2__a,v2__a,a2__a]=RBA(t+d_d,T,V,AV);           %Rett bane
    th_v=0;

    end
    R_k_n=R_k_n*R_a_n(BD(k,5:7)');                 %Regner ut KTM matrisen
    th(k+1)=th(k)+th_v;
    p_k_n(:,k+1)=p_k_n(:,k)+R_k_n*p__a(:,NT);
    p_k_n(:,k+1)=p_k_n(:,k)+R_k_n*p2__a(:,NT);

%-----Setter sammen banen-----
    if k==1
        tp=[t];
        p__n= repmat(p_k_n(:,k),1,NT)+R_k_n*p__a(:,:); %Skjøter posisjon
        v__n=R_k_n*v__a;
        a__n=R_k_n*a__a;
        p2__n= repmat(p_k_n(:,k),1,NT)+R_k_n*p2__a(:,:);
        v2__n=R_k_n*v2__a;
        a2__n=R_k_n*a2__a;
        twp=[tp(1); NT];
        twp=[twp [tp(1,end);0]];
        Pwp=p__n(:,1);
        Pwp=[Pwp p__n(:,end)];
        Vwp=v__n(:,1);
        Vwp=[Vwp v__n(:,end)];
        Awp=a__n(:,1);
        Awp=[Awp a__n(:,end)];

        for i=1:NT
            f__n(:,i)=g__n+a__n(:,i);             %spesifikk kraft
        end

        for i=1:NT
            b_1_n(:,i)=v__n(:,i)/norm(v__n(:,i));
            b_2_n(:,i)=cross(f__n(:,i),b_1_n(:,i))/norm(cross(f__n(:,i),b_1_n(:,i)));
            b_3_n(:,i)=cross(b_1_n(:,i),b_2_n(:,i))/norm(cross(b_1_n(:,i),b_2_n(:,i)));
        end

        for i=1:NT
            f2__n(:,i)=g__n+a2__n(:,i);           %spesifikk kraft
        end

        for i=1:NT
            b2_1_n(:,i)=v2__n(:,i)/norm(v2__n(:,i));
            b2_2_n(:,i)=cross(f2__n(:,i),b2_1_n(:,i))/norm(cross(f2__n(:,i),b2_1_n(:,i)));
            b2_3_n(:,i)=cross(b2_1_n(:,i),b2_2_n(:,i))/norm(cross(b2_1_n(:,i),b2_2_n(:,i)));
        end

    else
        if k<=NB

            a__nx=R_k_n*a__a;
            v__nx=R_k_n*v__a;
            a2__nx=R_k_n*a2__a;
            v2__nx=R_k_n*v2__a;

            end

        tp=[tp(1,1:end-1) tp(end)+t];
        p__n=[p__n(:,1:end-1), repmat(p_k_n(:,k),1,NT)+R_k_n*p__a(:,:)];
        v__n=[v__n(:,1:end-1), R_k_n*v__a];
        a__n=[a__n(:,1:end-1), R_k_n*a__a];
        p2__n=[p2__n(:,1:end-1), repmat(p_k_n(:,k),1,NT)+R_k_n*p2__a(:,:)];
        v2__n=[v2__n(:,1:end-1), R_k_n*v2__a];
        a2__n=[a2__n(:,1:end-1), R_k_n*a2__a];
        twp(2,k)=NT;

        twp=[twp [tp(end);0]];
        Pwp=[Pwp p__n(:,end)];
        Vwp=[Vwp v__n(:,end)];

```

```

Awp=[Awp a_n(:,end)];

[b2_1_n,b2_2_n,b2_3_n,f2_n,b_1_n,b_2_n,b_3_n,f_n]=RKM(f2_n,a2_nx,v2_nx,a_nx,v
_nx,f_n,NT,g_n,b_1_n,b_2_n,b_3_n,b2_1_n,b2_2_n,b2_3_n);
end

[w_b_nb,R]=R_b_nd(tp,b_1_n,b_2_n,b_3_n,b2_1_n,b2_2_n,b2_3_n,d_d); %finner
vinkelhastighet

for i=1:size(tp,2)
    f_b(:,i)=R(:,i)*f_n(:,i); %spesifikk kraft representert i b-
systemer
    E_v(:,i)=EV(R(:,i)); %Eulervinkler
end
p_n(:,end)=[];
v_n(:,end)=[];
a_n(:,end)=[];
f_b(:,end)=[];
tp(:,end)=[];
E_v(:,end)=[];

A=[p_n/v_n;a_n/f_b;w_b_nb;E_v;tp]; %Lagrer data fra banegenerator
end

```

### Appendiks B-3: Flyets stilling

```

function
[b2_1_n,b2_2_n,b2_3_n,f2_n,b_1_n,b_2_n,b_3_n,f_n]=RKM(f2_n,a2_nx,v2_nx,a_nx,v
_nx,f_n,NT,g_n,b_1_n,b_2_n,b_3_n,b2_1_n,b2_2_n,b2_3_n)
%[b_1_n,b_2_n,b_3_n,f_n,,tempf]=RKM(a_nx,v_nx,f_n,NT,g_n,b_1_n,b_2_n,b_3_n,f
_b)
for i=1:NT
    tempf(:,i)=g_n+a_nx(:,i);
    tempf2(:,i)=g_n+a2_nx(:,i);
end
f_n=[f_n(:,1:end-1),tempf];
f2_n=[f2_n(:,1:end-1),tempf2];

for i=1:NT
    temp1(:,i)=v_nx(:,i)/norm(v_nx(:,i));
    temp2(:,i)=cross(tempf(:,i),temp1(:,i))/norm(cross(tempf(:,i),temp1(:,i)));
    temp3(:,i)=cross(temp1(:,i),temp2(:,i))/norm(cross(temp1(:,i),temp2(:,i)));

    temp21(:,i)=v2_nx(:,i)/norm(v2_nx(:,i));
    temp22(:,i)=cross(tempf2(:,i),temp21(:,i))/norm(cross(tempf2(:,i),temp21(:,i)));
    temp23(:,i)=cross(temp21(:,i),temp22(:,i))/norm(cross(temp21(:,i),temp22(:,i)));
end

b_1_n=[b_1_n(:,1:end-1),temp1];
b_2_n=[b_2_n(:,1:end-1),temp2];
b_3_n=[b_3_n(:,1:end-1),temp3];

b2_1_n=[b2_1_n(:,1:end-1),temp21];
b2_2_n=[b2_2_n(:,1:end-1),temp22];
b2_3_n=[b2_3_n(:,1:end-1),temp23];
end

```

## Appendiks B-4: Vinkelhastighet

```
function [w_b_nb,R]=R_b_nd(tp,b_1_n,b_2_n,b_3_n,b2_1_n,b2_2_n,b2_3_n,d_d);
s=size(tp,2);

R=zeros(3,3,s);
for i=1:s
    R(:,:,i)=[b_1_n(:,i),b_2_n(:,i),b_3_n(:,i)];    %Setter sammen stillingsmatrisen
for hver tidsskritt
end
for i=1:s
    R2(:,:,i)=[b2_1_n(:,i),b2_2_n(:,i),b2_3_n(:,i)];    %Setter sammen
stillingsmatrisen for hver tidsskritt
end

R_d=zeros(3,3,s-1);
for i=1:s-1
R_d(:,:,i)=(R2(:,:,i)-R(:,:,i))/d_d;                %Finner den deriverte av
stillingsmatrisen
S(:,:,i)=R(:,:,i)'*R_d(:,:,i);                    %Setter sammen skjevsymmetriske matrise
end
w_b_nb=zeros(3,s-1,'int64');
for i=1:s-1
w_b_nb(1,i)=S(3,2,i);                                %Trekker ut vinkelhastighet x
w_b_nb(2,i)=S(1,3,i);                                %Trekker ut vinkelhastighet y
w_b_nb(3,i)=S(2,1,i);                                %Trekker ut vinkelhastighet z
end
```

## Appendiks B-5: 3-2-1 Eulervinkler

```
function [R_a_n]=R_a_n(v)
th1=v(1,1);
th2=v(2,1);
th3=v(3,1);
%-----Elementære RKM-----
R_x=[1 0 0;
0 cos(th1) -sin(th1);
0 sin(th1) cos(th1)];

R_y=[ cos(th2) 0 sin(th2);
0 1 0;
-sin(th2) 0 cos(th2)];

R_z=[cos(th3) -sin(th3) 0;
sin(th3) cos(th3) 0;
0 0 1];

R_a_n=R_z*R_y*R_x; %3-2-1 Eulervinkler
end
```

## Appendiks B-6: Navigasjonslikningene

```
function [x,H,att]=NAV(dt,u,A)
%-----Initialiseringer-----
N=size(A,2);
g__n=[0;0;9.81];
%-----Definerer starttilstand-----
p_b_n(:,1)=A(1:3,1);
v_b_n(:,1)=A(4:6,1);
evb(:,1)=A(16:18,1);
Rb_b_bn=R_a_n(A(16:18,1));
pe_b_n=A(1:3,1);
ve_b_n=A(4:6,1);
Re_b_bn=R_a_n(A(16:18,1));
evbe=A(16:18,1);
x(:,1)=[pe_b_n;ve_b_n];
att=zeros(6,N);
%-----Eulers metode-----
att(1:3,1)=evbe;
for k=1:N-1
    f_b=u(1:3,k); om=(u(4:6,k));

    pe_b_n=pe_b_n+dt*ve_b_n;
    ve_b_n=ve_b_n+dt*(Re_b_bn*f_b-g__n);
    Re_b_bn=Re_b_bn+dt*Re_b_bn*Skew(om);
    evbe=EV(Re_b_bn);
    x(:,k+1)=[pe_b_n;ve_b_n];
    att(1:3,k+1)=evbe;
end

%-----Heuns metode-----

att(4:6,1)=evb;
H(:,1)=[p_b_n;v_b_n];

for k=1:N-1
    f_b=u(1:3,k); fmb=u(1:3,k+1); om=(u(4:6,k)); omm=(u(4:6,k+1));

    pm_b_n=p_b_n+dt*v_b_n;
    vm_b_n=v_b_n+dt*(Rb_b_bn*f_b-g__n);
    Rm_b_bn=Rb_b_bn+dt*Rb_b_bn*Skew(om);
    evbm=EV(Rm_b_bn);

    p_b_n=p_b_n+dt*(v_b_n+vm_b_n)/2;
    v_b_n=v_b_n+dt*(Rb_b_bn*f_b-g__n+Rm_b_bn*fmb-g__n)/2;
    Rb_b_bn=Rb_b_bn+dt*(Rb_b_bn*Skew(om)+Rm_b_bn*Skew(omm))/2;
    evb=EV(Rb_b_bn);
    H(:,k+1)=[p_b_n;v_b_n];
    att(4:6,k+1)=evb;

end
end
```

## Appendiks B-7: Skjevensymmetriske matrisen

```
function [S]=Skew(e_v)

S=[0 -e_v(3) e_v(2);
   e_v(3) 0 -e_v(1);
  -e_v(2) e_v(1) 0];
end
```

## Appendiks B-8: Støyfunksjonen

```

function [hvit,farget,error,s_g]=Stoy(tp,dt,T,Qb,P0,F,value)

I=eye(3);
N=size(tp,2);
G=diag([1 1 1 1 1 1]);
H=[I I];

%-----diskretiserer-----
Fi=expm(F*dt);
[Ga,S]=kp2dpGa(F,G,Qb,dt);
%-----trekker initialverdier-----
Qb(1:3,1:3)=Qb(1:3,1:3)*dt;
Qb(4:6,4:6)=Qb(4:6,4:6)*(T/2)*(1-exp(-2*dt/T));
x(1:3,1)=sqrt(Qb(1:3,1:3))*randn(3,1);
x(4:6,1)=sqrt(Qb(4:6,4:6))*randn(3,1);
xm(1:3,1)=zeros(3,1);
xm(4:6,1)=zeros(3,1);
P(:,1)=P0;
s_g(:,1)=sqrt(diag(P(:,1)));

for k=1:N-1
x(:,k+1)=Fi*x(:,k)+sqrt(Qb)*randn(6,1);    %Bruker Ga eller sqrt(Qb)
xm(:,k+1)=Fi.^k*xm(:,1);                  %Middelverdi tilstanden
P(:,k+1)=Fi*P(:,k)*Fi'+S;                 %Kovarians utvikling
v(:,k)=(x(1:3,k+1)-x(1:3,k))/dt;           %Hvitstøyen derivert
s_g(:,k+1)=sqrt(diag(P(:,k+1)));          %Regner ut standardavviket
end
error=x(4:6,1:end-1)+v;                    %Hvit+farget støy
hvit=v; farget=x(4:6,1:end-1);
for k=1:N
kov(:,k)=diag(P(:,k));                    %Diagonaliserer kovariansmatrisen
end

if value==1
figure;
subplot(2,2,1); plot(tp,x(1,:)); hold on; plot(tp,xm(1,:)+s_g(1,:), 'r');hold
on;plot(tp,xm(1,:)-s_g(1,:), 'g');
subplot(2,2,1); plot(tp,x(1,:)); hold on; plot(tp,xm(1,:)+s_g(1,:), 'r');hold
on;plot(tp,xm(1,:)-s_g(1,:), 'g');
title('ARW \theta_x');xlabel('tid [s]'); ylabel('[rad]');
subplot(2,2,2); plot(tp,x(2,:)); hold on; plot(tp,xm(2,:)+s_g(2,:), 'r');hold on;
plot(tp,xm(2,:)-s_g(2,:), 'g');
title('ARW \theta_y');xlabel('tid [s]'); ylabel('[rad]');
subplot(2,2,3); plot(tp,x(3,:)); hold on; plot(tp,xm(3,:)+s_g(3,:), 'r');hold on;
plot(tp,xm(3,:)-s_g(3,:), 'g');
title('ARW \theta_z');xlabel('tid [s]'); ylabel('[rad]');
subplot(2,2,4); plot(tp(:,1:end-1),kov(1:3,1:end-1));
title('Middelverdiutvikling av kovariansen P_A_R_W for gyro');xlabel('tid [s]');
ylabel('P [rad^2]');
ha = axes('Position',[0 0 1 1], 'Xlim',[0 1], 'Ylim',[0
1], 'Box', 'off', 'Visible', 'off', 'Units', 'normalized', 'clipping', 'off');
text(0.5, 1, '\bf Gyro (Angular Random
Walk)', 'HorizontalAlignment', 'center', 'VerticalAlignment', 'top');

figure;
subplot(2,2,1); plot(tp,x(4,:)); hold on; plot(tp,xm(4,:)+s_g(4,:), 'r'); hold on;
plot(tp,xm(4,:)-s_g(4,:), 'g');hold on;plot(tp,xm(4,:));
title('Gauss Markov bidrag \omega_x');xlabel('tid [s]'); ylabel('[rad/s]');
subplot(2,2,2); plot(tp,x(5,:)); hold on; plot(tp,xm(5,:)+s_g(5,:), 'r'); hold on;
plot(tp,xm(5,:)-s_g(5,:), 'g');hold on;plot(tp,xm(5,:));
title('Gauss Markov bidrag \omega_y');xlabel('tid [s]'); ylabel('[rad/s]');
subplot(2,2,3); plot(tp,x(6,:)); hold on; plot(tp,xm(6,:)+s_g(6,:), 'r'); hold on;
plot(tp,xm(6,:)-s_g(6,:), 'g');hold on;plot(tp,xm(6,:));
title('Gauss Markov bidrag \omega_z');xlabel('tid [s]'); ylabel('[rad/s]');
subplot(2,2,4); plot(tp,kov(4:6,:));

```

```

title('Middelverdiutvikling av kovariansen P_\gamma for gyro');xlabel('tid');
ylabel(['rad^2/s^2']);
ha = axes('Position',[0 0 1 1],'Xlim',[0 1],'Ylim',[0
1],'Box','off','Visible','off','Units','normalized','clipping','off');
text(0.5, 1, '\bf Gyro (1.ordens Gauss
Markov)','HorizontalAlignment','center','VerticalAlignment','top');
end

if value==2
figure;

subplot(2,2,1); plot(tp,x(1,:)); hold on; plot(tp,xm(1,:)+s_g(1,:), 'r');hold
on;plot(tp,xm(1,:)-s_g(1,:), 'g');
title('VRW X');xlabel('tid [s]'); ylabel(['m/s']);
subplot(2,2,2); plot(tp,x(2,:)); hold on; plot(tp,xm(2,:)+s_g(2,:), 'r');hold on;
plot(tp,xm(2,:)-s_g(2,:), 'g');
title('VRW Y');xlabel('tid [s]'); ylabel(['m/s']);
subplot(2,2,3); plot(tp,x(3,:)); hold on; plot(tp,xm(3,:)+s_g(3,:), 'r');hold on;
plot(tp,xm(3,:)-s_g(3,:), 'g');
title('VRW Z');xlabel('tid [s]'); ylabel(['m/s']);
subplot(2,2,4); plot(tp,kov(1:3,:));
title('Middelverdiutvikling av kovariansen P_V_R_W for
akselerometer');xlabel('tid [s]'); ylabel('m^2/s^2');

ha = axes('Position',[0 0 1 1],'Xlim',[0 1],'Ylim',[0
1],'Box','off','Visible','off','Units','normalized','clipping','off');
text(0.5, 1, '\bf Akselerometer (Velocity random
walk)','HorizontalAlignment','center','VerticalAlignment','top');

figure;
subplot(2,2,1); plot(tp,x(4,:)); hold on; plot(tp,xm(4,:)+s_g(4,:), 'r'); hold on;
plot(tp,xm(4,:)-s_g(4,:), 'g');hold on;plot(tp,xm(4,:));
title('Gauss Markov bidrag X');xlabel('tid [s]'); ylabel(['m/s^2']);
subplot(2,2,2); plot(tp,x(5,:)); hold on; plot(tp,xm(5,:)+s_g(5,:), 'r'); hold on;
plot(tp,xm(5,:)-s_g(5,:), 'g');hold on;plot(tp,xm(5,:));
title('Gauss Markov bidrag Y');xlabel('tid [s]'); ylabel(['m/s^2']);
subplot(2,2,3); plot(tp,x(6,:)); hold on; plot(tp,xm(6,:)+s_g(6,:), 'r'); hold on;
plot(tp,xm(6,:)-s_g(6,:), 'g');hold on;plot(tp,xm(6,:));
title('Gauss Markov bidrag Z');xlabel('tid [s]'); ylabel(['m/s^2']);
subplot(2,2,4); plot(tp,kov(4:6,:));
title('Middelverdiutvikling av kovariansen P_\gamma for
akselerometer');xlabel('tid'); ylabel(['m^2/s^4']);
ha = axes('Position',[0 0 1 1],'Xlim',[0 1],'Ylim',[0
1],'Box','off','Visible','off','Units','normalized','clipping','off');
text(0.5, 1, '\bf Akselerometer (1.ordens Gauss
Markov)','HorizontalAlignment','center','VerticalAlignment','top');
end

```

## Appendiks B-9: Diskretisering av kontinuerlig system

```

function [Ga,S]=kp2dpGa(F,G,Qb_g,dt)
[m,n]=size(F);
F_b=[F G*Qb_g*G';zeros(m,n) -F'];
Fi_b=expm(F_b*dt); %eksponentiell utvikling av F dobbelbølge ganget med
d(=0,01s)
S=(Fi_b(1:m,n+1:2*n)*inv(Fi_b(m+1:2*n,n+1:2*n))); %GaQGat
Ga=chol(S)'; %Choleskyfaktorisering gir Gamma
end

```

## Appendiks B-10: Eulervinklene invers Euler

```

function [E_v]=EV(R_b_n)
E_v2=atan2(-R_b_n(3,1),sqrt(R_b_n(1,1)^2+R_b_n(2,1)^2));
E_v1=atan2(R_b_n(3,2)/cos(E_v2),R_b_n(3,3)/cos(E_v2));
E_v3=atan2(R_b_n(2,1)/cos(E_v2),R_b_n(1,1)/cos(E_v2));
E_v=[E_v1;E_v2;E_v3];
end

```

## Appendiks B-11: Figurer

```
function figur(twp,Pwp,Vwp,Awp,H,xm_euler,A,x_euler,x_heuns,velgbane);

if velgbane(1)==1;
figure; axis equal
subplot(2,2,1); plot3(A(1,:),A(2,:),A(3,:)); hold on
scatter3(Pwp(1,:),Pwp(2,:),Pwp(3,:),20,'k');
grid on; axis equal
title('Bane');xlabel('x [m]'); ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,2); plot(A(19,:),A(1:3,:)); hold on;
scatter(twp(1,:),Pwp(1,:),20,'k');
scatter(twp(1,:),Pwp(2,:),20,'k');
scatter(twp(1,:),Pwp(3,:),20,'k');
grid on; title('Posisjon');xlabel('t');ylabel('[m]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside')

subplot(2,2,3); plot(A(19,:),A(4:6,:)); hold on;
scatter(twp(1,:),Vwp(1,:),20,'k');
scatter(twp(1,:),Vwp(2,:),20,'k');
scatter(twp(1,:),Vwp(3,:),20,'k');
grid on; title('Hastighet');xlabel('t');ylabel('[m/s]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside')

subplot(2,2,4); plot(A(19,:),A(7:9,:)); hold on;
scatter(twp(1,:),Awp(1,:),20,'k');
scatter(twp(1,:),Awp(2,:),20,'k');
scatter(twp(1,:),Awp(3,:),20,'k');
grid on; title('Akselerasjon');xlabel('t');ylabel('[m/s^2]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside')

%-----Deterministisk bane Euler og Heuns integrasjon-----
figure;
subplot(2,2,1); plot3(x_euler(1,:),x_euler(2,:),x_euler(3,:)); hold on
grid on; axis equal
title('Euler posisjon deterministisk
sirkelbane');xlabel('x [m]'); ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,2); plot3(x_heuns(1,:),x_heuns(2,:),x_heuns(3,:)); hold on
grid on; axis equal
title('Heuns posisjon deterministisk
sirkelbane');xlabel('x [m]'); ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,3); plot3(x_euler(4,:),x_euler(5,:),x_euler(6,:)); hold on
grid on; axis equal
title('Euler hastighet deterministisk
sirkelbane');xlabel('x [m/s]'); ylabel('y [m/s]'); zlabel('z [m/s]');

subplot(2,2,4); plot3(x_heuns(4,:),x_heuns(5,:),x_heuns(6,:)); hold on
grid on; axis equal
title('Heuns hastighet deterministisk
sirkelbane');xlabel('x [m/s]'); ylabel('y [m/s]'); zlabel('z [m/s]');

%-----Differanse deterministisk-----
figure;
subplot(2,2,1); plot(A(19,:),x_euler(1:3,:)-A(1:3,:));
grid on;
title('Differanse sann og Euler posisjon deterministisk
sirkelbane');xlabel('t [s]'); ylabel('[m]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside');

subplot(2,2,2); plot(A(19,:),x_heuns(1:3,:)-A(1:3,:)); hold on
grid on;
```



```

title('                Differanse sann og Heuns posisjon deterministisk
sirkelbane');xlabel('t [s]'); ylabel('m');
hleg = legend('X','Y','Z',...
              'Location','NorthEastOutside');
subplot(2,2,3); plot(A(19,:),x_euler(4:6,)-A(4:6,)); hold on
grid on;
title('                Differanse sann og Euler hastiget deterministisk
sirkelbane');xlabel('t [s]'); ylabel('m/s');
hleg = legend('X','Y','Z',...
              'Location','NorthEastOutside');
subplot(2,2,4); plot(A(19,:),x_heuns(4:6,)-A(4:6,)); hold on
grid on;
title('                Differanse sann og Heuns hastighet deterministisk
sirkelbane');xlabel('t [s]'); ylabel('m/s');
hleg = legend('X','Y','Z',...
              'Location','NorthEastOutside');
%-----Sirkelbane med støy-----
figure;
subplot(2,2,1); plot3(xm_euler(1,:),xm_euler(2,:),xm_euler(3,)); hold on
grid on; axis equal
title('                Euler posisjon sirkelbane');xlabel('x [m]');
ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,2); plot3(H(1,:),H(2,:),H(3,)); hold on
grid on; axis equal
title('                Heuns posisjon sirkelbane');xlabel('x [m]');
ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,3); plot3(xm_euler(4,:),xm_euler(5,:),xm_euler(6,)); hold on
grid on; axis equal
title('                Euler hastighet sirkelbane');xlabel('x [m/s]');
ylabel('m/s');

subplot(2,2,4); plot3(H(4,:),H(5,:),H(6,)); hold on
grid on; axis equal
title('                Heuns hastighet sirkelbane');xlabel('x[m/s]');
ylabel('m/s');

%-----Differanse med støy sirkelbane-----
figure;
subplot(2,2,1); plot(A(19,:),xm_euler(1:3,)-A(1:3,));
grid on;
title('                Differanse sann og Euler posisjon stokastisk
sirkelbane');xlabel('t [s]'); ylabel('m');
hleg = legend('X','Y','Z',...
              'Location','NorthEastOutside');

subplot(2,2,2); plot(A(19,:),H(1:3,)-A(1:3,)); hold on
grid on;
title('                Differanse sann og Heuns posisjon stokastisk
sirkelbane');xlabel('t [s]'); ylabel('m');
hleg = legend('X','Y','Z',...
              'Location','NorthEastOutside');

subplot(2,2,3); plot(A(19,:),xm_euler(4:6,)-A(4:6,)); hold on
grid on;
title('                Differanse sann og Euler hastiget stokastisk
sirkelbane');xlabel('t [s]'); ylabel('m/s');
hleg = legend('X','Y','Z',...
              'Location','NorthEastOutside');

subplot(2,2,4); plot(A(19,:),H(4:6,)-A(4:6,)); hold on
grid on;
title('                Differanse sann og Heuns hastighet stokastisk
sirkelbane');xlabel('t [s]'); ylabel('m/s');
hleg = legend('X','Y','Z',...
              'Location','NorthEastOutside');

```

```

%-----3D-bane-----
elseif velgbane(1)==2
figure; axis equal
subplot(2,2,1); plot3(A(1,:),A(2,:),A(3,:)); hold on
scatter3(Pwp(1,:),Pwp(2,:),Pwp(3,:),20,'k');
grid on; axis equal
title('3D bane');xlabel('x [m]'); ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,2); plot(A(19,:),A(1:3,:)); hold on;
scatter(twp(1,:),Pwp(1,:),20,'k');
scatter(twp(1,:),Pwp(2,:),20,'k');
scatter(twp(1,:),Pwp(3,:),20,'k');
grid on; title('Posisjon');xlabel('t');ylabel('[m]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside')

subplot(2,2,3); plot(A(19,:),A(4:6,:)); hold on;
scatter(twp(1,:),Vwp(1,:),20,'k');
scatter(twp(1,:),Vwp(2,:),20,'k');
scatter(twp(1,:),Vwp(3,:),20,'k');
grid on; title('Hastighet');xlabel('t');ylabel('[m/s]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside')

subplot(2,2,4); plot(A(19,:),A(7:9,:)); hold on;
scatter(twp(1,:),Awp(1,:),20,'k');
scatter(twp(1,:),Awp(2,:),20,'k');
scatter(twp(1,:),Awp(3,:),20,'k');
grid on; title('Akselerasjon');xlabel('t');ylabel('[m/s^2]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside')
%-----Deterministisk bane Euler og Heuns integrasjon-----
figure;
subplot(2,2,1); plot3(x_euler(1,:),x_euler(2,:),x_euler(3,:)); hold on
grid on; axis equal
title('Euler posisjon deterministisk 3D bane');xlabel('x
[m]'); ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,2); plot3(x_heuns(1,:),x_heuns(2,:),x_heuns(3,:)); hold on
grid on; axis equal
title('Heuns posisjon deterministisk 3D
bane');xlabel('x [m]'); ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,3); plot3(x_euler(4,:),x_euler(5,:),x_euler(6,:)); hold on
grid on; axis equal
title('Euler hastighet deterministisk 3D
bane');xlabel('x [m/s]'); ylabel('y [m/s]'); zlabel('z [m/s]');

subplot(2,2,4); plot3(x_heuns(4,:),x_heuns(5,:),x_heuns(6,:)); hold on
grid on; axis equal
title('Heuns hastighet deterministisk 3D
bane');xlabel('x [m/s]'); ylabel('y [m/s]'); zlabel('z [m/s]');
%-----Differanse deterministisk-----
figure;
subplot(2,2,1); plot(A(19,:),x_euler(1:3,)-A(1:3,:));
grid on;
title('Differanse sann og Euler posisjon deterministisk
3D bane');xlabel('t [s]'); ylabel('[m]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside');

subplot(2,2,2); plot(A(19,:),x_heuns(1:3,)-A(1:3,:)); hold on
grid on;
title('Differanse sann og Heuns posisjon deterministisk
3D bane');xlabel('t [s]'); ylabel('[m]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside');

subplot(2,2,3); plot(A(19,:),x_euler(4:6,)-A(4:6,:)); hold on

```

```

grid on;
title('Differanse sann og Euler hastiget deterministisk
3D bane');xlabel('t [s]'); ylabel('[m/s]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside');
subplot(2,2,4); plot(A(19,:),x_heuns(4:6,)-A(4:6,)); hold on
grid on;
title('Differanse sann og Heuns hastighet deterministisk
3D bane');xlabel('t [s]'); ylabel('[m/s]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside');
%-----3D bane med støy-----
figure;
subplot(2,2,1); plot3(xm_euler(1,:),xm_euler(2,:),xm_euler(3,)); hold on
grid on; axis equal
title('Euler posisjon 3D bane');xlabel('x [m]');
ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,2); plot3(H(1,:),H(2,:),H(3,)); hold on
grid on; axis equal
title('Heuns posisjon 3D bane');xlabel('x [m]');
ylabel('y [m]'); zlabel('z [m]');

subplot(2,2,3); plot3(xm_euler(4,:),xm_euler(5,:),xm_euler(6,)); hold on
grid on; axis equal
title('Euler hastighet 3D bane');xlabel('x [m/s]');
ylabel('[m/s]');zlabel('[m/s]')

subplot(2,2,4); plot3(H(4,:),H(5,:),H(6,)); hold on
grid on; axis equal
title('Heuns hastighet 3D bane');xlabel('x[m/s]');
ylabel('[m/s]');zlabel('[m/s]')

%-----Differanse med støy 3D bane-----
figure;
subplot(2,2,1); plot(A(19,:),xm_euler(1:3,)-A(1:3,));
grid on;
title('Differanse sann og Euler posisjon stokastisk 3D
bane');xlabel('t [s]'); ylabel('[m]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside');

subplot(2,2,2); plot(A(19,:),H(1:3,)-A(1:3,)); hold on
grid on;
title('Differanse sann og Heuns posisjon stokastisk 3D
bane');xlabel('t [s]'); ylabel('[m]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside');

subplot(2,2,3); plot(A(19,:),xm_euler(4:6,)-A(4:6,)); hold on
grid on;
title('Differanse sann og Euler hastiget stokastisk 3D
bane');xlabel('t [s]'); ylabel('[m/s]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside');

subplot(2,2,4); plot(A(19,:),H(4:6,)-A(4:6,)); hold on
grid on;
title('Differanse sann og Heuns hastighet stokastisk 3D
bane');xlabel('t [s]'); ylabel('[m/s]');
hleg = legend('X','Y','Z',...
'Location','NorthEastOutside');

end
end

```