OPPDRAGSGIVER MRIH, ELEKTROAVDELING	REFERANSE
TITTEL HOVEDPROSJEKT MRIH AUTOMASJON 1994 MODELLERING SIMULERING REGULERING AV *OEV - FARTØY*	Dokument Dokument nr. Dokument type Rapport Dokument tilgang Åpen Dokument status Versjon nr. 1 Antall sider 79 Bibl. nr.

FORFATTERE Øyvind Kvålsvoll Jan Morten Krakeli	SIGNATUR
	DATO 01.06.1994

SAMMENDRAG

Dette er dokumentasjon for hovedoppgaven Modellering, Regulering, Simulering av OEV-fartøy, MRIH 1993/ 1994.

Hovedprosjektets målsetning og fremdriftsplan er beskrevet i PROSJEKTHÅNDBOK for hovedoppgaven Modellering, Regulering, Simulering av OEV- fartøy, MRIH 1993/ 1994. Forprosjekt er gitt som 1. vektallsfag ved MRIH, studieretning Automatiseringsteknikk, høsten 1993.

Forfatterne ville ha et hovedoppgaveprosjekt som omfatter modellering, simulering og regulering av et relativt komplekst system. Regulering av OEV- fartøy, en flyvende hydrofoil, ble valgt fordi det ser ut til å være en vanskelig oppgave å få til god regulering av en slik. Forfatterne mener også at dette er en fremtidsrettet og aktuell oppgave.

En ville se hvordan avansert reguleringsteknikk implementert i et objektorientert programmeringsspråk var i stand til å løse oppgaven, og samtidig lage et simuleringsprogram som var anvendelig til å løse lignende problemstillinger.

Rapporten er delt opp i to deler:

1. Modellering, Simulering, Regulering av OEV- fartøy.
2. Appendix for OEV- prosjektet.

EMNER
MODELLERING, SIMULERING, REGULERING

DISTRIBUSJON
MRIH

1. ORIENTERING

1.1. FORORD

Dette er dokumentasjon for hovedoppgaven Modellering, Regulering, Simulering av OEV- fartøy, MRIH 1993/ 1994.

Hovedprosjektets målsetning og fremdriftsplan er beskrevet i PROSJEKTHÅNDBOK for hovedoppgaven Modellering, Regulering, Simulering av OEV- fartøy, MRIH 1993/ 1994. Forprosjekt er gitt som 1. vektallsfag ved MRIH, studieretning Automatiseringsteknikk, høsten 1993.

Prosjekthåndboken inneholder spesifikasjon av oppgave, problemstilling, oversikt over nødvendige ressurser, målsetting, handlingsplaner, ansvarsområde for prosjektdeltakere.

Forfatterne ville ha et hovedoppgaveprosjekt som omfatter modellering, simulering og regulering av et relativt komplekst system. Regulering av OEV- fartøy, en flyvende hydrofoil, ble valgt fordi det ser ut til å være en vanskelig oppgave å få til god regulering av en slik. Forfatterne mener også at dette er en fremtidsrettet og aktuell oppgave.

En ville se hvordan avansert reguleringsteknikk implementert i et objektorientert programmeringsspråk var i stand til å løse oppgaven, og samtidig lage et simuleringsprogram som var anvendelig til å løse lignende problemstillinger.

Rapporten er delt opp i to deler:

1. Modellering, Simulering, Regulering av OEV- fartøy.
2. Appendix for OEV- prosjektet.

DE SOM VAR MED PÅ HOVEDPROSJEKTET:

Øyvind Kvålsvoll
Jan Morten Krakeli

Takk til:

Faglærere Harald Yndestad og Terje Aarseth for veiledning og inspirasjon.

Rolf Henriksen, NTH Institutt for Teknisk Kybernetikk, for kompendiet Multivariable Stokastiske Systemer.

Jens Glad Balchen , en trendsetter og inspirator innen regulering og lay-out.

Stig Ove Johansen, Kværner- Fjellstrand verft, for diverse artikler, teori om hydrofoilfartøyer.

Terje Våge, Avgangsstudent Skip og Offshore MRIH, for velvillig utlån av notater om bølger og foiler.

Medstudenter og ansatte ved MRIH som har hjulpet oss til å se lyset i tunge stunder.

1.2. INNHOLD

1.2.1. Rapportens struktur

Under overskrifter finnes en kort beskrivelse av kapittelets innhold.

Orientering er orienterende stoff.

Innledning beskriver prosjektets bakgrunn, formål og i hovedtrekk hvilke aktiviteter som er utført.

Teori omhandler anvendt teori innen fagdisiplinen reguleringsteknikk/ kybernetikk.

Metode beskriver hvordan oppgaven er løst. Underkapitler for aktivitetene modellering, simuleringsprogram og regulering.

Resultater og diskusjon presenterer simulering av regulert OEV og kommentarer. En diskusjon av erfaringer fra arbeidet med modellering og bruk av simuleringsprogrammet er også med.

Konklusjon oppsummerer de konklusjoner som kan trekkes.

Sammendrag oppsummerer rapportens viktigste momenter.

Detaljerte opplysninger finnes i Appendix- delen. Hvert enkelt appendix har egen innholdsfortegnelse. Appendix- delen er et separat dokument.

1.2.2. Innholdsfortegnelse

Appendix A: MathCad- filer.
Appendix B: MatLab- filer.
Appendix C: Programlisting, OEV- programmet.
Appendix D: Programlisting, KYBER- systemet.
Appendix E: Programlisting, systemkjerne.

1.3. DATAFILER

Liste over datafiler laget i forbindelse med prosjektet. Filene er pakket med PKZIP.

Disk	Filnavn	Format	Beskrivelse
1	BMP.ZIP	Komprimert	Bitmap bilder som er brukt i OEV- programmet.
2	FJEDRING.ZIP	Komprimert	Enkel simulering for test av OEV- programmet.
3	MATH.ZIP	Komprimert	MatLab- og MathCad- filer, samt resultatfiler generert av disse: Matrise i ASCII- format.
3	OEVR.ZIP	Komprimert	Hovedoppgaven som du nå leser; AMIPRO- format.
3	PH.ZIP	Komprimert	Forprosjektet for hovedoppgaven; AMIPRO- format.
4	PRG.ZIP	Komprimert	TurboPascal kildekode for simuleringsprogrammet.
4	EXE.ZIP	Komprimert	Ferdig OEV- program.
4	DATA.ZIP	Komprimert	Datafiler for objekter i OEV- programmet.

Forklaring til filformat i ZIP- filene

*.asm	Assembler kildekode.
*.bmp	Grafikk/ tekst lagret som bitmap.
*.exe	Executable, filer som er direkte kjørbare.
*.imp	Kildekode for implementasjonsdelen til Turbo Pascal UNIT.
*.int	Kildekode for interfaceseksjonen til Turbo Pascal UNIT.
*.log	Resultatet fra en simulering i OEV- programmet, regnearkformat.
*.m	Arbeidsfil for MatLab. ASCII- tekst format.
*.mcd	Arbeidsfil for MathCad.
*.mod	Datafil for objektet 'PROSESS' i OEV- programmet.
*.obj	Kompilert kildekode, MS- DOS objektformat.
*.pas	Kildekode for Turbo Pascal.
*.pif	Programinformasjonsfil for oppstart av program under Windows.
*.prj	Prosjektfil fra Turbo Pascal.
*.prn	ASCII- format fil som inneholder matriser eller simuleringsdata.
*.sam	Dokument fra AmiPro.
*.sdu	Ressursfil for OEV- program med konfigurasjonsdata, bilder, hjelp.
*.tpu	Ferdigkompilert Turbo Pascal UNIT.
*.txt	ASCII- format hjelp- sider.

1.4. TEKNISK MATERIELL

Programvare:

w0 MathCad versjon 5.0 plus.
w1 MatLab versjon 4.0.
w2 Borland Turbo Pascal versjon 7.0.

Maskinvare :

w3 TCI i486DX/33 , MRIH Nr 0063, 20 Mb RAM.

Operativsystem:

w4 MS-DOS versjon 5.0.
w5 Windows versjon 3.0

1.5. TERMINOLOGI

Kort beskrivelse av faguttrykk som forekommer i teksten.

Aerodynamikk: Læren om dynamiske fenomener i luft.

Aksellerometer:Instrument for måling av aksellerasjon.

Assemblerkode:Lavnivå program- kildekode beregnet på en spesiell type CPU.

C_D: Dragkoeffisient.

CG: Center Of Gravity, tyngdepunkt.

C_L: Løftkoeffisienten.

CPU: Control Prosessing Unit, annet navn for mikroprosessor.

Diskretisering: Oppdeling av kontinuerlig signal i punktprøver.

Dopplerskift: Forandring i frekvens pga hastighet.

Ducted fan: Propell for framdrift plassert i en slags kanal.

Dynamisk system: System der tilstander kan endre seg over tid.

Element: Basisfunksjon eller tjeneste i et system som representerer noe fra virkeligheten.

Foil: Forkortet skrivemåte for hydrofoil.

Gyroskop:Instrument for måling av rotasjon.

Hydrodynamikk: Læren om dynamiske fenomener i væsker.

Kavitasjon: Luftbobler som oppstår i flytende medie når en gjenstand beveger seg gjennom mediet med stor fart.

Kildekode: Tekst som beskriver hvordan et dataprogram implementeres etter bestemte regler.

Kompilator : Konverterer kildekode til kjørbart program.

Kybernetikk : Læren om gromme reguleringssystemer.

Lineært system: Et system som er avgrenset rundt et arbeidspunkt slik at det kan gi en forenklet beskrivelse av dynamikken rundt dette arbeidsområdet.

Logmodul : Animert vindu i OEV- programmet med mulighet for lagring av data til regnearkformat. Plotter fortløpende verdi på opptil 8 tilstander.

Modell: En representasjon av virkeligheten.

Objekt : En implementasjon av et element i en datamaskin. Implementasjonen har en innkapsling med lokale data og lokale metoder.

Objektorientert programmering:: Programmeringsmetode som er karakterisert ved modularisering, informasjonsgjemming, oppførsel og instansering.

OEV: Forkortelse for *Overflate Effekt Vinge,* vår norske oversettelse av begrepet *WIG*.

Prosess: En aktivitet som produserer data.

Regulator : Algoritme som styrer en prosess i den retning vi vil.

SBU: Forkortelse for Ser Bra Ut, en typisk ingeniørmetode for korrigering av realisert teori som ikke strekker til, slik at resultatet ser bedre ut.

Signal: data fra en prosess som kan fremstilles som en funksjon.

System: samling av elementer med felles formål som påvirker hverandre gjensidig.

Tilstand: En egenskap på et gitt tidspunkt.

Tilstandsmodell : Bilde av alle tilstander som forekommer .

Transporteffektivitet: Mål for et fartøys kraftbehov i forhold til fart og vekt:

$$\eta_{TRANSPORT} = \frac{Vekt \cdot Hastighet}{Kraftbehov}$$

Ultrasonic transducer: Et instrument som både kan sende og ta imot ultralyd.

VVG: Forkortelse for Vitenskapelige Ville Gjetninger, en ofte forekommende fremgangsmåte for løsning av praktiske problemer når en ikke har teori å støtte seg til.

WIG: Forkortelse for *Wing In Ground- effect*, et fartøy, fortrinnsvis sjøgående, som utnytter overflateeffekten til å få stort aerodynamisk løft.

1.6. REFERANSER

Litteratur og personer som har bidratt til prosjektet.

1.6.1. Litteratur

3. *J. Myers:* Handbook of Ocean and Underwater Engineering, McGraw-Hill, 1969. Teori for sjø, vær, skipsdynamikk.
4. *H. V. Borst*: The Aerodynamics of the Unconventional Air Vehicles of A. Lippisch. Teori for WIG- fartøy.
5. *F. van Walree*: The Powering characteristics of hydrofoil craft. The sixth internationel high speed surface craft conference, 14.- 15. jan. 1988. Teori for hydrofoilfartøy.
6. *Yeluda Manor*: WIGFOIL Interfave Craft Concept. The sixth international high speed surface craft conference, 14.- 15. jan. 1988. Beskrivelse av WIGFOIL fartøy.
7. *Lawrence J. Doctors*: Hydrodynamics of high- speed small craft. The University of Michigan, dept. of Naval Architecture and Marine Engineering. Teori for hydrofoilfartøyer og foiler.
8. Jane's High- Speed Marine Craft 1990: Eksempler på allerede eksisterende hurtiggående fartøy. Foilcat. WIG.
9. *H. O. Peitgen, D. Saupe*: The Science of Fractal Images. Springer Verlag 1988. Støyteori, fraktaler, fraktalalgoritmer.
10. *A. L. Winblad, S. D. Edwards, D. R. King:* Object- Oriented Software. Addison- Wesley 1990.
11. *J. Sanchez, M. P. Canton*: Programming Solutions Handbook for IBM Micro- Computers. McGraw- Hill 1991. Assembler- programmering. Forklaring av begrepet Objekt- Orientert.
12. *R. Duncan*: Advanced MS- DOS Programming, 2. ed. Microsoft Press 1988. Assembler- programmering og referanse for systemkall til DOS.
13. *A. Williams*: DOS 5: A Developer's Guide. Prentice- Hall 1991. Referanse for systemkall til DOS.
14. *C. H. Pappas, W. H. Murray*: Borland C++ Handbook. McGraw- Hill 1991. Oppslagsverk for C++.
15. *H. Schildt*: C++, The Complete Reference. McGraw- Hill 1991. Oppslagsverk for C++.
16. Microsoft MS- DOS 5 Programmers Reference. Microsoft Press 1991.

17. INTEL i486 Microprocessor Programmer's Reference Manual. INTEL- McGraw- Hill 1990. Referanse for assemblerprogrammering.
18. *R. Henriksen, Jens G. Balchen*: Multivariable Stokastiske Systemer. Kompendium, NTH Institutt for Teknisk Kybernetikk 1990.
19. *Jens G. Balchen*: Reguleringsteknikk, bind 2. TAPIR 1982 (1. opplag 1967). Diverse teori for reguleringsteknikk.
20. *Jens G. Balchen*: Reguleringsteknikk, bind 3. TAPIR 1989 (1. opplag 1970). Kalmanfilter, optimalregulering, tilstandsestimering.
21. *H. Yndestad*: Systemteori, del 1.- 6. Kompendier utgitt i forbindelse med faget Reguleringsteknikk 2, MRIH 1993.
22. *O. M. Faltinsen:* Forelesninger i Marin Teknologi om Bølgebelastinger, bølgeinduserte bevegelser og forankring. Kompendium, Institutt for havnebygging, NTH.
23. *S. Falch:* Seakeeping of foilkatamarans. Kompendium, Kværner Båtservice A/S.
24. *R. Johnson, W. O' Neill:* Automatic Control Systems for Hydrofoil craft. Naval Ship Research and Developement Center Maryland, U.S.A.
25. *C. Pieroth, F. Otto:* A Hydrofoil Advanced Tech. LIft and Propulsion System. AAIA 6. Marine Systems Conference.
26. *J. A. Borrie:* Stochastic Systems for Engineers. Prentice Hall 1992.
27. *G. F. Page, J. B. Gomm, D. Williams:* Application of Neural Networks to Modelling and Control. Chapman & Hall 1993.
28. *M.H.Kaplan:* Modern spacecraft dynamics and control.
29. Proceedings of the International Federation of Automatic Control (IFAC), 6th. World Congress, part 4, kap 6.1, 6.2.
30. *S. P. Banks:* Control Systems Engineering. Prentice- Hall.
31. *Ø. Kvålsvoll, J. M. Krakeli:* PROSJEKTHÅNDBOK for hovedoppgaven Modellering, Regulering, Simulering av OEV- fartøy, MRIH 1993/ 1994.
32. *O.A.Olsen*: Instrumenteringsteknikk.
33. *M.H.Kaplan*: Modern spacecraft dynamics and control.
34. Proceedings of the International Federation of Automatic Control (IFAC), 6th. World Congress, part 4, kap 6.1, 6.2.
35. *J.M.Krakeli, Ø.Kvålsvoll, H.O.Sørebø :* OEV- Instrumenter.

1.6.2. Personer

Øyvind Kvålsvoll og Jan Morten Krakeli: Prosjektgruppe.

Harald Yndestad: Faglærer, elektro- automasjon. Veiledning, fortrinnsvis innenfor fagdisiplinen reguleringsteknikk.

Terje Aarseth: Faglærer, elektro- automasjon. Veiledning og bistand innen modellering, regulering og instrumentering.

Arne Jan Sollied: Avd. leder MRIH skip & offshore. Kontaktperson innen maritimt miljø.

2. INNLEDNING

Beskrivelse av prosjektets opprinnelse, formål og hva som skal gjøres.

2.1. Prosjektets bakgrunn

En ville ha en oppgave som omfatter regulering, modellering og simulering, der en får vist anvendelse av moderne reguleringsteori, programmering og simulering. En kom over informasjon om hurtiggående båter basert på vinge- i - overflateeffekt (WIG- fartøy, heretter vil betegnelsen OEV bli brukt). Prinsippet har vært kjent lenge. Et stort problem ved praktisk realisering ser ut til å være stabilitet og manøvrerings-dyktighet. Følgende oppgave ble formulert:

MODELLERING, SIMULERING, REGULERING av hurtig-gående båt for passasjertransport av typen OEV- fartøy.

OEV- konseptet

Hurtiggående båter, fortrinnsvis for persontransport, er sett i flere varianter. De mest aktuelle typene kan grovt sett grupperes i fire kategorier:

w0 Luftputefartøy

w1 Katamaraner

w2 Hydrofoiler

w3 SES (Surface- Effect- Ship)

Felles for alle konseptene er at når hastigheten øker, synker transporteffektiviteten (Vann- motstand er kvadratisk avhengig av fart), fartøyet blir vanskeligere å kontrollere og sjødyktigheten blir dårligere. De beste av disse fartøyene kan holde en fart på opptil 50 knop i nokså urolig sjø

Figur 2.1.: OEV- konseptet

[Foil- Cat, ref. 6.]. Det er interessant å merke seg at Norge er blant de aller fremste når det gjelder teknologi for avanserte båter.

En har sett presentert idèer for konstruksjon av fartøy basert på den såkalte *over-flateeffekten* [WIG, Wing In Ground- effekt, ref. 2., 4., 6.]. Det går ut på at en utformer skroget som en vinge, slik at aerodynamiske krefter løfter hele fartøyet klar av vann-flaten. I tillegg får en løft fra lufttrykk som bygger seg opp under vingen, dette kalles RAM- vinge effekten. Overflateeffekten og RAM- vingeeffekten medfører at luftmot-stand- løftekraft- forholdet blir mye bedre enn for et vanlig fly, og ettersom skroget

ikke er i kontakt med vannet, blir vannmotstand eliminert. Derav kan en dra konklusjonen at dette prinsippet må være egnet for transport over sjø i høy hastighet. Ulempene er manglende manøvreringsdyktighet, ustabilitet ved overgang flytende- luftbåren og problemer med å få til sikker høyderegulering [ref. 1., 4.]. Teorier og praktiske forsøk med prototyper er utført blant annet av tyskeren A. Lippisch tidlig på 70- tallet [ref. 2.]. Hvis en utstyrer dette fartøyet med hydrofoilfinner, kan en kanskje ved hjelp av egnede regulatorer kontrollere ferden [ref. 4.]. Dette ønsker vi å studere nærmere.

Skyvekraft kan fremskaffes fra en gassturbindrevet ducted- fan. Det meste av løftekraften oppnås med skrogets vingefasong, som utnytter overflateeffekten. En tenker seg et foil-system med tre enheter, der alle har en tverrgående finne med justerbar horisontal angrepsvinkel, den bakre finnen skal også ha justerbar vertikal angrepsvinkel for å kunne fungere som sideror. Det kan vises at det eksisterer et optimalt forhold mellom aerodynamisk og hydrodynamisk løft fra foiler for et slikt fartøy [ref. 4.].

Hvis konseptet skal være realistisk til praktisk bruk, må følgende aspekter vurderes:

w4Økonomi: Fartøyet kan ikke være for dyrt å bygge, og drivstofforbruk må være akseptabelt.

w5Komfort: Ved passasjertransport må fartøyets bevegelser kontrolleres slik at krav til komfort tilfredsstilles.

w6Sikkerhet: Høye hastigheter til sjøs medfører store sikkerhetsmessige problemer.

w7Formgivning: Fartøyet må utformes slik at det ikke får uhensiktsmessige fysiske dimensjoner.

Regulering av OEV- fartøyet har direkte innvirkning på sikkerhet og komfort.

2.2. Prosjektets aktiviteter

Prosjektet har som hovedmålsetning å vise anvendelse av moderne reguleringsteori og simulering.

Som nevnt over, er problemet med OEV- fartøyet å holde flyhøyden stabil, og ved overgang mellom flytende og luftbåren tilstand blir fartøyet svært vanskelig å kontrollere.

En skal finne gode reguleringsstrategier for OEV- fartøyet, og simulere det regulerte systemets dynamiske egenskaper. En kommer til å legge mindre vekt på om den matematiske modellen av fartøyets dynamiske egenskaper er helt eksakt, dette er en oppgave for eksperter innen aerodynamikk/ strømningsdynamikk, det vil også kreves forsøk med fysiske modeller for å finne ut mer om fartøyets oppførsel før konstruksjon av fullskala- fartøy. Reguleringsystemet skal derfor lages slik det lett kan modifiseres til å passe en mer realistisk modell eller en annen fartøydesign. En skal komme frem til en egnet total reguleringsstrategi for OEV- fartøy, og simulere dette mest mulig realistisk i et dataprogram.

Reguleringssystemet skal styre foilene slik at fartøyet kan kontrollers manuelt fra et styrekonsoll eller fra en autopilot.

Prosjektets aktiviteter faller naturlig inn i tre kategorier:

w8 Modellering av OEV- fartøy og omgivelser.

w9 Utvikling av simuleringsprogram.

w10Utvikling av regulator for OEV- fartøy, der simuleringsprogrammet brukes som verktøy.

Prosjektets målsetning og aktiviter er beskrevet i detalj i rapporten PROSJEKTHÅNDBOK for HOVEDOPPGAVEN MODELLERING, SIMULERING og REGULERING av OEV- FARTØY [ref. 29].

3. TEORETISK GRUNNLAG

Teoretisk grunnlag for reguleringsteknikk/ kybernetikk anvendt i prosjektet.

3.1. TILSTANDSMODELLERING

Metode for modellering av komplekse dynamiske systemer.

3.1.1. Dynamisk modellering

Teori for dynamisk modellering bygger på kompendiet Systemteori [ref. 19.].
En ønsker å beskrive matematisk et systems oppførsel over tid. Dette gjøres ved å sette opp et sett av første ordens differensialligninger for systemet av typen:

en tilstand derivert m.h.p. tid = en funksjon av tilstander og ytre påvirkning

Dette kan skrives slik:

$$\frac{d}{dt}x_i = f(x_1..x_n, u, t) \qquad \text{[Teori: 1.]}$$

- der x_i, $x_1..x_n$ er en tilstander i systemet, u påvirkning utenfra.

Total energi og masse for et system er konstant. Differensialligninger settes opp ved å ta utgangspunkt i systemets balanse :

Endring = Egenendring + Det som tilføres - Det som taes ut

Balanseligninger kan settes opp ved å ta utgangspunkt i :

w6 Energibalanse
w7 Kraftbalanse
w8 Massebalanse

I et komplekst system med mange tilstander setter en opp balanseligninger for hver enkelt tilstand, slik at hele systemets oppførsel er beskrevet.

3.1.2. Tilstandsmodell

Teori for tilstandsmodellering bygger på kompendiet Systemteori [ref. 19.] og Stochastic Systems for Engineers [ref. 24.].
En matematisk dynamisk modell skal bringes over til formen:

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u}$$
$$\boldsymbol{y} = D\boldsymbol{x} \qquad \text{[Teori: 2.]}$$

- der **x** er tilstandsvektor, **u** er pådragsvektor, **y** er målevektor og A, B, D er systemets matriser.
Dette gjøres ved å ordne alle systemets tilstander og tilhørende differensialligning:

$dx_1/dt=f(x_1..x_n, u_1, u_2.. ,t)$
$dx_2/dt=f(x_1..x_n, u_1, u_2.. , t)$
...

$dx_n/dt=f(x_1..x_n, u_1, u_2.. , t)$

$y_1=f(x_1..x_n,t)$
$y_2=f(x_1..x_n, t)$
...
$y_k=f(x_1..x_n, t)$

Antall tilstander (størrelsen på x- vektoren) angir systemets orden. Matriser A, B, D bestemmes av koeffisientene i systemets differensialligninger. Hvis systemet er ulineært, kan en sette inn elementer på følgende vis:

$$\dot{x}_i = f(x_1..x_n, u_1, u_2.., t) = f(\boldsymbol{x}, \boldsymbol{u}, t) = \sum_{z=1}^{M} f_z(\boldsymbol{x}, \boldsymbol{u}, t)$$ [Teori: 3.]

$$\dot{x}_i = A(\boldsymbol{x})\boldsymbol{x} + B(\boldsymbol{x})\boldsymbol{u}$$ [Teori: 4.]

Dette innebærer en oppdeling av differensialligningen i M funksjoner, her kalt f_z. For hver enkelt f_z implementeres et element enten i A eller B- matrisen:

$$\Rightarrow A_{ij} = \frac{f_j}{x_i}, B_{ij} = \frac{f_j}{x_i}$$ [Teori: 5.]

En slik ulineær representasjon er stort sett brukelig bare til simuleringsformål.

Når systemet er på standard tilstandsform, kan all videre regning foretas på et høyere abstraksjonsnivå med systemets matriser og vektorer. Dette medfører følgende fordeler:

w9 Representasjon med enkle matriseligninger, uavhengig av systemets egentlige kompleksistet.

w10 Identisk form for alle systemer, slik at generell teori for analyse og kontroll kan anvendes.

3.1.3. Linearisering

Mesteparten av den teori som omhandler analyse og metoder for kontroll av tilstandsmodeller gjelder for lineære systemer. Hvis systemet er ulineært, må det foretas en form for linearisering. Dette gjøres ved å ta utgangspunkt i et arbeidspunkt, x_0, og linearisere rundt dette:

$$\dot{\boldsymbol{x}} = \frac{\partial}{\partial \boldsymbol{x}} f(\boldsymbol{x}_0) \cdot (\boldsymbol{x} - \boldsymbol{x}_0) = A \cdot \boldsymbol{x}_{lin}$$ [Teori: 6.]

$$A_{ij} = \frac{\partial}{\partial x_i} f(\boldsymbol{x}_0)$$ [Teori: 7.]

Dette gir elementer for systemets matriser. Det er viktig å merke seg at tilstandsvektoren er forskjøvet: $\mathbf{x}_{lin}=\mathbf{x}-\mathbf{x}_0$. Dette må en ta hensyn til hvis lineariserte modeller anvendes i reguleringsalgoritmer og estimatorer. En slik linearisert modell vil som regel gi en god beskrivelse av systemet dynamikk i et snevert område rundt arbeidspunktet. Hvor anvendelig modellen er når tilstander forflyttes ut fra arbeidspunktet avhenger av hva slags ulineariteter en har med å gjøre.

3.1.4. Modal analyse

Et systems grunnleggende egenskaper kan analyseres ved å bringe det over på modal form med lineærtransformasjonen:

$$\mathbf{x} = M\mathbf{q}, \qquad \text{[Teori: 8.]}$$
$$\mathbf{y} = DM\mathbf{q} = F\mathbf{q}$$

der M er systemets modalmatrise gitt av egenvektorer til systemet. Systemet blir da på diagonalform:

$$d\mathbf{q}/dt = \Lambda\mathbf{q} + H\mathbf{u} \qquad \text{[Teori: 9.]}$$
$$\mathbf{y} = F\mathbf{q}\ ,\ F = DM$$

-der Λ=diagonalmatrise med systemets egenverdier, $H = M^{-1}B$.

Systemets dynamiske egenskaper kan finnes ved å analysere egenverdiene. Hvis en eller flere egenverdier er >0 , er systemet utstabilt, det vil si at en eller flere av tilstandene vokser over alle grenser når $t \rightarrow \infty$.

At et system er observerbart betyr at målevektoren y inneholder informasjon om alle tilstander, selv om rang D<n. Systemet er observerbart hvis matrisen F ikke har kolonner med bare null- elementer.

At et system er styrbart betyr at en via pådragsvektoren kan bringe en hvilken som helst tilstand til hvilken som helst verdi, selv om en ikke har tilgang til å gi pådrag direkte til alle variable. Et system er styrbart hvis ingen rekker i H inneholder bare null-elementer.

3.1.5. Diskretisering

Diskretisering er nødvendig hvis signalbehandling skal foretas i en datamaskin. Formålet kan være simulering av systemet eller algoritmer for regulatorer som skal implementeres. Diskretisering av tilstandsmodeller innebærer at en innfører nye systemmatriser, slik at en får representasjonen:

$$x(n+1) = \Phi x(n) + \Delta u,\ y(n) = Dx(n) \qquad \text{[Teori: 10.]}$$

Dette kan gjøres på forskjellige måter. En metode som ikke krever mye regnekraft er Euler diskretisering:

Innfører:

$$\frac{d\mathbf{x}}{dt} \approx \frac{\mathbf{x}(n+1) - \mathbf{x}(n)}{T} \qquad \text{[Teori: 11.]}$$

i systemet

$$d\mathbf{x}/dt = A\mathbf{x} + B\mathbf{u}$$

som gir:

$$\mathbf{x}(n+1) = (I+AT)\mathbf{x}(n) + BT\mathbf{u}(n) \Rightarrow \Phi = I + AT,\ \Delta = BT. \qquad \text{[Teori: 12.]}$$

- der T angir samplingsintervallet, $T = 1/f_s$.

En metode som gir bedre nøyaktighet er å ta utgangspunkt i eksponentialrekkeutvikling:

$$\Phi = e^{AT} = I + AT + (1/2!)A^2T^2 + \ldots + (1/N!)A^NT^N \qquad \text{[Teori: 13.]}$$

Matriser for kobling fra pådrag og støy (Støy: se ligning 28.) diskretiseres slik:

$\Delta=A^{-1}(e^{AT}-I)B=IBT+(1/2!)ABT^2+..+(1/N!)A^{(N-1)}BT^N$ [Teori: 14.]

Antall ledd velges ut i fra krav til nøyaktighet og tilgjengelig regnekraft.
Valg av samplingsfrekvens for diskrete systemer bestemmes ut i fra systemets egenverdier, krav til nøyaktighet og tilgjengelig regnekraft. Minstekrav er samplingsfrekvens større enn 5 ganger frekvens som tilsvarer raskeste egenverdi.

3.1.6. Simulering av ulineært system

Ligninger 3. - 5. beskriver metode for å sette opp en ulineær modell på formen:

$$\dot{x} = f(x,u,v) = A(x)\cdot x + B(x)\cdot u + C(x)\cdot v \qquad \text{[Teori: 15.]}$$

En betrakter A- matrisen. Euler diskretisering gir:

$$x(n+1) = \Phi(x)x(n)\ ,\ \Phi(x) = I + A(x)\cdot T \qquad \text{[Teori: 16.]}$$

Hvis samplingfrekvens velges tilstrekkelig høy slik at:

$$x(n+1) \approx x(n) \qquad \text{[Teori: 17.]}$$

- kan følgende antas:

$$\Phi(n+1) \approx \Phi(n) \Rightarrow \Phi(x) \approx \Phi(n) \qquad \text{[Teori: 18.]}$$

Ligning 16. gir umiddelbart:

$$\Phi(n) = I + A(n)\cdot T, A(n) = \frac{f(x)}{x(n)} \qquad \text{[Teori: 19.]}$$

Diskret tilnærming for ligning 15. blir da:

$$x(n+1) \approx \Phi(n)\cdot x(n), \Phi(n) = I + A(n)\cdot T \qquad \text{[Teori: 20.]}$$

En ser at simulering kan foretas ved for hvert sample først å oppdatere ulineære elementer i matrisene A, B, C, deretter foreta en diskretisering av matrisene, og til slutt beregne **x**(n+1).

3.2. STOKASTISKE SYSTEMER

Hvordan beskrive støy.

3.2.1. Stokastiske signaler

Tilfeldige, uforutsigbare hendelser over tid generer et signal som kalles stokastisk prosess. Karakteristisk for disse er at fremtidige verdier aldri kan forutsies eksakt, og at de beskrives med statistiske metoder.

3.2.2. Wiener- prosesser

Påvirkning fra omgivelser har ofte en form for tilfeldig natur. I 1827 oppdaget botanikeren R. Brown at små partikler i en væske beveget seg på en uforutsigbar og irregulær måte da han satt og så på dem gjennom et mikroskop. Denne observasjonen er opphavet til begrepet *Brownske prosesser*, eller *Wiener- prosess* som den også kalles. Mønsteret kan beskrives som en integrasjon av normalfordelte tilfeldige tall. Senere er det påvist at tilfeldig støy følger dette mønsteret, enten det er snakk om støy i elektroniske kretser eller andre fysiske fenomener.

3.2.3. Hvit støy

Hvis et signal genereres fra en tallgenerator som returnerer normalfordelte tilfeldige tall slik at:

$w(n)=N(0,1)$

kalles signalet diskret hvit Gauss- støy. En kontinuerlig modell for hvit støy kan beskrives ved autokorrelasjon

$r_w=q\delta\tau$ [Teori: 21.]

der q angir støyens styrke, δ er en Dirac delta- funksjon. En ser da at kontinuerlig hvit støy egentlig er en matematisk fiksjon som ikke kan realiseres.

3.2.4. Fraktaler og støy

Brownske prosesser kan ikke beskrives med en analytisk ligning, men en kan lett implementere mønsteret i en rekursiv algoritme, da kaller en mønsteret *fraktal*.

Begrepet *fraktaldimensjon*, D=E-H, sier noe om hvordan spektralfordelingen for mønsteret er. E er systemets euklidiske dimensjon (E=1 for linje, 2 for flater, osv.), og H={0..1} angir utseende. Hvit støy har lineær spektralfordeling, og er høyst irregulær og tilfeldig av utseende, faktoren H=0. En ren integrasjon av hvit støy gir H=0.5. Et landskap har et mer jevnt utseende, typiske verdier for H kan være 0.7- 0.9. For å finne en simuleringsmodell for et støysignal, kan en ta en testmåling og prøve seg fram med forskjellige verdier for H til modellen har omtrent samme utseende som den virkelige støyen. En modell for støy kan være en diskret Gauss- generator som leverer tall N(0,1), som kan integreres/ filtreres til signalet får ønsket fasong.

3.2.5. Effektspekter

En metode for beskrivelse av en stokastisk stasjonær prosess er å betrakte signalets effektfordeling som funksjon av frekvens. Effektspekteret (ofte betegnet som *power spektral density* i litteraturen), S(ω), defineres slik:

$$S(\omega) = \mathcal{F}(r(\tau)) = \int_{-\infty}^{+\infty} r(\tau)e^{-j\omega\tau}d\tau \qquad \text{[Teori: 22.]}$$

- der r(τ) er signalets autokorrelasjon.

Hvit støy har konstant spektralfordeling, det vil si at alle frekvenser er likt representert i signalet. Brownske prosesser har spektralfordeling $S_n \neq$ konstant, og kan betraktes som hvit støy filtrert gjennom et lavpassfilter. Stokastiske prosesser kan også være periodiske, i den forstand at signalets frekvensspekter har en resonanstopp. Bølger på havet er et eksempel på en slik prosess.

Hvis en kjenner effektspekteret for et stokastisk signal $S_n(\omega)$, kan prosessen modelleres ved å filtrere hvit støy med et filter med fouriertransform- funksjon $F(j\omega)$:

$$S_n(\omega)=F(j\omega)F(-j\omega)S_f(\omega) \qquad \text{[Teori: 23.]}$$

Hvis f(t) er gauss- støy med spektraltetthet 1, får en:

$$S_n(\omega)=F(j\omega)F(-j\omega) \qquad \text{[Teori: 24.]}$$

Overføringsfunksjonen for filteret, $F(j\omega)$, finnes ved å faktorisere signalets effektspekterfunksjon til produktet av $F(j\omega)$, $F(-j\omega)$.

3.2.6. Tilstandsmodell med prosesstøy og målestøy

En tilstandsmodell for et system påvirket av prosesstøy og hvit målestøy beskrives slik:

$$\mathbf{dx}/dt=f(\mathbf{x}(t),\mathbf{u}(t))+C(t)\mathbf{v}(t) \qquad \text{[Teori: 25.]}$$
$$\mathbf{y}=g(\mathbf{x}(t),t)+\mathbf{w}(t)$$

Denne representasjonen er ikke matematisk holdbar, fordi støyen ikke er deriverbar. En alternativ fremstilling er:

$$d\mathbf{x}(t)=f(\mathbf{x}(t),\mathbf{u}(t),t)dt+C(t)d\boldsymbol{\beta}(t) \qquad \text{[Teori: 26.]}$$
$$d\mathbf{z}(t)=g(\mathbf{x}(t),t)+d\boldsymbol{\lambda}(t)\ ,\ \mathbf{y}(t)=d\mathbf{z}(t)/dt$$

-der $\boldsymbol{\lambda}$, $\boldsymbol{\beta}$ er Wiener- prosesser. Hvit støy kan betraktes som den deriverte av slike prosesser, slik at **v** og **w** kan formuleres slik:

$$\mathbf{v}=d\boldsymbol{\beta}(t)/dt\ ,\ \mathbf{w}=d\boldsymbol{\lambda}(t)/dt \qquad \text{[Teori: 27.]}$$

Selv om ligning 25. ikke er holdbar matematisk sett, brukes ofte denne formen som representasjonen for prosesser utstatt for stokastiske forstyrrelser. Hvis systemet er lineært og systemmatrisene er konstante fåes følgende form:

Tilstandsmodell, analog representasjon:

$$\mathbf{dx}/dt=A\mathbf{x}+B\mathbf{u}+C\mathbf{v} \qquad \text{[Teori: 28.]}$$
$$y=D\mathbf{x}+E\mathbf{w}$$

Hvis **w** er hvit Gauss- støy med vekt 1, angir matrisen E målestøyens styrke. Denne modellen er utgangspunktet for videre behandling av kontinuerlige stokastiske systemer.

Diskret form av ligning 25. er matematisk gyldig og beskrives slik:

Tilstandsmodell, diskret representasjon:

$$x(n+1)=\Phi x(n)+\Delta u(n)+\Omega v(n), y(n)=Dx(n)+Ew(n) \qquad \text{[Teori: 29.]}$$

- der **v**(n), **w**(n) er diskret hvit Gauss- støy.

Hvis prosesstøyen **v** ikke er hvit støy, men har spektralfordeling $S_v(\omega) \neq$ konstant, kan en likevel i mange tilfeller betrakte systemet som om denne var en hvit prosess. Da utnyttes imidlertid ikke all tilgjengelig informasjon om systemet. Mens hvit støy er fullstendig upredikterbar, kan en nå si noe om fremtidige verdier for **v**(t). Et system som skal reguleres med prediksjon av fremtidige verdier for prosesstøy blir naturlig nok mer komplekst, derfor bør en vurdere hvor vidt dette er nødvendig. Hvis ønsket resultat kan oppnås uten prediksjon av **v**, er den enklere løsningen å foretrekke.

3.3. TILSTANDSESTIMERING

Metode for å finne informasjon om tilstander en ikke kan måle direkte.

3.3.1. Tilstandsestimator

I praktiske systemer vil det ofte være slik at en ikke har tilgang til måling av alle tilstandsvariable. Hvis systemet er observerbart kan en estimere disse med en tilstandsestimator. Denne bruker en modell av prosessen for å beregner et estimat av tilstandsvektoren på grunnlag av de målinger som er tilgjengelig og pådragsvektoren. Systemet kan beskrives slik:

$$\frac{d}{dt}\hat{\boldsymbol{x}} = A\hat{\boldsymbol{x}} + B\boldsymbol{u} + K(\boldsymbol{y} - D\hat{\boldsymbol{x}}) = (A - KD)\hat{\boldsymbol{x}} + K\boldsymbol{y} + B\boldsymbol{u} \quad \text{[Teori: 30.]}$$

En ser at innsvingningsforløpet for estimatet er bestemt av matrisen (A-KD). Matrisen K kan bestemmes ved å velge egenverdier for estimatoren, og regne seg fram til K.

3.3.2. Optimal tilstandsestimator: Kalman- filter

Et Kalman- filter er en tilstandsestimator der tilbakekoblingsmatrisen K er bestemt slik at estimatet for **x** alltid vil være optimalt når prosessen utsettes for stokastisk støypåvirkning og målingen inneholder målestøy.

I kompendiet Multivariable Stokastiske Systemer [ref. 16.] beskrives Kalman- filter for kontinuerlige lineære systemer. Prosessmodellen har formen (ligning 28.):

$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} + C\mathbf{v}$
$\mathbf{y} = D\mathbf{x} + E\mathbf{w}$

Støysignalene **v**, **w** er gjensidig ukorrelerte stokastiske støy- prosesser med kovariansmatriser gitt ved:

$E\{\mathbf{v}(t)\mathbf{v}^T(\tau)\} = V(t)\delta(t-\tau)$, $E\{\mathbf{w}(t)\mathbf{w}^T(\tau)\} = W(t)\delta(t-\tau)$ [Teori: 31.]

Estimatoren har formen:

[Teori: 32.]

$$\frac{d}{dt}\hat{\boldsymbol{x}} = A\hat{\boldsymbol{x}} + B\boldsymbol{u} + K(t)(\boldsymbol{y} - D\hat{\boldsymbol{x}}) = (A - K(t)D)\hat{\boldsymbol{x}} + K(t)\boldsymbol{y} + B\boldsymbol{u}$$

Statistisk informasjon om systemet er grunnlaget for beregning av den optimale K-matrisen. Dette gjøres ved å oppdatere en kovariansmatrise, X, på grunnlag av kovariansmatrisene V(t), W(t):

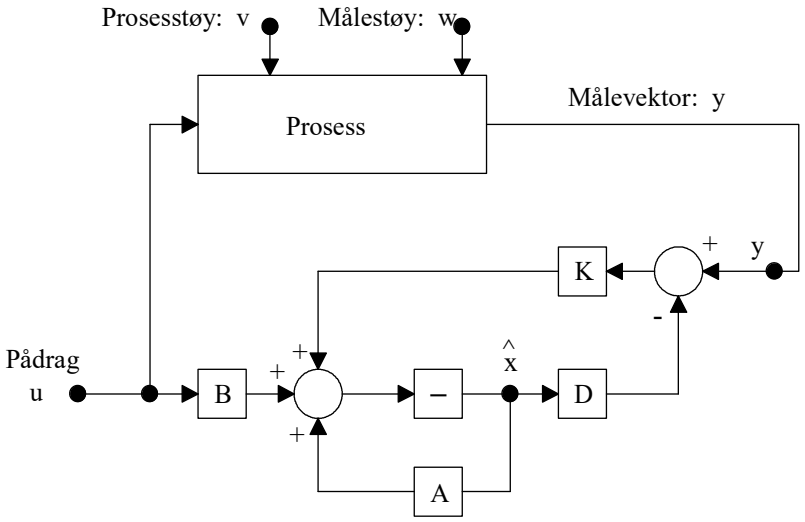
$$\frac{dX}{dt} = AX(t) + X(t)A^T + CV(t)C^T - X(t)D^T W(t)^{-1} DX(t) \quad \text{[Teori: 33.]}$$

Dette er en såkalt matrise- Riccati ligning. Denne kan enten oppdateres kontinuerlig eller en kan beregne X på forhånd slik at K blir konstant. Da er X gitt av ligning 33. der dX/dt→0 når t→∞, dette gir optimal K hvis prosessen er stokastisk stasjonær. K er gitt av ligningen:

$$K(t) = X(t)D^T W(t)^{-1}$$ [Teori: 34.]

Dette gir den optimale K hvis verdier for kovariansmatrisene V, W og modellmatrisene A, B, C, D beskriver prosessen eksakt. Ofte har en med ulineære systemer å gjøre, og det er ikke alltid like lett å bestemme kovariansmatrisene V, W. Da kan en simulere systemet for å kontrollere at estimatoren fungerer slik en ønsker, og prøve seg fram med forskjellige verdier i støymatrisene. Større verdier i W- matrisen gir bedre filtrering og dårligere følging, mens større verdier i V- matrisen gir bedre følging.

Figur 3.3.2.: Tilstandsestimator



3.3.3. Diskret tilstandsestimator

Kalman- estimatorer for diskrete systemer er ofte basert på best mulig utnyttelse av siste måling, med oppdeling av tilstandsvektoren i et aprioriestimat og et aposteriories-timat. Her ønskes imidlertid en representasjon der en kan bruke matriser og modell-struktur fra kontinuerlig tilstandsmodell direkte, derfor tar en utgangspunkt i Kalman-estimator for kontinuerlig tid. Euler diskretisering av estimatoren gir:

$$\boldsymbol{x}(n+1) = (A - KD)T\boldsymbol{x}(n) + BT\boldsymbol{u}(n) + \boldsymbol{x}(n$$ [Teori: 35.]

Innfører notasjon for diskret form. Diskret K- matrise betegnes gK.

$$\boldsymbol{x}(n+1) = \Phi\boldsymbol{x}(n) - gK \cdot D \cdot \boldsymbol{x}(n) + gK \cdot \boldsymbol{y}(n) + \Delta\boldsymbol{u}(n$$ [Teori: 36.]

$$\Phi = I + A \cdot T, gK = K \cdot T, \Delta = B \cdot T$$ [Teori: 37.]

Euler diskretisering av Riccati- ligning for beregning av **X** gir:

[Teori: 38.]

$$X(n+1) = [AX(n) + X(n)A^T + CVC^T - X(n)D^T W^{-1} DX(n)] \cdot T + X(n)$$

$$K(n) = X(n) \cdot D^T \cdot W^{-1}$$ [Teori: 39.]

Kalman- estimatoren kan realiseres ved å implementere ligningene 35., 38., 39. i en datamaskin.

3.4. OPTIMALREGULERING

Metode for regulering av dynamiske systemer.

3.4.1. Klassisk optimalregulering

Det finnes mye litteratur som omhandler klassisk optimalregulering. I kompendiet Multivariable Stokastiske Systemer [ref. 16.] beskrives kort prinsippet som utgangspunkt for regulering av stokastiske systemer.

En prosess er beskrevet av:

$$\mathbf{dx}/dt=A\mathbf{x}+B\mathbf{u} \qquad \text{[Teori: 40.]}$$
$$\mathbf{y}=\mathbf{x}$$
$$\mathbf{x}(t_0)=\mathbf{x}_0$$

En ønsker å regulere denne slik at objektfunksjonalen:

$$J=\int_{t_0}^{t_1}[\boldsymbol{x}^T(t)\cdot Q\cdot\boldsymbol{x}(t)+\boldsymbol{u}^T(t)\cdot P\cdot\boldsymbol{u}(t)]dt \qquad \text{[Teori: 41.]}$$

antar et minimum. Dette oppnås ved å gi et pådrag, $\mathbf{u}(t)$, slik:

$$\mathbf{u}(t)=-P^{-1}B^TR(t)\mathbf{x}(t) \qquad \text{[Teori: 42.]}$$

der matrisen R(t) er løsningen av Riccati- ligningen:

$$dR(t)/dt=-R(t)A-A^TR(t)+R(t)BP^{-1}B^TR(t)-Q \qquad \text{[Teori: 43.]}$$

Matrisene Q og P er absolutt positive matriser, som bestemmer optimalkriteriet for reguleringen. Q betegnes som optimalvekt, denne bestemmer hvilke tilstander det er viktigst å holde under kontroll. P betegnes som kostnadsvekt, denne bestemmer hvilke pådrag som må begrenses.

Det finnes ingen enkel metode for valg av matrisene Q og P. Som utgangspunkt kan en velge diagonalmatriser, der diagonalelementene angir vekt for tilhørende tilstand og pådragsvariabel. Større verdi for en tilstand i matrisen Q medfører raskere konvergens for denne. Større verdi for en pådragsvariabel i P medfører mindre utslag i pådraget til denne. En kan benytte simuleringer som verktøy for å finne passende optimalvekter. Analyse av det regulerte systemets egenverdier gir også god indikasjon på hvorvidt en er på rett vei.

Systemet vil konvergere mot $\mathbf{x}=0$ når $t\rightarrow\infty$. Fordi all informasjon om systemet er kjent, kan pådraget beregnes på forhånd. En alternativ måte er å implementere pådraget som en tilbakekobling fra tilstandsvektoren. En merker seg at optimalregulering via en tilbakekoblingssløyfe krever måling av alle tilstandsvariable.

At systemet er regulert optimalt behøver ikke å bety at reguleringen er ideell. Begrepet optimalt betyr her at en har den best mulige løsning for de optimalvekter, Q og P, som er valgt, under forutsetning at matrisene A og B beskriver prosessen eksakt.

3.4.2. Optimalregulering av stokastiske systemer

Optimalregulering av systemer med stokastisk påvirkning leder til løsninger av samme type som i det deterministiske tilfellet, med den forskjell at nå må regulatoren implementeres som en tilbakekobling fra tilstandsvektoren.

En prosess:

$$\mathbf{dx}/dt=A\mathbf{x}+B\mathbf{u}+C\mathbf{v} \qquad \text{[Teori: 44.]}$$

$\mathbf{y}=\mathbf{x}$

der **v** er en hvit- støy prosess, skal reguleres slik at objektfunksjonalen (ligning 41.)

$$J = \int_{t_0}^{t_1} [\boldsymbol{x}^T(t) \cdot Q \cdot \boldsymbol{x}(t) + \boldsymbol{u}^T(t) \cdot P \cdot \boldsymbol{u}(t)]dt$$

antar et minimum. Dette gir samme løsning som i det deterministiske tilfellet (ligning 42.):

$$\mathbf{u}(t)=-P^{-1}B^TR(t)\mathbf{x}(t)$$

Forklaringen er at **v**(t) er upredikterbar, slik at det beste estimat en har for denne er å sette **v**=0.

R(t) er gitt av ligning 43. Hvis øvre integrasjonsgrense i integralet for J går mot ∞ får en en stasjonær løsning for R. Denne kan beregnes av:

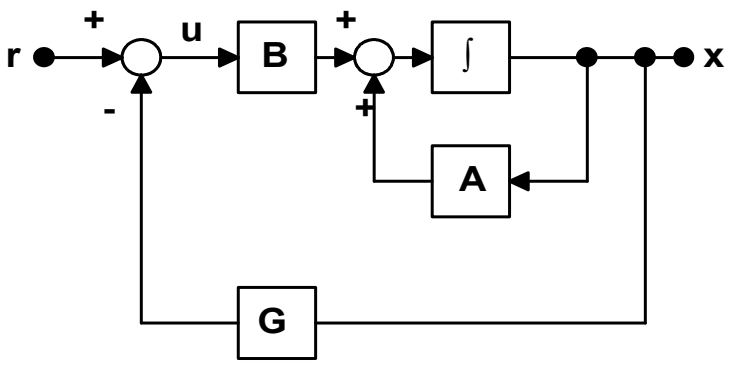
$$dR(t)/dt = -R(t)A-A^TR(t)+R(t)BP^{-1}B^TR(t)-Q = 0 \qquad \text{[Teori: 46.]}$$

Det er oftest denne løsningen som er aktuell for regulering av stokastiske systemer, der formålet som regel er å holde tilstander i en prosess så nær en referanse som mulig når systemet utsettes for forstyrrelser. Da gir den stasjonære løsningen av R en fast tilbakekoblingsmatrise.

Regulatoren må realiseres som en tilbakekoblingssløyfe:

$$\mathbf{u} = -G\mathbf{x}\ ,\ \ G = P^{-1}B^TR \qquad \text{[Teori: 47.]}$$

Figur 3.4.2.: Optimal regulator



3.4.3. Optimalregulering med måling via optimal tilstandsestimator.

Hvis prosessen ikke gir mulighet for ideell måling, det vil si en har målestøy og tilgang til bare et begrenset antall målinger, blir det mer interessant. Teori for dette beskrives inngående i kompendiet Multivariable Stokastiske Systemer [ref. 16.].

En prosess (ligning 28.):

$$d\mathbf{x}/dt=A\mathbf{x}+B\mathbf{u}+C\mathbf{v}$$
$$\mathbf{y}=D\mathbf{x}+E\mathbf{w}$$

der **v**, **w** er gjensidig ukorrelerte hvit- støy prosesser og rang D < n , skal reguleres slik at objektfunksjonalen (ligning 41.)

$$J = \int_{t_0}^{t_1} [\boldsymbol{x}^T(t) \cdot Q \cdot \boldsymbol{x}(t) + \boldsymbol{u}^T(t) \cdot P \cdot \boldsymbol{u}(t)]dt$$

antar et minimum. Dette gir løsningen:

$$\mathbf{u}(t)=-P^{-1}B^TR(t)\mathbf{x}(t) \qquad \text{[Teori: 48.]}$$

der **x** er det optimale estimat for tilstandsvektoren **x**, gitt av Kalman- filteret for systemet. Reguleringen implementeres på samme måte som i punkt 3.4.2., med den forskjell at tilbakekoblingen hentes fra tilstandestimatoren. Matrisene R(t), G er gitt av ligninger 46., 47.

Separasjonsteoremet

En har erstattet tilstandsvektoren **x** med det optimale estimat for denne, **x**. At dette virkelig gir den optimale løsning bevises gjennom separasjonesteoremet:

Separasjonsteoremet, kontinuerlig tid:

*Optimalregulering av et lineært stokastisk system med hvit prosess- og observasjonsstøy fremkommer ved lineær tilbakekobling fra det optimale tilstandsestimatet **x** som genereres av et Kalman- filter. Det totale reguleringssystemet består av to separate funksjoner:*

1. *Optimal estimering av **x** i et Kalman- filter.*

2. *Optimal tilbakekobling fra **x** som om denne var identisk med tilstanden **x** i et tilsvarende deterministisk optimaliseringsproblem.*

For en detaljert gjennomgang av dette henvises til kompendiet Multivariable Stokastiske Systemer [ref. 16.].

3.4.4. Diskretisering av regulert system

En har sett at diskretisering av systemer på tilstands- form leder til løsninger der matriser som inngår i modellen endres. For en enkel reguleringssløyfe av typen:

$$d\mathbf{x}/dt=A\mathbf{x}-BG\mathbf{x}=(A-BG)\mathbf{x} \qquad \text{[Teori: 49.]}$$

får en , med Euler diskretisering:

$$\mathbf{x}(n+1)=(I+AT)\mathbf{x}(n)-BT(G\mathbf{x}(n)) \qquad \text{[Teori: 50.]}$$

$$\mathbf{x}(n+1)=\Phi\mathbf{x}(n)+\Delta\mathbf{u}(n)\ ,\ \mathbf{u}(n)=-G\mathbf{x}(n) \qquad \text{[Teori: 51.]}$$

En ser at tilbakekoblingsmatrisen G kan nyttes direkte i det diskrete systemet.

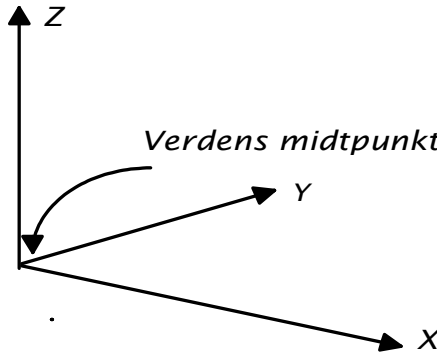
4. METODE

Hvordan prosjektets aktiviteter er utført.

4.1. KOORDINATSYSTEMER

En definerer to koordinatsystemer, ett inertial- referansesystem og ett som refererer til OEV- fartøyet.

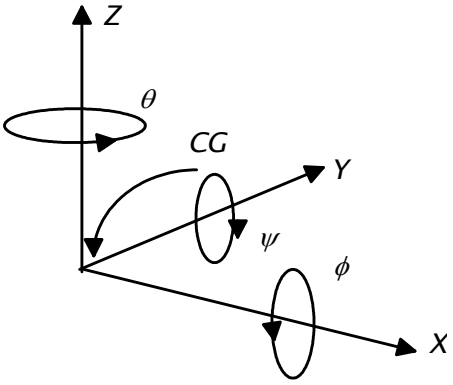
Figur 4.1.1.: Referansesystemets koordinatakser (verden):



Verdens koordinatsystem brukes for å bestemme fartøyets orientering i rommet i z-retning og fartøyets rotasjon i forhold til omgivelsene.

Vind genereres med referansretninger i forhold til verden.
Bølger genereres med referansretninger i forhold til verden.

Figur 4.1.2.: Referanseakser for OEV- fartøyet:



x- akse peker forover.
z- akse peker oppover.
y- akse peker i sideretning, mot babord (venstre).

ϕ: Fartøyets rotasjon rundt x- aksen: Dreiningsvinkel fra verdens xz- plan til fartøyets xz- plan.
ψ: Fartøyets rotasjon rundt y- aksen: Dreiningsvinkel fra verdens xy- plan til fartøyets xy- plan.
θ: Fartøyets rotasjon rundt z- aksen: Dreiningsvinkel fra verdens zy- plan til fartøyets zy- plan.

4.2. MODELLERING AV OMGIVELSER

4.2.1. Vind

En stokastisk vindmodell.

Betraktninger om vind og vindens natur bygger på Handbook of Ocean and Underwater Engineering [ref. 1] og The Science of Fractal Images [ref. 7.].

Vind varierer sterkt som funksjon av tid og sted. En ønsker en vindmodell som angir vindens hastighetskomponenter i global x- og y- retning, slik at aerodynamiske krefters innvirkning på OEV- fartøyet kan finnes ved å addere vindens hastighetsvektor med fartøyets hastighetsvektor. En gjør følgende forenklinger:

w11 Antar at vinden ikke endrer seg over OEV- fartøyets eksitasjonsområde, hvilket medfører at en kan anta samme vind over hele verden.

w12 Middelverdi, retning antas parallell med global x- akse. Dette forenkler modelleringen, og har ingen praktisk betydning for simuleringen da fartøyet skal kunne svinge.

w13 Vindens hastighet i z- retning antas lik 0. Det viser seg i praksis at vindens hastighet i z- retning er svært lav nær overflaten.

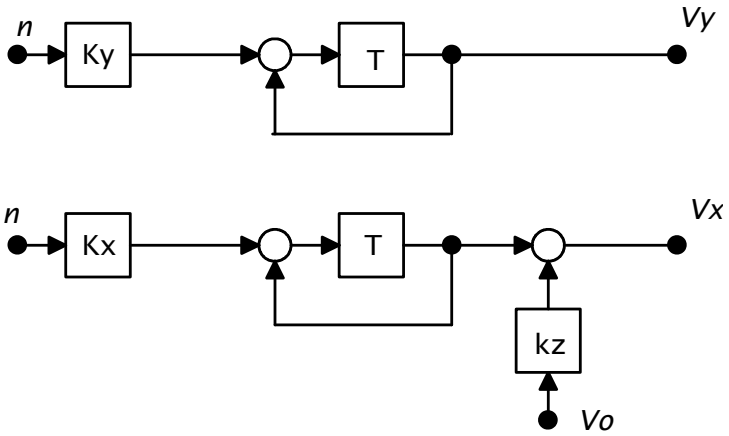
Vindens statistiske natur ligner på såkalt simpel Brownian motion, slik at en kan modellere vinden ved å addere integrert hvit støy til vindens middelverdi. En legger til støy både i x- og y- retning, slik at både amplitude og retning for vinden varierer over tid.

Middelverdi, vindstyrke angis vanligvis i 10 meters høyde. Følgende forhold gjelder for omregning til en gitt høyde over overflaten:

$$\frac{v(z)}{v(10)} = \left(\frac{z}{10}\right)^{\frac{1}{7}}$$ [Vind: 1.]

En ser at vindstyrken varierer i z- retning. En velger å se bort i fra dette, og regner vindstyrken om til en passende høyde, 3 m. over havflaten.

Følgende modell kan nå settes opp:



I simuleringsprogrammet skal en kunne endre følgende parametre:

w14 Middelverdi for vindstyrke, Vo.

w15 Faktor for forstyrrelse i x- retning, Kx. (Angir standardavvik).

w16 Faktor for forstyrrelse i y- retning, Ky. (Angir standardavvik).

Modellen simuleres i MathCad [Fil: VIND.MCD, Appendix A].

I simuleringsprogrammet har modulen som genererer vind tilgang til fartøyets tilstandsvektor. Da kan vindvektoren transformeres til fartøyets koordinatakser slik:

$$Vx = Vx_G \cdot \cos\theta + Vy_G \cdot \sin\theta \qquad \text{[Vind: 2.]}$$

$$Vy = Vx_G \cdot \sin\theta + Vy_G \cdot \cos\theta \qquad \text{[Vind: 3.]}$$

Vx, Vy er da vind som virker på fartøyet i henholdsvis lengde- og sideretning. Vx_G., Vx_Y er vinden generert i modellen med verdens koordinatsystem som referanse.

4.2.2. Bølger

Beskrivelse av en stokastisk bølgemodell, fra teori til implementering i simuleringsprogram.

Teori for bølger er hentet fra Handbook of Ocean and Underwater Engineering [ref.1] og kompendiet Forelesninger i Marin Teknologi om Bølgebelastinger [ref. 20].

Bølger på sjøen består av et komplekst sammensatt bølgespekter, der hver enkelt komponent har sin bølgeformhastighet, bølgehøyde og retning. Av bølgegenererende fenomener kan nevnes vind, båter i fart, undersjøiske jordskjelv.

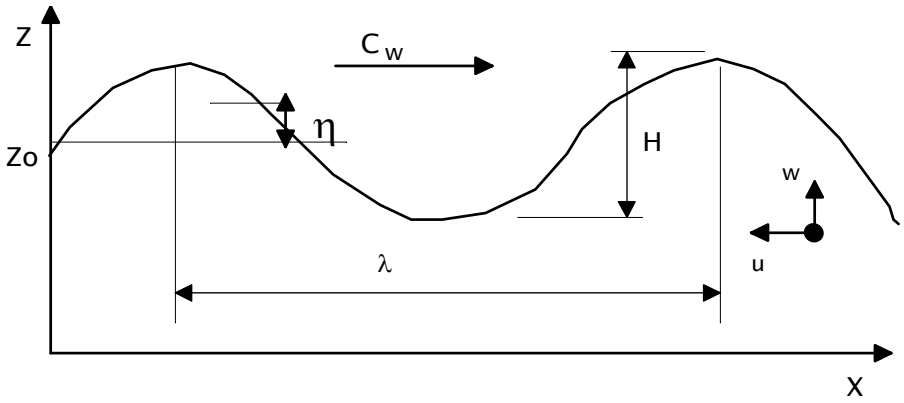
Vi vil begrense oss til å studere frie bølger generert av vind. Videre betraktes sjøen som uendelig dyp, det er en bra tilnærmelse når bølgelendgen er mindre enn halve vanndybden.

Airy's bølgeteori

Følgende betraktninger gjelder for en enkelt bølgekomponent:

w17 Overflategrensen er sinusformet
w18 Forholdet mellom bølgehøyde og bølgelengde er mindre enn 0.1.
w19 Når bølgelengden er gitt, er også bølgeformhastighet og periodetid bestemt.
w20 Vannstrøm under overflaten kan uttrykkes som en hastighetsvektor.

Bølgekomponent med $\delta\Phi/\delta z=0$ (beveger seg i x- retning), ved et gitt tidspunkt:



H= bølgehøyde
λ= bølgelengde
c_w= bølgeformhastighet
η= overflategrense
τ= periodetid

$$\eta(x,t) = A \cdot \sin(\omega t - kx), \omega = \frac{2\pi}{\tau}, k = \frac{2\pi}{\lambda} \qquad \text{[Bølger: 1.]}$$

Sammenhenger mellom λ, c_w, τ:

$$\lambda = 1.56 \cdot \tau^2 \qquad \text{[Bølger: 2.]}$$

$$c_w = 1.56 \cdot \tau \qquad \text{[Bølger: 3.]}$$

Hastighetspotensiale for bølge som forplanter seg i positiv x- retning:

$$\varphi = c \cdot A \cdot e^{k \cdot z} \cdot \cos(\omega t - kx), k = \frac{2\pi}{\lambda} \qquad \text{[Bølger: 4.]}$$

Hastighetskomponenter under overflaten:

$$u = \frac{\partial \varphi}{\partial x} = \omega \cdot A \cdot e^{k \cdot z} \cdot \sin(\omega t - kx) \qquad \text{[Bølger: 5.]}$$

$$w = \frac{\partial \varphi}{\partial z} = \omega \cdot A \cdot e^{k \cdot z} \cdot \cos(\omega t - kx) \qquad \text{[Bølger: 6.]}$$

Studier av hastighetsvektor under overflaten ved hjelp av programvaren MathCad viser at det er ønskelig å få med innvirkning fra bølger ved modellering av nedsenkede foiler. Da blir imidlertid bølgemodellen særdeles kompleks, derfor velger en å se bort i fra bølgers innvirkning på hastighetsvektoren under overflaten.

Pierson- Moscowicz spekteret

Et bølgespekter består av flere enkelt- bølger som hver har individuell bølgehøyde og bølgelengde, der høyde og lengde vil være statistisk fordelt rundt middelverdier. En modell basert på dette konseptet kan implementeres ved å lage et gitt antall sinus- bølger med fase, frekvens og retning stokastisk fordelt etter Pierson- Moskowicz spek- teret:

$$S(f, \phi) = \frac{a}{f^5} e^{-(\frac{b}{f})^{-4}} \cdot D(f, \phi), \qquad \text{[Bølger: 7.]}$$

$$D(f, \phi) = \frac{1}{N(f)} (\cos(\frac{\phi}{2}))^{2p(\zeta)}, \qquad \text{[Bølger: 8.]}$$

f : Bølgefrekvens.
ϕ : Bølgeretning.
p(ζ): Stokastisk funksjon som bestemmer variasjon i retning.
a, b: Konstanter.

Dette er en nokså tungvint, lite anvendelig fremgangsmåte for simulering i en til- standsmodell.

Bretshneiders bølgespekter

En bedre metode vil være å implementere et filter med overføringsfunksjon som tilsvarer spektralfordelingen for sjø. Ved å påtrykke hvit støy på filterets inngang, vil en få sjø med korrekt utseende på utgangen. Bretschneider har funnet et uttrykk for sjøens spektralfordeling:

$$S\eta(\omega) = 1.35 \cdot h_s^2 \cdot \frac{1}{\omega} \cdot (\frac{\omega \cdot T_s}{2\pi})^{-4} \cdot e^{-0.675 \cdot (\frac{\omega \cdot T_s}{2\pi})^{-4}}, \qquad \text{[Bølger: 9.]}$$

h_s: Signifikant bølgehøyde
T_s: Signifikant bølgeperiode
Sη(ω): Spektralfordeling for måling av overflateposisjon (z- retning) over tid i et fast punkt i xy- planet.

Denne funksjonen implementers i et filter, slik at følgende sammenheng opprettholdes [Ligning Teori:24.]:

$$S\eta(\omega)=F(j\omega)F(-j\omega)$$

Bretshneider's funksjon er svært kompleks slik at det er vanskelig å finne et uttrykk for F som passer, dessuten må filterfunksjonen kunne implementeres i simuleringsprogrammet med en rask filteralgoritme. Det er fristende å prøve med en tilnærmingsfunksjon, et 2. ordens båndpassfilter (Dette er egentlig et 1.- ordens BP- filter, men en velger å kalle det et 2.- ordens filter fordi overføringsfunksjonen er av 2. orden) :

$$H_{LP}=\frac{kf}{s+a}\Rightarrow H_{BP}=H_{LP,s=\frac{s^s+\omega_0^2}{Bs}}=\frac{B\cdot kf\cdot s}{s^2+B\cdot a\cdot s+\omega_0^2}$$ [Bølger: 10.]

der en velger B=1. Spektralfunksjonen for dette filteret blir:

$$S(\omega)=\frac{kf^2\cdot\omega^2}{\omega^4+\omega^2(a^2-2\omega_0^2)+\omega_0^4}$$ [Bølger: 11.]

Koeffisientene a og kf velges slik at amplitude ved senterfrekvens og spekterts form blir mest mulig lik Bretschneider- funksjonen. En velger signifikant bølgelengde og bølgehøyde, og får da [Ligning 2., 3.]:

$$Ts=\sqrt{\frac{\lambda s}{1.56}},\omega_0=\frac{2\pi}{Ts}$$ [Bølger: 12.]

Senterfrekvens for Bretschneider spekteret (den frekvens hvor spekterets amplitude er størst) er forskjøvet i forhold til bølgefrekvensen:

$\omega_s=0.833\omega_o$ [Bølger: 13.]

Denne sammenhengen er funnet ved observasjon.

Ved å studere spekterfunksjonene har en kommet fram til at senterfrekvens for 2. ordens filteret, ω_2 kan plasseres midt mellom ω_s og ω_o. Videre viser det seg at Q- faktoren for beste kurveformtilpasning er konstant, Q=2. Da har en nok informasjon til å bestemme parametrene a og k_f:

$$\omega_2=\frac{\omega_s+\omega_0}{2}$$ [Bølger: 14.]

Fra allment kjent teori for 2. ordens funksjoner:

$$a=\frac{\omega_2}{Q}=\frac{\omega_2}{2}$$ [Bølger: 15.]

Spekterfunksjonenes verdi ved senterfrekvens skal være den samme:

$$S\eta(\omega_s)=S(\omega_2)\Rightarrow kf=a\cdot\sqrt{S\eta(\omega_s)}$$ [Bølger: 16.]

Overfører til kanonisk- form tilstandsmodell, som kan implementeres i simuleringsprogrammet:

$$\dot{x}=Ax+Bu,y=Dx$$ [Bølger: 17.]

$$A:=\begin{pmatrix}0 & 1\\ -\omega 2^2 & -a\end{pmatrix}\qquad B:=\begin{pmatrix}0 & 0\\ 0 & kf\end{pmatrix}\qquad D:=\begin{pmatrix}1 & 0\\ 0 & 0\end{pmatrix}$$

Filteret er simulert med MathCad [BOELGER4.MCD, appendix A]. En ser at den tilnærmede 2. ordens funksjonen følger Bretschneider- spekteret svært godt.

Kobling til fartøyet

Fartøyets hastighet vil endre bølgespekterets senterfrekvens (doppler- effekt):

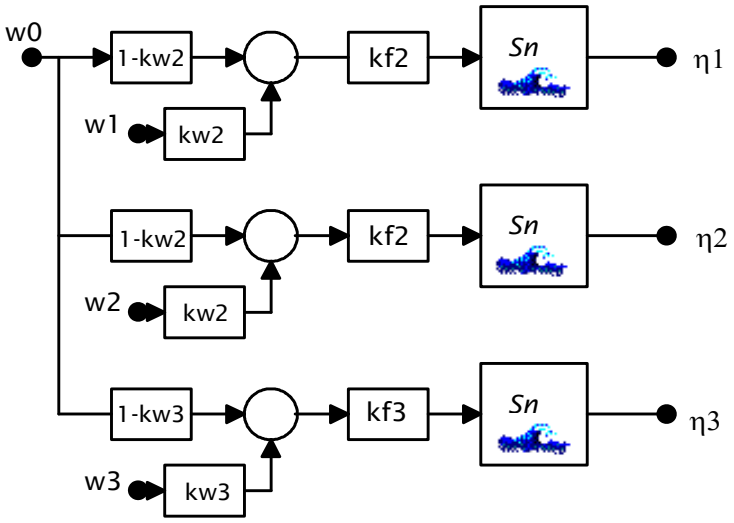
$$\omega_s' = \omega_s + \frac{Vx \cdot \cos\theta}{g} \cdot \omega_s^2 \qquad \text{[Bølger: 18.]}$$

Denne ligningen flytter senterfrekvensen i samsvar med fartøyets hastighet i bølgeretningen.

I programmet implementeres doppler- skiftet med korreksjonsfaktor for sjøens variasjon i global y- retning. En setter at bølgelengden i y - retning er ti ganger større enn i x- retning. Da opprettholdes en mer realistisk modell når fartøyet kjører på tvers av bølgeretningen (Vx=0, Vy ≠ 0).

En ønsker å påvirke fartøyet i tre punkter, ved senter for hver enkelt foil- enhet. Tre signaler med spektralfordeling lik Bretschneider's bølgespekter kan framskaffes som vist i figur 4.2.2.1.

Figur 4.2.2.1: Bretshcneider bølgesimulator



Sη: Bretschneider filter.
w0, w1, w2, w3: Hvit støy signaler.
η1: Sjøens overflateoffset ved høyre fremre foilenhet.
η2: Sjøens overflateoffset ved venstre fremre foilenhet.
η3: Sjøens overflateoffset ved aktre foilenhet.
kw2: Koeffisient for variasjon på tvers av fartøyet. 0<kw2 <1.
kw3: Koeffisient for variasjon i fartøyets lengderetning. 0<kw3 <1.

Koeffisienter kw2, kw3 velges slik at en får en variasjon mellom signalene som er mest mulig virkelighetsnært. kw2 bestemmer variasjonen i sideretning, kw3 bestemmer variasjonen i lengderetning. Koeffisientene dreies i forhold til fartøyets kurs. Se Appendix C: Programlisting, filen OEV0.PAS, objektet OEVOmgivelser, metode runproc for detaljer.

Etter veining med koeffisientene kw2, kw3 og summering må signalene multiplisereres med faktorene kf2, kf3, slik at varians for signalene inn til bølgefilter opprettholdes (En ønsker normalfordelt støy med varians 1, N(0,1)). Det er innlysende at følgende sammenheng for summering av normalfordelte, uavhengige stokastiske variable gjelder:

$$\xi_i \in N(0,\sigma) \Rightarrow \sum c_i \cdot \xi_i \in N(0, \sqrt{\Sigma c_i^2 \sigma_i^2}\,)$$ [Bølger: 19.]

Da blir uttrykket for å finne koeffisientene kf2, kf3 slik:

$$N(0,1) = kf \cdot N(0, \sqrt{(1-kw)^2 + kw^2}\,)$$ [Bølger: 20.]

$$\Rightarrow kf = \sqrt{1 - 2kw + 2kw^2}^{\,-1}$$ [Bølger: 21.]

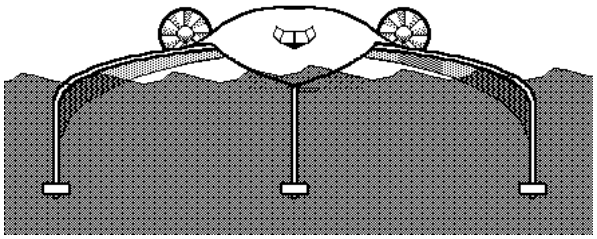
En har nå nok informasjon til å kunne implementere en bølgemodell i simuleringsprogrammet.

4.3. MODELLERING AV OEV- FARTØY

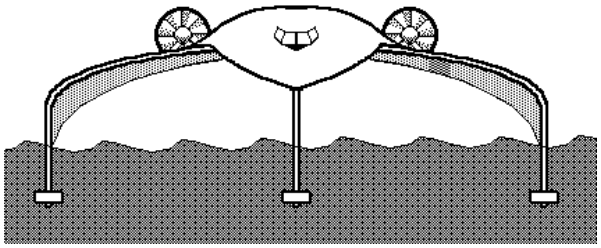
4.3.1. Design av OEV

OEV- fartøyets fysiske dimensjoner og grunnleggende egenskaper.

Figur 4.3.1.1.: OEV, nedsenket.



Figur 4.3.1.2.: OEV, luftbåren.



FYSISKE DATA FOR OEV- FARTØY:

Lengde: 20 m.
Bredde: 18 m.

Hastighetsområde, senket: 0 - 20 knop
Hastighetsområde, luftbåren: 54 - 100 knop
Design hastighet: 80 knop

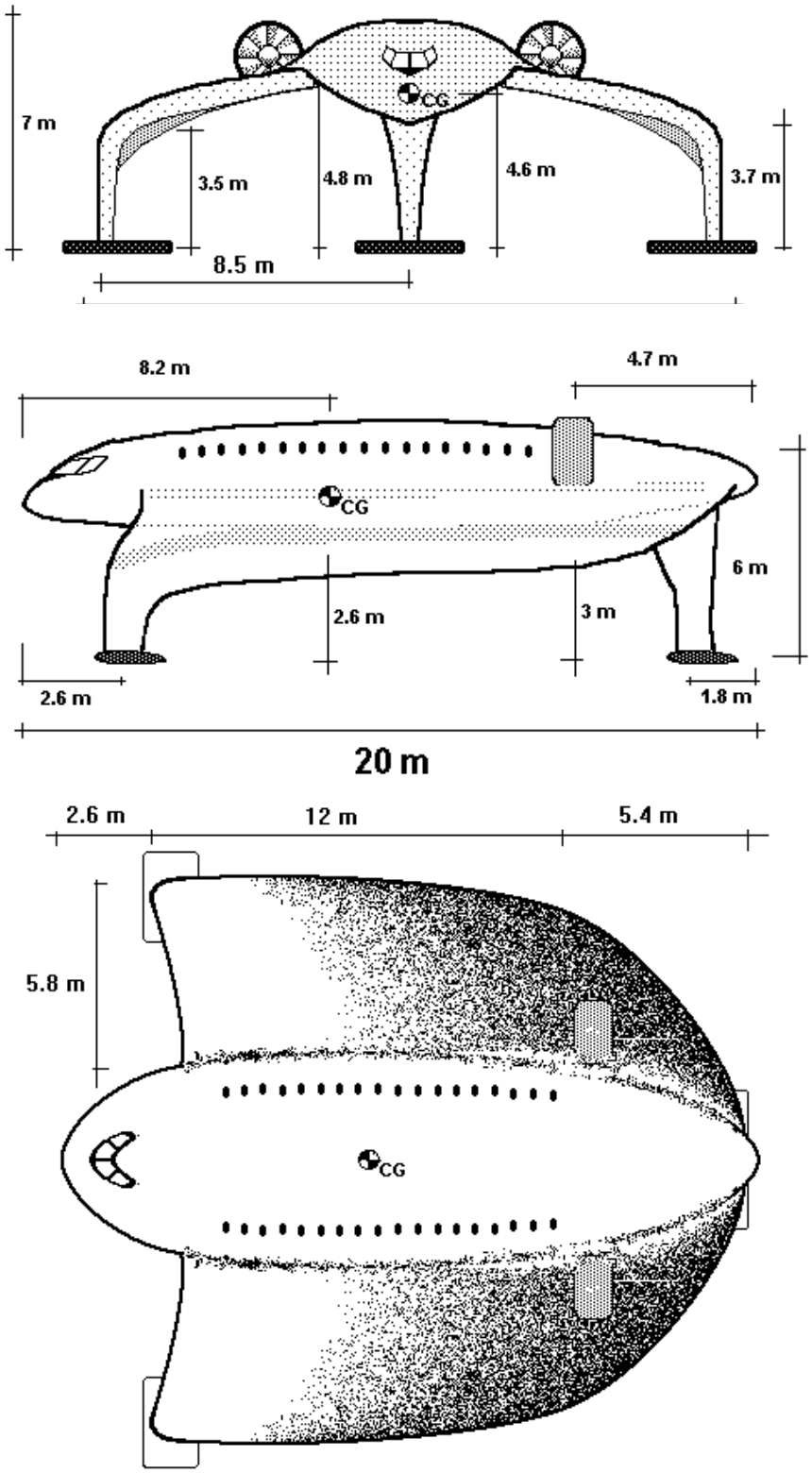
Flyhøyde: 1.7 m. (Avstand sjø- skrog, midtseksjon).

Skyvekraft motorer: 14000 kp.

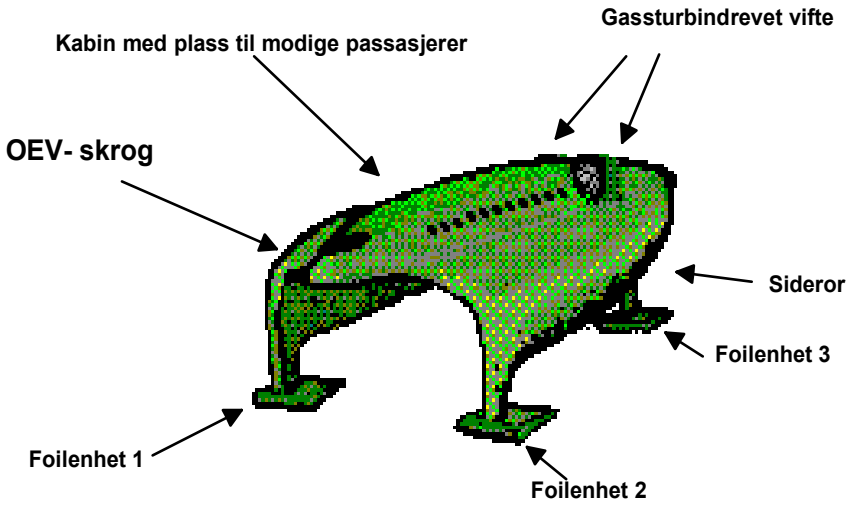
Totalvekt: 40 tonn.
Nyttelast: 6 tonn ?

Inertimoment, x- akse: 150 000 kgm²
Inertimoment, y- akse : 900 000 kgm²
Inertimoment, z- akse: 1000 000 kgm²

Figur 4.3.1.3.: Fysiske dimensjoner.



Figur 4.3.1.4.: OEV- fartøyets pådragsorganer



Foilenhet 1:

Høyderor, maksimalt utslag $\pm 10°$ fra nullstilling.
Rotasjonsretning som vinkel ψ.

Foilenhet 2:

Høyderor, maksimalt utslag $\pm 10°$ fra nullstilling.
Rotasjonsretning som vinkel ψ.

Foilenhet 3:

Høyderor, maksimalt utslag $\pm 10°$ fra nullstilling.
Rotasjonsretning som vinkel ψ.
Sideror, maksimalt utslag $\pm 10°$ fra nullstilling.
Rotasjonsretning som vinkel θ.

Skyvekraft:

Maksimalt pådrag 14000 kp.

Design av OEV- fartøyet baserer seg på Lippisch' resultater av forsøk med WIG-fartøyer [ref. 2.]. Den høye designhastigheten er nødvendig for å få effekt av aerodynamisk løftekraft.

Fartøyet ble tegnet av prosjektdeltakerne før inngående studier innen hydrodynamikk ble foretatt. Det viste seg senere, ved simulering av fartøyet, at konstruksjonen har enkelte uheldige aspekter.

4.3.2. Dynamisk modellering

Beskrivelse av kreftene som virker på fartøyet. Metode for hvordan en kommer fram til et sett med differensialligninger som beskriver fartøyets dynamikk. Detaljert beskrivelse i appendix A: SKROG.MCD.

Modellens anvendelsesområde

Vår modell blir en forenklet beskrivelse av OEV- fartøyet, der en tar hensyn til de faktorer som en antar har størst innvirkning på oppførselen, under gitte arbeidsforhold. En prøver å lage en modell som er best mulig rundt fartøyets normale arbeidsforhold: Vx= 40 knop, Zg= 2.9 m., ϕ=0, ψ=0. Oppførsel ved veldig lave hastigheter (< 20 knop) og når fartøyets skrog er i kontakt med vannet (nedsenket driftsmodus) er ikke så viktig. Hvor god modellen er, avhenger av fartøyets aktuelle tilstand i forhold til de forutsetninger en har gjort.

En ønsker å operere med to modeller, en linearisert approksimasjon som gjelder for små endringer rundt et gitt arbeidspunkt, og en mer fullstendig modell beskrevet av ulineære ligningssett. Den ulineære modellen kan implementeres i et simuleringsprogram. Den lineariserte modellen er nødvendig hvis en skal anvende teori for lineære systemer for beregning av regulator.

Kraftkomponenter

Kreftene som virker på skrog og foiler defineres i forhold til fartøyets tyngdepunkt, slik at fartøyets dynamiske oppførsel kan beregnes ved å summere krefter og momenter langs koordinataksene x, y, z :

$$\begin{aligned} \Sigma F_x &= m \cdot a_x^I \\ \Sigma F_y &= m \cdot a_y^I \\ \Sigma F_z &= m \cdot a_z^I \end{aligned} \qquad \text{[M: 1.]}$$

$$\begin{aligned} \Sigma \Gamma_\phi &= I_x \cdot \alpha_\phi \\ \Sigma \Gamma_\psi &= I_y \cdot \alpha_\psi \\ \Sigma \Gamma_\theta &= I_z \cdot \alpha_\theta \end{aligned} \qquad \text{[M: 2.]}$$

Ligninger for krefter fra fartøyets foiler og skrog implementers i MathCad [Appendix A: SKROG.MCD]. En linearisert modell (ref. punkt 3.1.3.) kan konstrueres ved å partiellderivere kreftene med hensyn på alle variable som forekommer i ligningen:

Ligning for kraft:

$$F = f(x_1, x2..x_n) \qquad \text{[M: 3.]}$$

- gir lineariserte elementer:

$$\begin{aligned} F_1 &= f(x_1) = \tfrac{\partial}{\partial x_1} F, \\ F_2 &= f(x_2) = \tfrac{\partial}{\partial x_2} F, \\ &\ldots \\ F_n &= f(x_n) = \tfrac{\partial}{\partial x_n} F \end{aligned} \qquad \text{[M: 4.]}$$

Dette kan gjøres direkte i MathCad.

Euler's ligninger for bevegelse

Hvis en gjenstand med masse m har hastighet referert til gjenstandens referanseakser x, y, z, og samtidig roterer rundt aksene, vil en gjenstanden være akselleret langs koordinataksene:

$$a_x^I = -\omega_\psi \cdot v_z + \omega_\theta \cdot v_y \qquad \text{[M: 5.]}$$

$$a_y^I = \omega_\theta \cdot v_x - \omega_\phi \cdot v_x$$

$$a_z^I = \omega_\phi \cdot v_y - \omega_\psi \cdot v_x$$

Da får en følgende uttrykk for massesenterets akselleresjon:

$$\frac{dV_x}{dt} = a_x = \frac{\sum F_x}{m} - \omega_\psi \cdot v_z + \omega_\theta \cdot v_y \qquad \text{[M: 6.]}$$

$$\frac{dV_y}{dt} = a_y = \frac{\sum F_y}{m} + \omega_\theta \cdot v_x - \omega_\phi \cdot v_x$$

$$\frac{dV_y}{dt} = a_y = \frac{\sum F_y}{m} + \omega_\phi \cdot v_y - \omega_\psi \cdot v_x$$

Ligningssett 5., 6., sammen med ligningssett 2., beskriver fartøyets dynamikk.

Rotasjon rundt aksene vil føre til svinghjulseffekter som motvirker dreining om rotasjonsplanets akse. Euler's ligninger gir en fullstendig beskrivelse av fartøyets rotasjonsdynamikk, der påvirkning fra rotasjon om flere akser samtidig er med. Disse effektene ser en bort fra, fordi en kan anta at fartøyets rotasjonshastighet er forholdsvis lav under normale driftsforhold, og innbyrdes påvirkning mellom rotasjonshastighetene bli da liten.

Linearisering av sin, cos til vinkler

En setter som forutsetning at fartøyets rotasjon rundt x- akse og y- akse er liten. Da blir uttrykkene for de forskjellige kreftene av overkommelig kompleksitet, og sin(x), cos(x) kan approksimeres slik (x i radianer):

$$\frac{d}{dx}\sin(x)_{x=0} = 1 \Rightarrow \sin(x) \approx x\,, \quad \frac{d}{dx}\cos(x)_{x=0} = 0 \Rightarrow \cos(x) \approx 1 \qquad \text{[M: 7.]}$$

Simuleringsmodellen som tar hensyn til ulineære elementer er signifikant bare hvis ϕ, $\psi < \pi/2$, og ved vinkler opp mot $\pi/2$ vil det oppstå betydelige feil. Det har liten praktisk betydning, for da har fartøyet likevel havarert.

V_{ZG}: Hastighet i global z- retning

Fartøyets hastighet i global z- regning avhenger av rotasjonen rundt x- og y- akse, og hastighetskomponentene langs fartøyets referanseakser. En antar at fartøyets dreining i forhold til globalt xy- plan er liten, og får da:

$$V_{ZG} = V_{ZG_x} + V_{ZG_y} + V_{ZG_z} = \cos\phi \cdot \cos\psi \cdot V_Z - \sin\psi \cdot V_X - \sin\phi \cdot V_Y \qquad \text{[M: 8.]}$$

For implementasjon i simuleringsprogram settes sin(x)=x:

$$V_{ZG} = \cos\phi \cdot \cos\psi \cdot V_Z - \psi \cdot V_X - \phi \cdot V_Y \qquad \text{[M: 9.]}$$

For å få en linearisert modell må en gjøre følgende forenklinger:

$V_y=0$ [M: 10.]
V_x=konstant=V_{x0}

cos(x)=1

$$\Rightarrow V_{ZG,lin} \approx V_Z - \psi \cdot V_{X0} \qquad \text{[M: 11.]}$$

Dekomponering av G

Fartøyets tyngde virker langs global z- akse, i negativ retning. Komponenter langs fartøyets akser blir avhengig av fartøyets helning rundt x- og y- akse. Hvis en antar at helningen er relativt liten (<< 90°) kan komponentene finnes slik:

$$\begin{aligned} G_x &= m_{tot} \cdot g \cdot \sin\psi \cdot \cos\phi \\ G_y &= -m_{tot} \cdot g \cdot \cos\psi \cdot \sin\phi \\ G_z &= -m_{tot} \cdot g \cdot \cos\psi \cdot \cos\phi \end{aligned} \qquad \text{[M: 12.]}$$

En får relativt liten feil ved å sette sin(x)=x, denne tilnærmingen kan benyttes i simuleringsprogrammet. En linearisert modell må også erstatte leddene cos(x) slik at cos(x)=1. En får da:

$$\begin{aligned} G_{x,lin} &= m \cdot g \cdot \psi \\ G_{y,lin} &= -m \cdot g \cdot \phi \\ G_{z,lin} &= -m \cdot g \end{aligned} \qquad \text{[M: 13.]}$$

Gz, lin er konstant og utkanselleres av løftekreftene i arbeidspunktet.

Aerodynamiske krefter

Teori for modellering av skrog bygger på Lippisch' beskrivelse av WIG- fartøyer [ref. 2.]. Aerodynamiske krefter vil ha formen:

$$L_A = C_L \cdot \tfrac{1}{2} \cdot \rho \cdot v^2 \qquad \text{[M: 14.]}$$

$$D_A = C_D \cdot \tfrac{1}{2} \cdot \rho \cdot v^2 \qquad \text{[M: 15.]}$$

- der L_A er løftekraft normalt på hastighetsvektoren, og D_A er drag langs hastighetsvektoren, ρ angir mediets tetthet, v er hastighetsvektorens absoluttverdi.

Hvis angrepsvinkelen er liten, kan en si at løft virker langs fartøyets positive z- akse, og drag langs fartøyets negative x- akse.

På grunn av overflateeffekten og ram- løft vil løftet øke når fartøyet kommer nærmere overflaten. C_L blir en funksjon av fartøyets posisjon i forhold til overflaten. Lippisch har gjort modellforsøk med en skrogfasong utformet som en del av en ellipse. En anvender resultatene fra disse forsøkene for å finne passende koeffisienter. Vind i x- retning trekkes fra fartøyets hastighet i x- retning.

Vind i sideretning vil føre til krefter i y- retning og løft i z- retning. En velger å forenkle representasjonen av disse kreftene ved å anta sannsynlige verdier for C_L og C_D. Når fartøyets hastighet er høy i forhold til vinden, vil disse kreftene være små i forhold til de andre løfte- og dragkreftene, slik at feilen blir liten. Følgende koeffisienter velges:

$$\begin{aligned} C_{Dy} &= 0.3 + 0.05 C_{Ly}^2 \\ C_{Ly} &= 0.6 + 0.1 \alpha_y \end{aligned} \qquad \text{[M: 16.]}$$

I modellen blir angrepsvinkelen $\alpha_y = -\phi$. Videre kan en anta at fartøyets hastighet i y- retning er neglisjerbar. En har sett bort fra eventuelle induserte momenter grunnet vind i y- retning.

Hydrodynamiske krefter

I sideretning vil det virke hydrodynamiske krefter som funksjon av hastighetsvektor relativt skrogflater under overflaten. Disse modelleres som foiler, der arealet er avhengig av fartøyets posisjon i forhold til overflaten. Kreftene vil være funksjoner av koeffisientene C_L, C_D, og er gitt av ligningene 14., 15. Løftekraften vil her være kraft i y- retning, altså i sideretning. En tar utgangspunkt i teori og praktiske forsøk beskrevet i Handbook of Ocean and Underwater Engineering [ref. 1.] og Hydrodynamics of high- speed small craft [ref. 5.] for å finne sannsynlige verdier for koeffisientene (vinkelkoeffisienter refererer til radianer):

$$K_{hydro},C_{D0} = 0.008 \qquad \text{[M: 17.]}$$

$$K_{hydro}.C_{D\alpha} = 0.3$$

$$K_{hydro},C_{L\alpha} = 0.859$$

Siderorets pådragsmekanisme modelleres på samme måte som for foiler.

Det implementeres et enkelt uttrykk for statisk oppdrift, slik at fartøyet ikke synker når farten er lav. Statisk oppdrift ved hver enkelt foilenhet vil være en funksjon av fartøyets posisjon i forhold til overflaten.

Hydrodynamisk løft fra skrog vil være avhengig av om skroget er i kontakt med vannflaten. En enkel modell for skrogets løft og drag tas med i den ulineære modellen, slik at en får simulert overgang flytende- luftbåren.

Foiler

En velger en enkel representasjon for løftekraft og drag fra foiler, der en ser bort fra kavitasjonseffekter (løftet vil avta ved økende hastighet) og påvirkning fra bølger:

$$L_F = C_L \cdot \tfrac{1}{2} \cdot \rho_A \cdot A \cdot v^2, der\ v = Vx, C_L = C_{L0} + k_{C_L\alpha} \cdot \alpha \qquad \text{[M: 18.]}$$

$$D_F = C_D \cdot \tfrac{1}{2} \cdot \rho_A \cdot A \cdot v^2, der\ v = Vx, C_D = CD0 + k_{C_D\alpha} \cdot C_L^2 \qquad \text{[M: 19.]}$$

En tar utgangspunkt i teori og praktiske forsøk beskrevet i Handbook of Ocean and Underwater Engineering [ref. 1.] og Hydrodynamics of high- speed small craft [ref. 5.] for å finne sannsynlige verdier for koeffisientene (vinkelkoeffisienter refererer til radianer):

$$K_{foil},C_{D0} = 0.006 \qquad \text{[M: 20.]}$$

$$K_{foil},C_{D\alpha} = 0.3$$

$$K_{foil},C_{L\alpha} = 0.859$$

$$K_{foil},C_{L0} = 0.02$$

$$A_{ref,foil} = 2$$

Foilenes angrepsvinkel justeres ved å gi et pådrag til en mekanisme som dreier hele foilen. Dette kan for eksempel være hydrauliske aktuatorer. Da kan en tenke seg følgende sammenheng mellom pådrag og foilvinkel:

$$\alpha = \alpha_u \cdot \frac{a}{s+a} \qquad \text{[M: 21.]}$$

- der 1/a= tidskonstant for foilmekanismen. Ligning for endringshastighet, foilvinkel blir da:

$$\frac{d}{dt}\alpha = -a \cdot \alpha + a \cdot \alpha_u \qquad \text{[M: 22.]}$$

En velger a=1/t,foil= 2π5 for alle tre foiler og sideroret.

4.3.3. Instrumentering

Beskriver hvordan målesignaler fremskaffes og hvilke målinger en har tilgang til.

Instrumenteringens formål

Målevektor for OEV-modellen er definert i kapittel 4.3.4.: Tilstandsmodellering, y-vektor. Det er disse signalene instrumenteringsutstyret skal levere. Tabellen under viser de tilstander en regner med å kunne måle direkte:

Måling: Målesignaler				Forslag til instrument
i	**Tilstand**	**Enhet**	**Beskrivelse**	
1	v_x	m/s	Måling av hastighet i kjøreretning (x)	Pitotrør, ultralyd, Dopplereffekt...
2	a_x	m^2/s	Måling av akselerasjon i kjøreretning (x)	Aksellerometer
3	a_y	m^2/s	Måling av akselerasjon i sideretning (y)	Aksellerometer
4	a_z	m^2/s	Måling av akselerasjon i høyderetning (z)	Aksellerometer
5	ϕ	rad	Vinkel om x-akse	Gyro
6	ψ	rad	Vinkel om y-akse	Gyro
7	θ	rad	Vinkel om z-akse	Gyro, gyrokompass
8	zm1	m	Høydeforskjell CG->overflate, ved foil1	Ultralyd
9	zm2	m	Høydeforskjell CG->overflate, ved foil2	Ultralyd
10	zm3	m	Høydeforskjell CG->overflate, ved foil3	Ultralyd

En har gått gjennom mulige løsninger for fysisk instrumentering. Løsningene er vurdert og de antatt beste alternativene ble valgt.

Det må detekteres posisjonsforandring og endring i akselleresjon på tre akser, hastighet i kjøreretningen samt høyde over havet.

OEV-fartøyet vil forårsake store endringer i de forskjellige målemedia selv under normale driftsforhold. Endringene kan være turbulente luftstrømmer, vannstrømmer som oppstår på grunn av hastigheten, kavitasjon, sjøsprøyt. Problemet blir løst ved å bruke flere instrumenter, med ulike normal- betingelser, som i kombinasjon gir det signalet en ønsker. Dette kan styres ved hjelp av en estimator.

Måling av hastighet

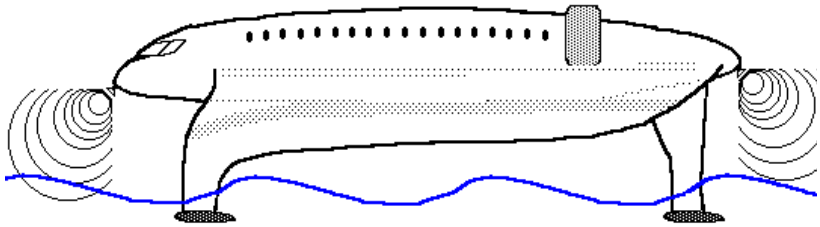
OEV-fartøyet's normalbetingelser for hastighet vil ligge mellom 0 -120 knop, dvs 61 m/s eller ca 220 km/t. Marsjfart er tenkt til ca 80 knop, dvs 41m/s eller ca 150 km/t. Den høye hastigheten forårsaker forstyrrelser i målemediet som det må taes hensyn til.

Pitotrør er en måleteknikk som baserer seg på trykkdifferansen mellom to målepunkt. I luft så er denne metoden ubrukbar pga vind. Pitotrør i sjø vil også bli problematisk pga turbulens og kavitasjonseffekter etter hvert som hastigheten øker. Ved lave hastigheter (opptil 30 knop) kan et pitotrør for sjø festet rett over foil 1 eller 2 være en brukbar løsning, se figur 4.3.3.5. Transducer og Pitotrør.

Dopplereffekten er en godkjent metode til å måle hastighet med. En tenker seg to målere, en i baugen og en i hekken , slik at man kan filtrere vekk den feilen som bølgehastigheten ville gitt med bare en måler. Et problem er å filtrere vekk den støyen som sjøsprøyt kan gi.

Løsningen må altså bli pitotrør i sjø for lave hastigheter, og dopplerskiftbasert måling ved hjelp av ultralyd for høyere hastigheter. Dette gir oss signalet V_x.

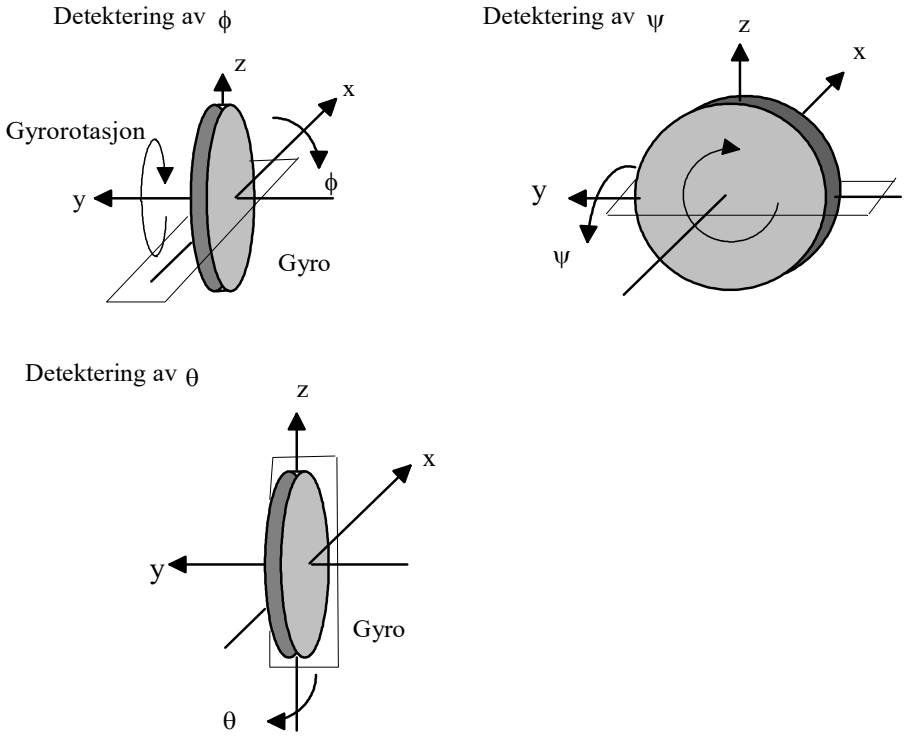
Figur 4.3.3.1. Dopplermåling av hastighet



Måling av bevegelser

For å kunne detektere akselerasjon og rotasjon bruker vi henholdsvis akselerometere og gyroer. Velger å plassere 3 enkeltakse- gyroer og 3 akselerometere i CG, dette gir tilstrekkelig informasjon om fartøyets bevegelser. Gyroens plassering i forhold til akse vises på figur 4.3.3.2. Gyroer, opphengsramme for gyro tegnet med stiplet linje.

Figur 4.3.3.2. Gyroer



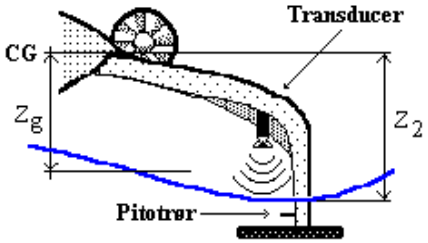
Høydemåling

Til høydemåling vil vi bruke ultralyd. Vi har valgt å bruke transducere plassert som på figur 4.3.3.3. og figur 4.3.3.4. Dette resulterer i målesignaler for høydene z1, z2, z3. Signalene angir avstand fra tyngepunktets plassering i z- retning ned til overflaten:

$$z_1 = Z_g - 8.5\phi - 5.5\psi - v_3 \qquad \text{[I: 1.]}$$
$$z_2 = Z_g + 8.5\phi - 5.5\psi + v_4$$
$$z_3 = Z_g + 10\psi - v_5$$

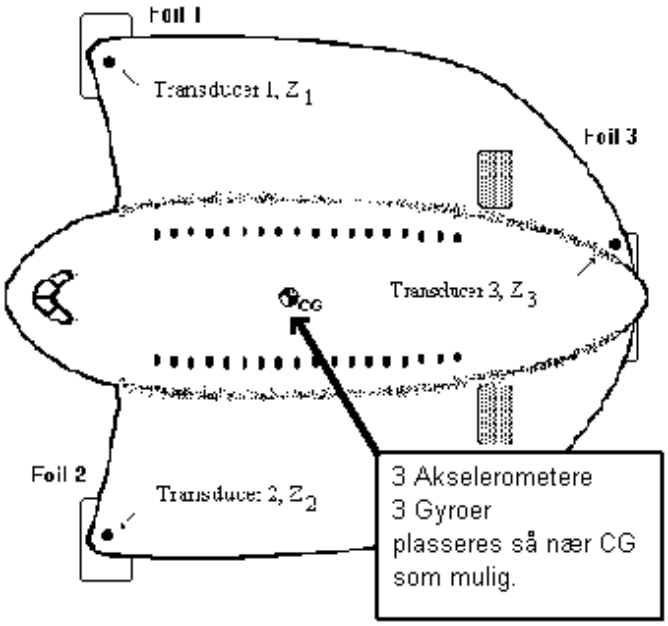
- der v_3, v_4, v_5 angir sjøens overflate i punktene der målingen foretas.

Figur 4.3.3.3. Transducer og Pitotrør.



Vi vil imidlertidig få en del støy på grunn av sprayeffekt fra sjøen, dette tas hensyn til i implementeringen av estimator. Videre må vi bruke en estimator til å regne ut Z_g da dette er høyden som ønskes som referanse i regulatoren.

Figur 4.3.3.4. Plassering av transducere, aksellerometre og gyroer.



Tilstandsestimator for instrumentering

Denne tilstandsestimatoren skal fremskaffe målesignalene v_x, v_y, v_z, ϕ, ψ, θ, ω_ϕ, ω_ψ, ω_θ på grunnlag av målingene V_x , a_x , a_y , a_z , ϕ , ψ , θ.

Med utgangspunkt i ligninger fra kapittel 4.3.2. får en følgende utledninger for kreftene i de respektive retningene:

$$\Sigma F_x = ma_x = m(\frac{dV_x}{dt} + \omega_\psi V_z - \omega_\theta V_y)$$

$$\Sigma F_y = ma_y = m(\frac{dV_y}{dt} + \omega_\theta V_x - \omega_\phi V_z)$$

$$\Sigma F_z = ma_z = m(\frac{dV_z}{dt} + \omega_\phi V_y - \omega_\psi V_x)$$

Dette gir :

$$a_x = \frac{dV_x}{dt} + \omega_\psi V_z - \omega_\theta V_y$$

$$a_y = \frac{dV_y}{dt} + \omega_\theta V_x - \omega_\phi V_z$$

$$a_z = \frac{dV_z}{dt} + \omega_\phi V_y - \omega_\psi V_x$$

$$\omega_\phi = \frac{d\phi}{dt}$$

$$\omega_\psi = \frac{d\psi}{dt}$$

$$\omega_\theta = \frac{d\theta}{dt}$$

For Z_g har en følgende, ved å ta utgangspunkt i fartøyets fysiske dimensjoner og plassering av transducere for høydemåling:

$$\dot{z}_g = V_z - \psi V_x - \phi V_y$$

$$Z_g = 8.5\phi + 5.5\psi - Z_1 + V_3$$

$$Z_g = -8.5\phi + 5.5\psi - Z_2 + V_4$$

$$Z_g = -10\psi - Z_3 + V_5$$

Disse ligningene danner utgangspunktet for en modell av fartøyet. Denne modellen behøver ikke ha samme struktur som fartøyets egentlige modell, her er det mer hensiktsmessig å velge en representasjon som har færrest mulig komponenter relatert til fartøyets fysiske data, modellen bør også være lineær.

Målesignalene splittes i en ny y- vektor og en u- vektor, slik at akselerasjon kommer inn som et pådrag til estimatormodellen:

Estimator: y-vektor				Forslag til instrument
	Tilstand	**Enhet**	**Beskrivelse**	
1	v_x	m/s	Måling av hastighet i kjøreretning (x)	Pitotrør, ultralyd, Dopplereffekt...
2	ϕ	rad	Vinkel om x-akse	Gyro
3	ψ	rad	Vinkel om y-akse	Gyro
4	θ	rad	Vinkel om z-akse	Gyro, gyrokompass
5	zm1	m	Høydeforskjell CG->overflate, ved foil1	Ultralyd
6	zm2	m	Høydeforskjell CG->overflate, ved foil2	Ultralyd
7	zm3	m	Høydeforskjell CG->overflate, ved foil3	Ultralyd

Estimator: u-vektor				Forslag til instrument
	Tilstand	**Enhet**	**Beskrivelse**	
1	a_x	m^2/s	Måling av akselerasjon i kjøreretning (x)	Akselerometer
2	a_y	m^2/s	Måling av akselerasjon i sidereretning (y)	Akselerometer
3	a_z	m^2/s	Måling av akselerasjon i høyderetning (z)	Akselerometer

Påvirkning på vinkelakselleraSjon modelleres som støy fra v- vektoren. Nå kan følgende matriser defineres:

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & -x_3 & -x_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -x_4 & x_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_4 & 0 & -x_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -x_3 & x_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$V = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.004 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.004 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.004 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -8.5 & -5.5 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 8.5 & -5.5 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 10 & 0 & 0 & 0 & 0 \end{bmatrix}$$

A- matrisen er ulineær. De ulineære elementene kan oppdateres kontinuerlig med utgangspunkt i den estimerte tilstandsvektoren. Hvis de ulineære elementene varierer sakte i forhold til estimatorens X- matrise, kan dette gå bra. Ellers kan en risikere at estimatoren ikke konvergerer (K- matrise ubestemt) eller blir ustabil (K- matrisen konvergerer for sakte).

I A- matrisen kan en utelate de ulineære elementene og får:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Verdier for støymatrisene V og W velges ut i fra sannsynlig målestøy for de respektive signaler og forventet variasjon for påvirkning fra omgivelser:

$$W = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 \cdot 10^{-4} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 \cdot 10^{-4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 \cdot 10^{-4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.8 \end{bmatrix}$$

K beregnes med programvaren MatLab [Appendix B: OEVEST.M]:

$$K = \begin{bmatrix} 0 & 0 & 0.192 & 0 & 0.204 & 0.204 & 0.205 \\ 0.003 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.011 & 0 & 0.012 & 0.012 & 0.012 \\ 0 & 12.283 & 0 & 0 & -0.039 & 0.039 & 0 \\ 0 & 0 & 12.231 & 0 & -0.025 & -0.025 & 0.046 \\ 0 & 0 & 0 & 12.779 & 0 & 0 & 0 \\ 0 & 79.523 & 0 & 0 & -0.253 & 0.253 & 0 \\ 0 & 0 & 79.298 & 0 & -0.164 & -0.164 & 0.297 \\ 0 & 0 & 0 & 81.65 & 0 & 0 & 0 \end{bmatrix}$$

Denne tilstandsestimatoren er utprøvd i forbindelse med prosjektoppgave i faget Instrumenteringsteknikk 2: OEV- Instrumenter [ref. 33].

SEATEX Motion Reference Unit

På et senere stadium kom en over et instrument for bevegelsesmåling beregnet på båter, SEATEX MRU-6 (SEATEX A/S, N-7005 Trondheim), som leverer alle nødvendige målesignaler direkte. Dette er et såkalt smart instrument, som har innebygd CPU. Regulator og tilhørende estimator for regulatoren baserer seg på at målesignalene v_y, v_z, ϕ, ψ, θ, ω_ϕ, ω_ψ, ω_θ fremskaffes ved bruk av dette instrumentet.

4.3.4. Tilstandsmodell

Overgangen fra dynamisk modell til tilstandsmodell. Detaljert beskrivelse i Appendix A: OEVMOD.MCD.

En definerer en tilstandsmodell for fartøyet slik at fartøyets oppførsel over tid beskrives av ligningen (ref. kapittel 3.2.6.):

$$\dot{x} = Ax + Bu + Cv, y = Dx + Fv + Ew$$

- der x er tilstandsvektor for fartøyet, u er pådragsvektor, y er målevektor, v er støyvektor fra omgivelser, w er hvit støy. Matrisen F må være med for å få måling av sjøens overflate i forhold til fartøyet.

En definerer tilstander og vektorer for modellen:

X- vektor: i	Tilstand	Enhet	Beskrivelse	Beregningsgrunnlag: dx/dt=Ax+Bu+Cv
1	Zg	m	Posisjon CG i z- retning ref. globalt xy- plan.	dz/dt=Vz,g=f(Vx,Vy,Vz)
2	Vx	m/s	Hastighet i x- retning ref. fartøy.	Fx=m*a,x
3	Vy	m/s	Hastighet i y- retning ref. fartøy.	Fy=m*a,y
4	Vz	m/s	Hastighet i z- retning ref. fartøy.	Fz=m*a,z
5	ϕ	rad	Dreining rundt x- akse (sidehelning).	δϕ/δt=ωϕ
6	ψ	rad	Dreining rundt y- akse (foroverhelning).	δψ/δt=ωψ
7	θ	rad	Dreining rundt z- akse (fartøyets kurs).	δθ/δt=ωθ
8	ωϕ	rad/s	Dreiehastighet rundt x- akse.	Γ,ϕ=α,ϕ*I,ϕ
9	ωψ	rad/s	Dreiehastighet rundt y- akse.	Γ,ψ=α,ψ*I,ψ
10	ωθ	rad/s	Dreiehastighet rundt z- akse.	Γ,θ=α,θ*I,θ
11	αz	rad	Sideror vinkel ref. fartøy.	d/dt(vinkel)=(1/t_foil)*(-1+u2)
12	α1	rad	Foil 1 angrepsvinkel ref. fartøy.	d/dt(vinkel)=(1/t_foil)*(-1+u3)
13	α2	rad	Foil 2 angrepsvinkel ref. fartøy.	d/dt(vinkel)=(1/t_foil)*(-1+u4)
14	α3	rad	Foil 3 angrepsvinkel ref. fartøy.	d/dt(vinkel)=(1/t_foil)*(-1+u5)

Pådrag: u- vektor i	Tilstand	Enhet	Beskrivelse	Beregningsgrunnlag: dx/dt= Ax + Bu + Cv
1	Tx	N	Skyvekraft fra motorer.	Skyvekraft direkte i x- retning.
2	α_{uz}	rad/s	Ønsket vinkel, sideror	d/dt(L)=(1/t_foil)*(-L+u)
3	α_{u1}	rad/s	Ønsket vinkel, foil 1	d/dt(L)=(1/t_foil)*(-L+u)
4	α_{u2}	rad/s	Ønsket vinkel, foil 2	d/dt(L)=(1/t_foil)*(-L+u)
5	α_{u3}	rad/s	Ønsket vinkel, foil 3	d/dt(L)=(1/t_foil)*(-L+u)

Omgivelser: v- vektor i	Tilstand	Enhet	Beskrivelse	Beregningsgrunnlag: dx/dt= Ax + Bu + Cv
1	vind,x	m/s	Vindhastighet i global x- retning.	Wiener- prosess
2	vind,y	m/s	Vindhastighet i global y- retning.	Wiener- prosess
3	Zg1	m	Sjøens overflate ved foil 1	Spektralfordeling
4	Zg2	m	Sjøens overflate ved foil 2	Spektralfordeling
5	Zg3	m	Sjøens overflate ved foil 3	Spektralfordeling

Målevektoren baserer seg på at instrumenteringsutstyret leverer måling av alle tilstandene v_x, v_y, v_z, ϕ, ψ, θ, ω_ϕ, ω_ψ, ω_θ, i tillegg til måling av overflatens posisjon:

Måling: y- vektor:				Beregningsgrunnlag:
i	**Tilstand**	**Enhet**	**Beskrivelse**	**y= Dx + Fv + Ew**
1	Vx	m/s	Hastighet i x- retning ref. fartøy.	y1=x2+E1,1*w
2	Vy	m/s	Hastighet i y- retning ref. fartøy.	y2=x3+E2,2*w
3	Vz	m/s	Hastighet i z- retning ref. fartøy.	y3=x4+E3,3*w
4	ϕ	rad	Dreining rundt x- akse (sidehelning).	y4=x8+E4,4*w
5	ψ	rad	Dreining rundt y- akse (foroverhelning).	y5=x9+E5,5*w
6	θ	rad	Dreining rundt z- akse (fartøyets kurs).	y6=x10+E6,6*w
7	$\omega\phi$	rad/s	Dreiehastighet rundt x- akse.	y7=x8+E7,7*w
8	$\omega\psi$	rad/s	Dreiehastighet rundt y- akse.	y8=x9+E8,8*w
9	$\omega\theta$	rad/s	Dreiehastighet rundt z- akse.	y9=x10+E9,9*w
10	zm1	m	Høydeforskjell CG->overflate, ved foil1	y8=f(x, v)+E10,10*w
11	zm2	m	Høydeforskjell CG->overflate, ved foil2	y9=f(x, v)+E11,11*w
12	zm3	m	Høydeforskjell CG->overflate, ved foil3	y10=f(x, v)+E12,12*w

Målestøyen, w, modelleres som hvit støy med vekt 1. Målestøy- vektoren må ha samme dimensjon, 12, som y- vektor. Diagonalelementene i E- matrisen bestemmer målestøyens styrke for de enkelte målevariable. Her velges verdier som en tror er sannsynlige, på grunnlag av informasjon om instrumenteringens oppbygning. Matrisene D, E og F kan nå settes opp.

Med utgangspunkt i resultater fra studier av foiler, skrog og fartøy kan en nå definere elementer i matrisene A, B, C. Dette er gjort ved hjelp av MathCad [Appendix A: OEVMOD.MCD]. Elementene i matrisene A, B, C vil være avhengig av fartøyets tilstandsvektor. En kan linearisere rundt et gitt arbeidspunkt, og få en lineær approksimasjon av det ulineære systemet. Systemet beskrives da av uttrykket (ref. kapittel 3.1.3.):

$$\dot{x} = \frac{\partial}{\partial x} f(x_0) \cdot (x - x_0)$$

Denne lineariserte modellen vil da gjelde for små variasjoner rundt arbeidspunktet. Ved å velge en såkalt designhastighet og flyhøyde for fartøyet, kan en sette inn tallverdier i A, B og C- matrisene, slik at det blir mulig å analysere systemets egendynamikk i arbeidspunktet. Normalbetingelsene kan benyttes for design av regulator. I filen OEVMOD.MCD plasseres elementer fra utledning av krefter i filen SKROG.MCD [Appendix A: SKROG.MCD]. Dette resulterer i en komplett linearisert tilstandsmodell for OEV- fartøyet.

Matrisene A og C kan ikke benyttes i den ulineære modellen, her er det nødvendig å implementere de ulineære ligningene i simuleringsprogrammet direkte. Implementasjon av disse baserer seg på beskrivelsene i MathCad- filene SKROG.MCD og OEVMOD.MCD. Tilstander for foiler og sideror begrenses slik at vinkel ut fra nullposisjon ikke overskrider maksimalverdiene.

Detaljert informasjon om implementasjon av ulineær modell får en ved å studere programfilen OEV0.PAS, metodene OEVProsess.calcgOmatrix, OEVProsess.calcgFmatrix, OEVProsess.runproc [Appendix C: OEV0.PAS].

4.4. SIMULERINGSPROGRAM

Beskrivelse av programmets struktur, de enkelte moduler og bruk av simuleringsprogrammet.

4.4.1. Generell beskrivelse

Det er utviklet et program som simulerer dynamiske systemers utvikling over tid av typen (punkt 3.2.6.):

Analog representasjon:

$$\dot{x} = Ax + Bu + Cv, y = Dx + Ew$$

I programmet diskretiseres denne modellen, slik at utviklingen fremover i tid kan beregnes trinnvis:

Diskret representasjon:

$$x(n+1) = \Phi x(n) + \Delta u + \Omega v, y(n) = Dx(n) + Ew(n)$$

I tillegg kan programmet simulere ulineære systemer av typen (punkt 3.1.6.):

Analog representajon, ulineært system:

$$\dot{x} = f(x, u, v) = A(x) \cdot x + B(x) \cdot u + C(x) \cdot v$$

Metode for simulering:

$$x(n+1) \approx \Phi(n) \cdot x(n) + \Delta(n) \cdot u(n) + \Omega(n) \cdot v(n),$$

- der Φ(n), Δ(n), Ω(n) er diskretisert på samme vis som for lineært system, med oppdatering for hvert enkelt sample.

For å få til dette implementeres de ulineære matrisene i programmets kode. Grad av nøyaktighet vil avhenge av hvor raskt de ulineære elementene endrer seg i forhold til samplefrekvensen. Som grunnregel kan en si at samplefrekvens skal velges minst like høy som hvis systemet var lineært, og deretter øke denne hvis det ser ut til å oppstå problemer i form av utstabilitet eller unøyaktighet.

Diskretiseringsmetode er enten Euler eller eksponentialrekke- utvikling med 4 ledd. Dette kan velges fritt for hver enkelt modul.

Tallrepresentasjon er 32- bit flyttall, valgt på grunnlag av avveining mellom nøyaktighet og krav til lagringsplass.

For å få simulert et komplett reguleringssystem kjører programmet prosessmodell, estimator, regulator og logmoduler som selvstendige enheter. En global klokke styrer tiden, med mulighet for valg mellom reell tid eller så raskt som maskinen greier å regne. Objektorientert oppbygning medfører at det vil være lett å implementere nye moduler spesielle for dette programmet, eksempelvis en ny regulator. Dette gjøres ved å lage en ny regulator som arver egenskapene til den opprinnelige regulatoren, og så lage en ny metode for selve reguleringsalgoritmen.

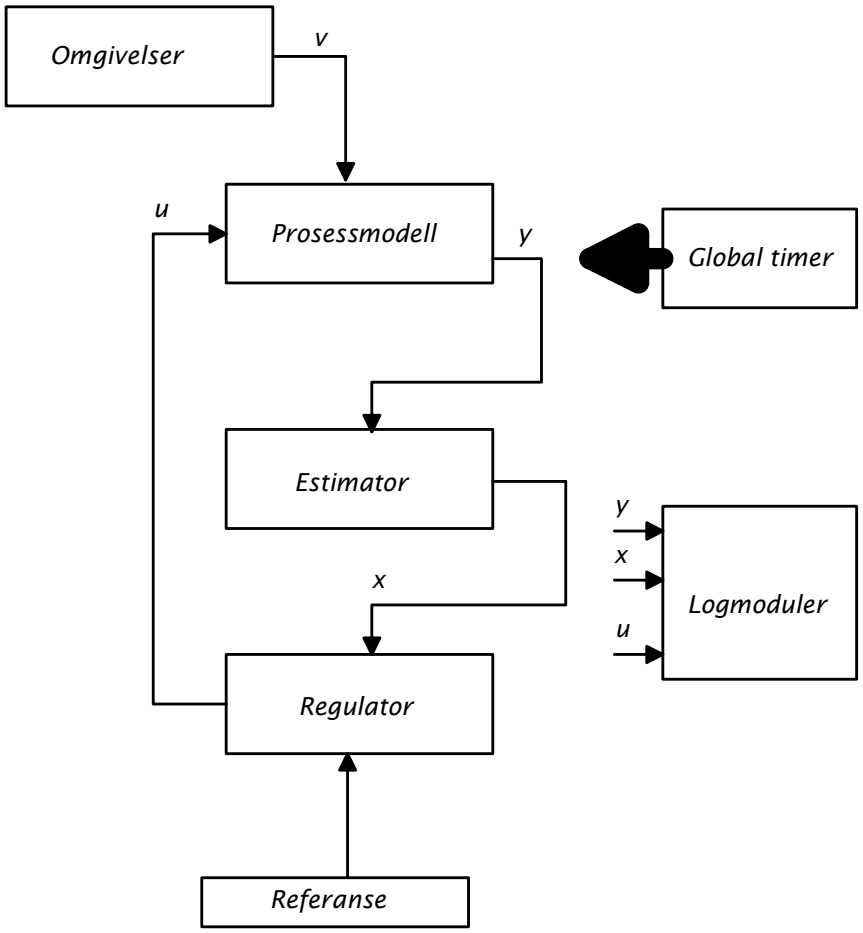
Simuleringsprogrammet kan simulere OEV- fartøyet med påvirkning fra de eksterne forstyrrelsene bølger og vind, estimering av tilstandsvektor og regulering.

Det er lagt vekt på at en skal kunne observere fartøyets tilstander mens simuleringen kjører. Dette gjøres via logmoduler som kan startes etter ønske. Logmodulene kan vise fortløpende verdi på opptil 8 tilstander i et grafisk plot. I tillegg kan vind og bølger observeres grafisk i omgivelsesmodulens vindu.

Objektet OEVkonsoll viser en grafisk representasjon av OEV- fartøyet, slik at fartøyets bevegelser kan observeres kontinuerlig.

Programmet er beregnet på plattformen maskin med intel - prosessor i486DX og MS- DOS operativsystem. Programmet kan kjøres under MS- Windows 3.1 og OS/2.

Figur 4.4.1: Simuleringsprogram, blokkskjema



4.4.2. Programmeringsteknisk beskrivelse

Simuleringsprogrammet er bygd opp rundt en kjerne som håndterer kjøring av parallelle prosesser. Programmet er implementert med Borland TurboPascal 7.0. Det kjører i grafisk modus, VGA 18 (640x480 punkter, 16 farger) og benytter XMS-memory via XMS- driveren HIMEM.SYS. Ved valg av programmeringsspråk er det lagt vekt på følgende faktorer:

v0 Ønske om å kunne anvende allerede ferdig programkode for håndtering av grensesnitt og grafikk.

v1 Kompilator må gi effektiv kode for kjøring på valgt maskinplattform.

v2 Mulighet for linking av ekstern assemblerkode.

v3 Mulighet for objektorientert programmering.

v4 Rask kompilering.

v5 Fleksibel implementering (syntaks).

v6 Gruppemedlemmenes kompetanse.

Valget stod mellom C++ og TurboPascal. TurboPascal gir særdeles effektiv kode for Intel- maskiner som kjører MS- DOS, faktisk bedre enn de fleste C- kompilatorer. Syntaks og fleksibilitet er nesten på høyde med C++, og en hadde ferdig mye Pascal-kode for grafisk brukergrensesnitt. Hvis en skulle skrive all koden om igjen, ville en nok valgt C++, ikke minst fordi en da lettere kan skifte til en annen plattform. Ellers er det klart at programmeringen ikke først og fremst består av implementering i et bestemt språk, men mest systemering og modellering.

Programmet har en objektorientert oppbygging. Det medfører store fordeler ved håndtering av større og komplekse programsystemer. I C++ og Pascal kan en blande objekter og 'vanlig kode' om hverandre. Enkelte hevder at dette er uheldig, men det er ingen tvil om at dette gir større fleksibilitet enn rene objektorienterte språk.

I TurboPascal kan en dele opp store prosjekter i flere UNIT'er. Disse kan sammenlignes med header- filer og bibliotek i C. I dette prosjektet har en brukt mange UNIT'er som allerede var ferdig (Det viste seg etterhvert at en måtte endre og forbedre på mye i disse enhetene, men tanken var i utgangspunktet at dette var ferdig kode klar til bruk).

Objekter for simulering av kybernetisk system og simulering av OEV- fartøyet lages for dette prosjektet.

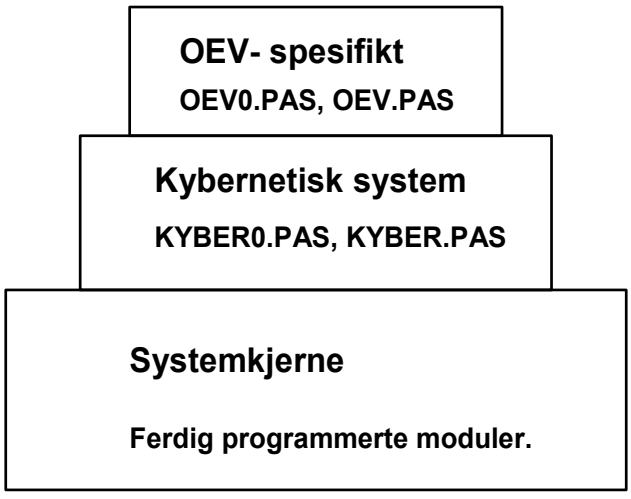
En hadde planer om å lage et C++- program som kunne simulere OEV- fartøyet tredimensjonalt i sann tid. Dette skulle vært et eksternt program som fikk data om OEV- fartøyet (modell) fra hovedprogrammet. Grunnet prosjektets omfang og begrenset nytteverdi av slik simulering ble denne delen skrinlagt.

En ville ha et brukervennlig grafisk grensesnitt, med hjelp tilgjengelig for de mest kritiske funksjoner. Allerede ferdige objekter for knapper, vinduer, listemenyer, tekstbokser og hjelp brukes. En benytter Borlands grafiske driver for VGA- modus. Den er treg og ikke særlig pålitelig, men en har ingen gode alternativer på dette tidspunkt. En del grafikkfunksjoner er erstattet med 32- bit assemblerkode (da går det unna!) som er både pålitelige og raske (Se filen BMPSWP.ASM).

En ekstern ressursfil inneholder hjelpetekster, konfigurasjonsfil og grafiske bilder. Denne lages med programmet SDU.EXE. Da får en samlet alle data programmet benytter seg av i en fil, og en har enkel aksess til bilder og konfigurasjonsdata via allerede ferdige objekter. SDU.EXE kan også linke ressursfilen til programfilen, slik at hovedprogrammet da vil bestå av bare en fil.

Programmet kan betraktes på tre hierarkiske nivå: Systemkjerne, kybernetisk system og OEV- spesifikt. Systemkjernen er objekter og funksjoner som håndterer hardware- og plattformavhengige funksjoner, slik som tastatur, memory, grafikk. Kybernetisk system bygger videre på systemkjernen, og innfører funksjoner som muliggjør simulering av tilstandsmodeller. OEV- spesifikke objekter bygger på det kybernetiske systemet.

Figur 4.4.2.: Programmets hierarkiske struktur



4.4.3. Programfiler

Følgende filer utgjør det ferdige kjørbare programmet:

OEV.EXE: Hovedprogram. MS- DOS exe- format.

OEV.SDU : Ressursfil med hjelp, konfigurasjonsdata og grafiske bilder for hovedprogrammet OEV.EXE.

Datafiler som benyttes av programmet:

*.MOD: Datafil for objekt OEVProsess. Filnavn velges i programmets fil- dialogboks.

*.EST: Datafil for objekt OEVEstimator. Filnavn velges i programmets fil- dialogboks.

*.REG: Datafil for objekt OEVRegulator. Filnavn velges i programmets fil- dialogboks.

*.OMG: datafil for objekt OEVOmgivelser. Filnavn velges i programmets fil- dialogboks.

*.LOG: Logmodul datastruktur. Filnavn velges i programmets fil- dialogboks.

*.PRN: ASCII- format datafil. Inneholde matriser for import i programmets matriseobjekter eller log- data fra en simulering. (Ekstensjonen PRN er valgt for å få kompatibilitet med programvaren MathCad).

Følgende TPU- er (Turbo- Pascal- Unit) anses som ferdig på det tidspunkt prosjektet startes:

GCRT.PAS: Prosedyrer og objekter for håndtering av tastatur og tid.

BGISYS, BGISYS2: Prosedyrer for linking av grafikkdrivere og fonter.

COMM.PAS: Objekter og prosedyrer for håndtering av peker, grafikk, bilder fra ressursfilen. Kommunikajon mot peker og tastatur. XMS- memory.

CTRL.PAS: Objekter og prosedyrer håndtering av kommunikasjon mellom objekter. Knapper, listemenyer.

WINS.PAS: Objekter og prosedyrer for vinduer, håndtering av prosesser, skjerm. Flere knapper, tekstbokser.

WINUTILS.PAS: Hjelp og fillistevindu.

INTRRUPTS.PAS: Prosedyrer for håndtering av tastaturinterrupt og Break- interrupt. Statusvindu.

Følgende TPU- er er utviklet i forbindelse med prosjektet:

KYBER.PAS, KYBER0.PAS: Objekter for simulering av kybernetisk system. Matriseobjekter.

OEV0.PAS: Objekter for simulering av OEV- fartøy. Klokkegenerator. Hovedvinduet: OEVmain.

Hovedprogrammet:

OEV.PAS: Hovedprogram.

4.4.4. Programmets ressurskrav

For å få kjørt programmet må følgende krav til datamaskinen tilfredsstilles:

v7 Intel- prosessor, minimum i386 og flyttallsprosessor i387. Anbefalt i486DX/ 33.

v8 VGA skjerm og skjermkort som kan kjøre standard VGA- mode 18, 640x480 /16- farger.

v9 Harddisk må ha nok ledig kapasitet til å ta unna log- data. Typisk 10- 20Mb. Ingen særlige krav til ytelse.

v10Operativsystem MS- DOS v. 3.30 eller nyere. Programmet kjører fint under MS- Windows 3.1. Programmet kjører særdeles bra under OS/2 v. 2.1.

v11Minimum 580Kb ledig DOS- memory. Fordelaktig med så mye som mulig.

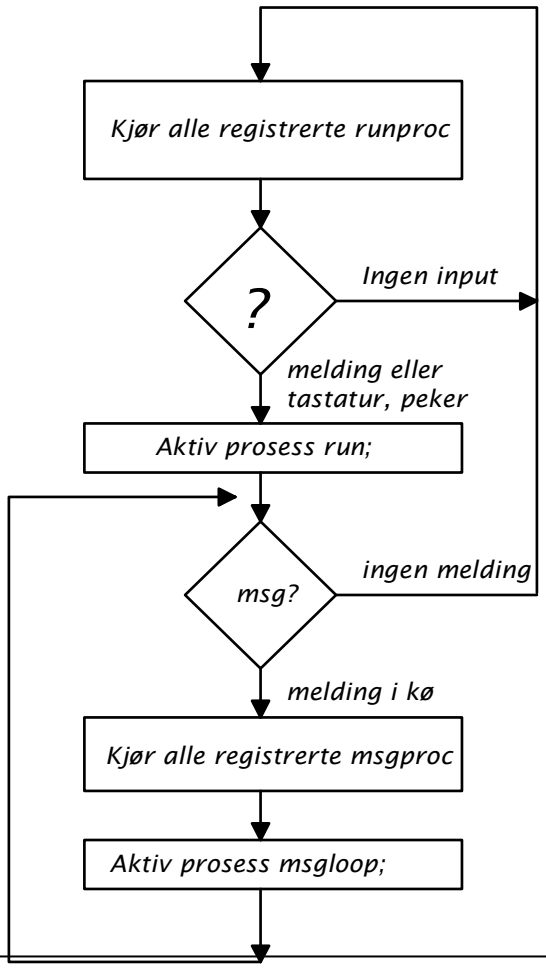
v12XMS- memory- manager HIMEM.SYS eller tilsvarende kompatibel driver. Fordelaktig med 10- 20 Mb XMS.

v13Microsoft- kompatibel peker (mus) med driver, eks. Microsoft MOUSE.SYS v. 7.04.

4.4.5. Systemkjernen

Systemets kjerne er de objekter og prosesser som håndterer meldinger mellom objekter, tastaturinput, pekerinput, aktivisering og terminering av prosesser, timeren, kjøring av prosesser.

Figur 4.4.5.1.: Metode windowhandler.run



Objektet winhandler, en instans av windowhandler- klassen, styrer de andre prosessene. Den står og kjører i en uendelig sløyfe i metoden run, helt til flagget systemshutdown blir satt. Da går en ut av sløyfen, og programmet termineres.

Kjøring av prosesser foregår slik at alle prosessenes aktiviserte runproc- metode står og kjører helt til et av flaggene kbdevflag eller pdevflag blir satt. Da kjøres den aktive prosessens metode run. Metoden run inneholder kjøring av alle knapper o.l. som er styrt fra tastaur eller peker. Knapper o.l. generer en melding som håndteres av mainloop. Hvis det etter kjøring av den aktive prosessens run - metode er en melding tilgjengelig i mainloop, startes en sløyfe som først kjører alle prosessenes metode msgproc, deretter den aktive prosessens msgloop. Slik kan alle prosessene motta meldinger fra den aktive prosessen.

Alle vinduer og prosesser må registreres i winhandler. Dette gjøres ved å la alle vinduer og prosesser arve egenskapene til klassen smallappwin, denne kaller opp en metode i winhandler og sender sin egen adresse over til denne. Ved å lage virtuelle funksjoner for subklasser kan winhandler nå kjøre metodene run, runproc, activate, deselect, msgproc og msgloop for det objektet som er registrert, og det er alltid den metoden som hører til den registrerte instansen som kjøres. Registrerte prosesser legges i en kø (en simpel lenket liste), og hver enkelt prosess får sin egne unike nøkkel (handle). Prosesser kan kommunisere med hverandre via denne nøkkelen.

Winhandler har tre lokale stacker: activestack, runstack og msgstack. Activestack brukes for å holde rede på alle registrerte vinduer. I runstack ligger nøkkel til alle prosesser som har registrert metode runproc, den blir da kjørt kontinuerlig i sløyfen mens en venter på meldinger eller brukerinput. I msgstack ligger nøkkel til alle prosesser som har registrert msgproc, den blir da alltid kjørt før den aktive prosessen henter meldinger ut fra køen.

Objektet mainloop, en instans av klassen eventhandler, håndterer meldinger fra knapper, listemenyer, andre meldingsgenererende objekter. Meldingene blir lagt i en FIFO- kø. Alle sløyfer i objekter og prosesser som venter på hendelse, eks. tastaturinput, kjører metoden nocomm. I nocomm aktiviseres blant annet hjelp og statusvindu hvis brukeren ber om det. Dette er gjort slik fordi mange objekter er av typen som kjører i en intern uendelig sløyfe, slik at winhandler eller andre prosesser ikke slipper til før objektet termineres. Tekstboksene er et eksempel på et slikt objekt. En har også større grad av fleksibilitet ved å gjøre det slik, en kan lage et vindu som opprettes, utfører en jobb og termineres, uten å registrere dette som en prosess. Hvis systemet skal brukes som et sanntidssystem med håndtering av tidskritisk ekstern kommunikasjon kan en ikke gjøre det slik, men for et simuleringsprogram spiller det egentlig ingen rolle.

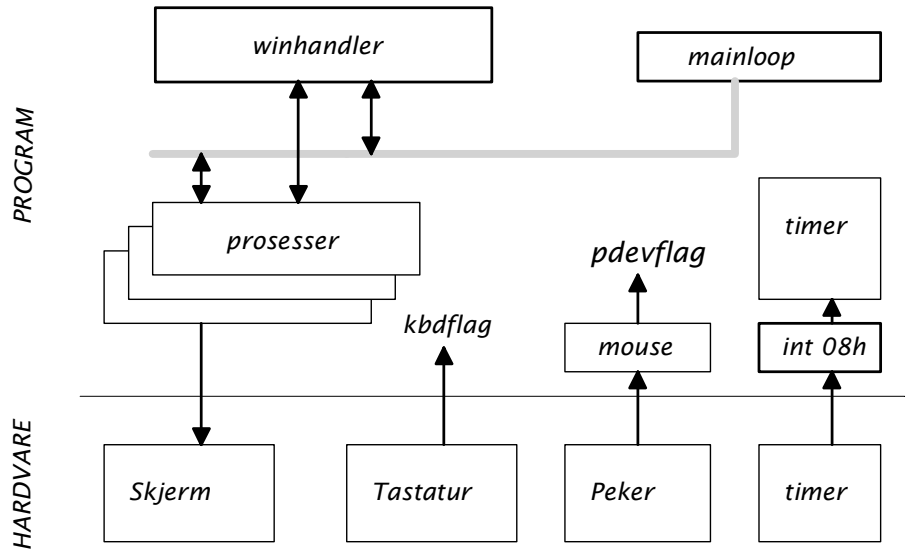
Tastaturinterrupt int 09h styrer et flagg som signaliserer tastetrykk og leser tastetrykk inn i den globale strukturen keybuff.

Peker håndteres av instansen mouse av typen gmouse. Denne installerer en interruptrutine, mousehandler, som kalles opp automatisk av pekerdriveren når pekeren beveges eller en knapp trykkes ned. Pekerdriver må være kompatibel med Microsoftstandarden.

Timeren er styrt fra maskinens timerinterrupt int 08h. Hardvare- timeren omprogrammeres, slik at en får firedoblet frekvensen. Timeren oppdaterer de globale variablene timer1, timer2 og global32timer 72 ganger pr. sekund (Fil: X4TIMER.ASM).

Kildekode for typedefinisjoner, globale variabler og prosedyrer i systemet finnes i appendix E.

Figur 4.4.5.2.: Systemkjerne



4.4.6. Kyber- systemet

Modulene KYBER0.PAS, KYBER.PAS inneholder objekter og prosedyrer som håndterer simulering av tilstandsmodeller.

For å få til sanntidssimuleringer må en ha referanse til en klokke. Dette er gjort ved å hekte seg inn på timer- interruptet, int08h. Dette interruptet blir automatisk kjørt med en frekvens på 18.2 Hz. Timeren omprogrammeres slik at frekvens firedobles til 72Hz. Dette blir da det minste målbare tidsintervall for sampletider (Se X4TIMER.ASM for detaljer). Objekter kan likevel kjøre med høyere samplingsfrekvens internt, dette vil være aktuelt for prosessmodeller og estimatorer som trenger høy samplingsfrekvens for å unngå ustabilitet. En komplett reguleringssløyfe må imidlertid kjøre på 72Hz eller lavere, fordi signaler mellom objektene ikke kan oppdateres raskere.

De forskjellige objektene må ha mulighet for å sende signaler til hverandre. Dette løses ved å opprette en vektorlink mellom to objekter. Objektene har evne til selv å holde vedlike linkene.

Generering av stokastiske signaler kan foretas med funksjonen Gauss.

Objektet objtimer

Timer som returnerer alarm når gitt tidsintervall er overskredet. Brukes til å holde rede på samplingsintervaller.

Objektet stdmatrix

Base objektklasse for matriser og vektorer. Håndterer addressering av elementer og allokering av memory.

Objektet f32matrix

Matrise for element av typen 32- bit flyttall. Har evne til å utføre matrise regneoperasjoner.

Objektet f32vector

Matrise med en kolonne, ellers samme egenskaper som f32matrix.

Objektet vectorwin

Vektor med brukergrensesnitt. Muliggjør inspeksjon av elementer og inntasting av nye verdier.

Objektet importvectorwin

Vektor med brukergrensesnitt og mulighet for link til vektorer i andre objekter.

Objektet wnvectorwin

Vektor med brukergrensesnitt som returnerer hvit støy med midlere vekt lik intastet verdi.

Objektet matrixwin

Matrise med brukergrensesnitt. Muliggjør inspeksjon av elementer, inntasting av nye elementer og innlesing av matrise fra fil i ASCII- format.

Objektet stdKybwin

Baseobjektklasse for alle andre simuleringsobjekter. Holder rede på kobling til objektets datafil og opprettholder vektorlinker.

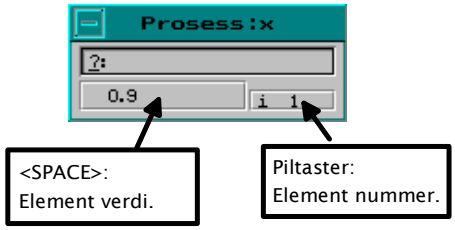
Objektet stdProsess

Simulerer en standard lineær prosessmodell. Kan også simulere ulineære systemer. Da må en deklarere et nytt objekt for den spesifikke prosessmodellen, der en implementerer nye metoder for beregning av matrisene A, B, C på grunnlag av tilstandsvektorene.

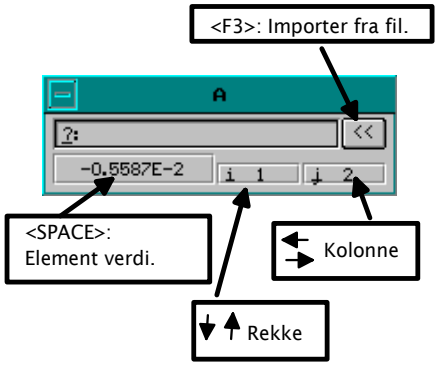
Diskretiseringsmetode og samplingsfreksvens er valgfri.

Objektet stdLogmodul

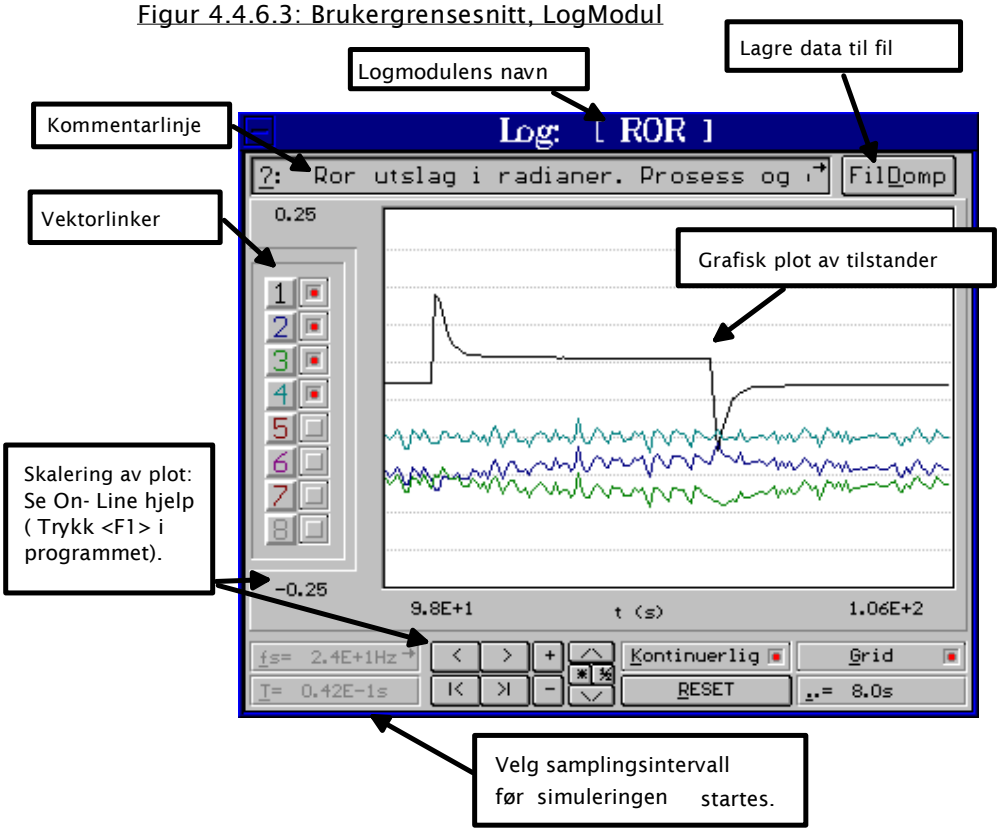
Figur 4.4.6.1: Grensesnitt, vektorwin



Figur 4.4.6.2: Grensesnitt, matrixwin



Figur 4.4.6.3: Brukergrensesnitt, LogModul



Lagrer kontinuerlig tilstander til XMS- memory. Tilstander kan vises kontinuerlig i skjermplottet. Tilstander kan lagres til fil i ASCII- format.

Skjermplottet er lett å arbeide med fordi en raskt kan flytte seg langs aksene i plot-tet.

Samplingsfrekvens er valgfri.

Objektet stdEstimator

Tilstandsestimator med mulighet for kontinuerlig oppdatering av optimal K (Kalman- filter). Algoritmen er beskrevet i kapittel 3.3: Teori, tilstandsestimering.

Diskretiseringsmetode og samplingsfekvens er valgfri.

Objektet stdRegulator

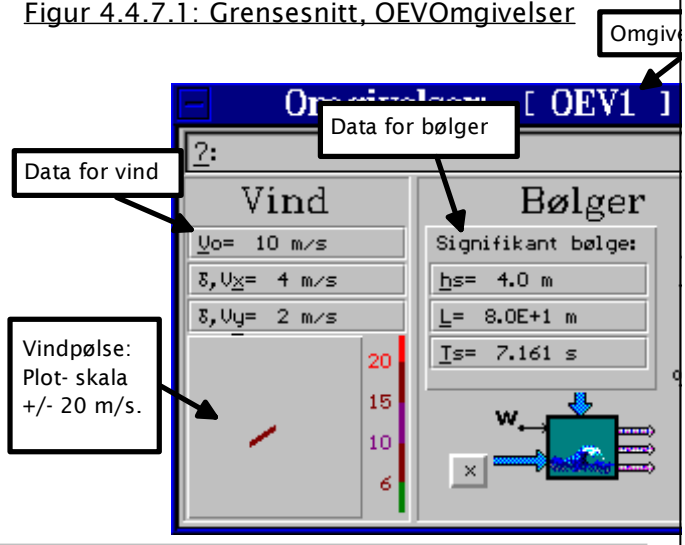
Implementerer en enkel regulator med fast tilbakekoblingsmatrise.

Ved å overlagre metoden calcgGmatrix i et nytt objekt kan en lage mer avanserte adaptive regulatorer.

4.4.7. OEV- programmet

OEV0.PAS inneholder de objektene som er utviklet i forbindelse med dette prosjek-

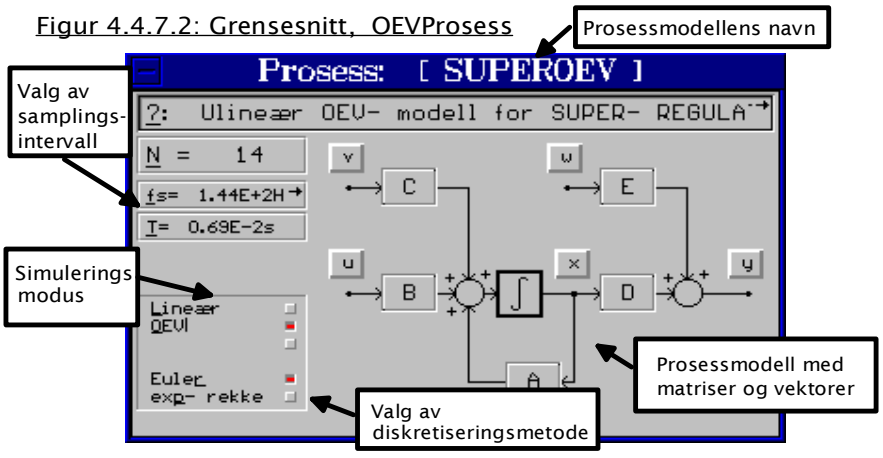
Figur 4.4.7.1: Grensesnitt, OEVOmgivelser



tet.

Objektet ctrlwin

Figur 4.4.7.2: Grensesnitt, OEVProsess



Styrer systemets tid. Resetter timer- flagget ved hvert gjennomløp (runproc) slik at X4TIMER- interruptet kan fortsette oppdatering av timeren. En kan velge mellom sanntid eller full fart (En ser forøvrig at skifte mellom sanntid og 'full- fart' ikke har innvirkning ved simulering av mer komplekse systemer fordi maskinen ikke greier å regne raskt nok).

Objektet OEVOmgivelser

Simulerer OEV- fartøyets omgivelser, vind og bølger. Stokastiske metoder nyttes for å få mest mulig realistiske støysignaler.

Vind og bølgerforhold er valgfrie.

Grafisk visning av vindvektoren og bølgehøyde ved fartøyets foilenheter.

OEVOmgivelser linkes til x- vektor i OEVProsess.

Objektet OEVProsess

Overlagrer metodene calcgDmatrix, calcgOmatrix, calcgFmatrix, runproc slik at Prosessmodul kan simulere OEV- fartøyet som en ulineær prosess.

En kan velge mellom ulineær og lineær simuleringsmetode. Den ulineære simuleringen gjelder bare for OEV- fartøyet, ettersom det er ligningene for OEV som er implementert i programmet.

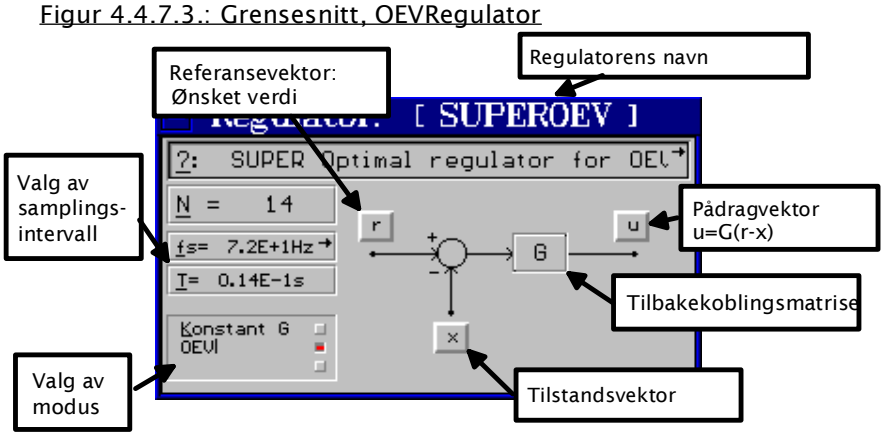
Objektet OEVRegulator

Overlagrer metodene runproc, calcgGmatrix slik at en får en adaptiv regulator for OEV- fartøyet.

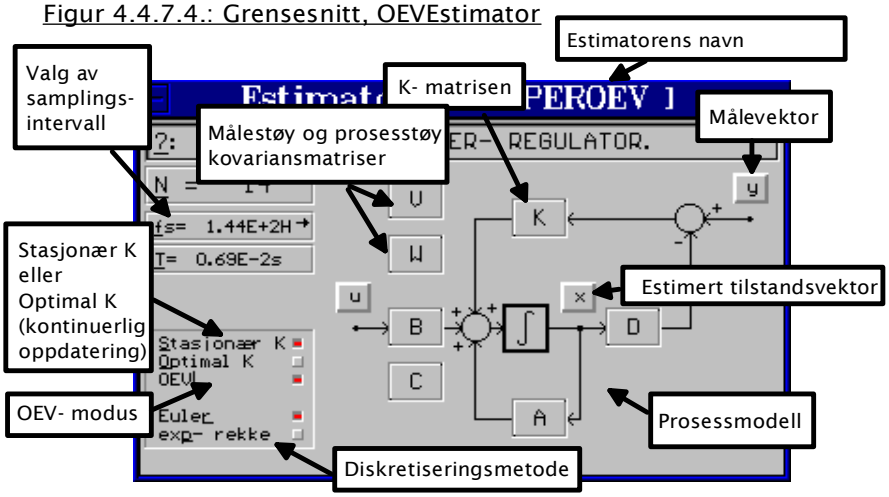
Objektet OEVEstimator

Overlagrer metoden runproc i stdEstimator.

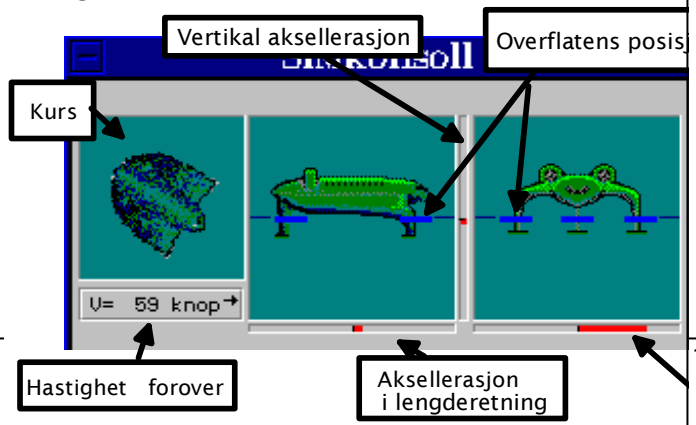
Figur 4.4.7.3.: Grensesnitt, OEVRegulator



Figur 4.4.7.4.: Grensesnitt, OEVEstimator



Figur 4.4.7.5.: Grensesnitt, OEVkonsoll



Objektet OEVkonsoll

Viser en grafisk representasjon av OEV- fartøyets bevegelser, sett fra siden, ovenfra og forfra. Stopper simuleringen hvis fartøyet havarerer, slik at programhavari grunnet flyttallsfeil unngås. Akselleresjon i lengde- side- og høyderetning vises kontinuerlig i bar- display, skala -0.5g - +0.5g.

OEVkonsoll oppretter automatisk link til x- vektor i OEVProsess.

Den grafiske animasjonen er laget ved at en tegnet OEV- fartøyet i forskjellige vinkelposisjoner, sett ovenfra, fra siden og forfra. Bildene ble lagt inn i SDU- filen slik at OEVkonsoll får tilgang til disse. Ut i fra informasjon i fartøyets tilstandsvektor velges visning av det bildet som ligger nærmest den aktuelle tilstanden. Animasjonen blir litt hakkete fordi det er et begrenset antall bilder til rådighet.

Objektet mnuwin

Hovedvindu for OEV- programmet. Håndterer oppstart av moduler, hovedmenyer, terminering av moduler.

4.4.8. Simulering og analyse av vilkårlig system

Selv om programmet er laget spesielt for simulering av OEV- fartøyet, kan Prosess-modulen også simulere et hvilket som helst annet lineært system med opptil 126 forskjellige tilstander. Dette utnyttet vi for å finne programmets regnenøyaktighet og eventuelle regnefeil. Ved å simulere et system med kjent oppførsel og så studere programmets resultater får en kontrollert hvorvidt prosessmodul og logmodul fungerer som forventet.

Hjelp- funksjonen (Trykk høyre peker- knapp eller funksjonstast <F1>) er mangelfull, men en kan få nyttig informasjon om brukergrensesnittet, spesielle taste- kombinasjoner og hvordan en beveger seg i plottet i LogModul.

Framgangsmåte for simulering av et lineært system:

36. En starter med å skrive inn data for prosessen i prosessviduet der utgangspunktet er en generell tilstandsmodell. Elementene i matrisene A, B, C, D skrives inn eller hentes fra fil, samplefrekvens velges, diskretiseringsmetode velges. Dette resulterer i en *.MOD- fil når prosessmodellen lagres. En må passe på at opsjonen Lineær er aktivisert.
37. Åpne en logmodul og link vektorer for de tilstandene en vil studere. Når samplefrekvens og vektorlinker er bestemt, kan en lagre til ny *.LOG- fil.
38. Start simuleringen med klokkegeneratoren. Pass på tilstander i prosessen, hvis systemet er ustabilt får en overflow i flyttallsprosessoren, og programmet totalhaverer. Hvis en får uventede problemer med ustabilitet, kan en prøve å endre samplefrekvens for prosessen, eller benytte exp- rekke diskretiseringsmetode.
39. Hvis en vil analysere simuleringen mer grundig, kan en lagre tilstandsdata til ASCII- fil fra logmodulen. Velg ekstensjon *.PRN hvis filen skal importeres i MathCad.
40. Når programmet termineres lagres desktop- layout automatisk, slik at neste gang programmet startes slipper man å åpne objektene på nytt.

Filene FJEDR_3.MOD, FJEDRING.LOG, FJEDR3.PRN, FJEDR_3.MCD viser simulering av et fjæringssystem for bil, der veien er modellert inn i prosessmodellen (Se datafilen FJEDRING.ZIP). Eksempelet er hentet fra Stochastic Systems for Engineers [ref. 24.], s. 93- 95. I filen FJEDR_3.MCD har en foretatt en analyse av bilens oppførsel i frekvensplanet, der en tydelig ser effekten av karosseriets og den uavfjærede massens resonansfrekvenser. Resultatene stemmer godt overens med eksempelet i boken.

4.4.9. Simulering av OEV- fartøy

Ved simulering av OEV- fartøy må følgende utføres i tillegg (ref. pkt. 4.4.8.):

41. Matrisene i OEV- modellen er store. Det er en fordel å lage disse ferdig i forbindelse med modelleringsarbeidet, enten i MathCad eller MatLab. Matrisene lagres til fil i ASCII- format, slik at OEV- programmet kan importere matrisene direkte.
42. Velg opsjonen OEV! i prosessmodulen. Velg diskretiseringsmetode Euler, ellers går simuleringen altfor tregt. Sett tilstandsvektoren til passende verdier før simuleringen startes, for eksempel hastighet=80 knop, høyde= 2.9m.
43. Åpne OEVkonsoll, slik at fartøyets bevegelser kan observeres. Stopp simuleringen hvis fartøyet kommer ut av kontroll, ellers risikerer en at programmet havarerer. OEVkonsoll vil automatisk stoppe klokkegeneratoren når havari detekteres, men det viser seg at denne opsjonen er upålitetelig.
44. Simuler bølger og vind med omgivelses- modulen.
45. Regulator åpnes hvis simulering av regulert fartøy ønskes. Velg opsjonen OEV!. Sett inn passende verdier i referansevektoren, for eksempel høyde=2.9m.
46. Åpne Estimator hvis estimering av tilstandsvektor ønskes. Velg opsjonen OEV! og pass på at diskretiseringsmetode er Euler. Ved å definere matrisene V, W kan optimal K beregnes med opsjonen Optimal K.
47. God fornøyelse.

4.5. REGULERING

Krav til regulering. Regulatorens struktur. Simulering som verktøy for å finne parametre til regulatoren.

4.5.1. Krav til reguleringen

Regulatoren skal, ved å gi pådrag til aktuatorer som styrer vinkler på foilene, kontrollere fartøyet under påvirkning fra eksterne forstyrrelser, bølger og vind. Følgende krav til regulering spesifiseres:

w21 Fartøyet skal være stabilt slik at ingen tilstander antar ukontrollerte verdier over tid.

w22 Fartøyets høyde over sjøens overflate skal holdes på et referansenivå slik at skroget alltid er over overflaten og foilene alltid er under overflaten.

w23 Fartøyets xy- plan skal holdes parallelt med referansesystemets (verden) xy-plan.

w24 Fartøyets bevegelser skal være så små som mulig slik at best mulig komfort oppnås.

w25 Fartøyet skal kunne svinge (rotasjon rundt fartøyets z- akse) kontrollert med styring fra et referansesignal som angir rotasjonshastighet.

w26 Regulatoren skal ikke styre motorenes skyvekraft.

Med utgangspunkt i tilstandsmodellen for fartøyet blir kravene slik:

w27 Alle egenverdier for regulert system > 0.

w28 $x_1=Z_g=h_{ref}$.

w29 $x_5=\phi=0$, $x_6=\psi=0$.

w30 Tidsderiverte av Vx, Vy, Vz, ϕ, ψ, θ holdes så lavt som mulig.

w31 $x_{10}=\omega\theta=\omega\theta_{ref}$.

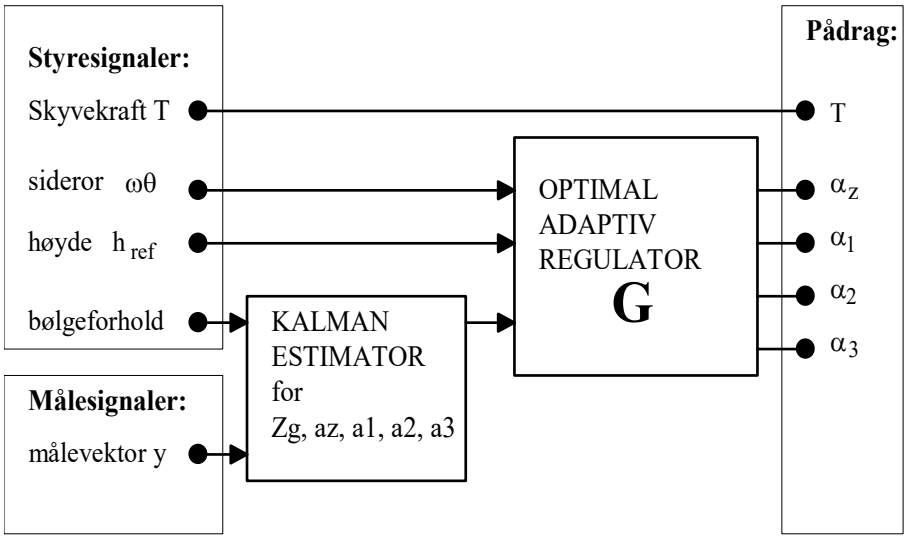
4.5.2. Regulatorens struktur

En modal analyse av linearisert tilstandsmodell [Appendix A: EST.MCD] viser at fartøyet er ustabilt: flere modi er ukontrollerte, men systemet er både styrbart og observerbart. Da er det mulig å lage en regulator som kontrollerer fartøyet slik at stabilitet for alle modi oppnås. Dette kan gjøres ved å innføre en tilbakekoblingsmatrise, G, mellom målevektoren y og pådragsvektoren u. To metoder for å finne denne matrisen er aktuelle: Modalregulering som baserer seg på forflytning av egenverdier, eller optimalregulering. Optimalregulering har den fordelen at en kan spesifisere optimalvekter for de enkelte tilstander og kostnadsvekter for de enkelte pådragene. En står da fritt til å prioritere hvilke tilstander som er mest kritisk å holde fast, og pådragene kan holdes innenfor rimelige grenser ved å velge kostnader som står i forhold til utstyringsområdet. Ulemper med optimalregulering er at en har liten kontroll over hvordan egenverdiene for det regulerte systemet blir, og det finnes ingen enkle metoder for bestemmelse av optimalvekter og kostnadsvekter.

Fartøyets dynamiske egenskaper endrer seg som funksjon av de aktuelle tilstander, derfor er det nødvendig å ha en form for adaptiv regulator der tilbakekoblingsmatrisen oppdateres kontinuerlig med en algoritme som tar utgangspunkt i tilstandsvektoren x.

Tilbakekoblingsmatrisen G krever måling av alle tilstander i tilstandsvektoren x. Dette oppnås med en optimal tilstandsestimator (Kalman- estimator). Fartøyets høyde over overflaten og foilenes egentlige utslagsvinkel må estimeres.

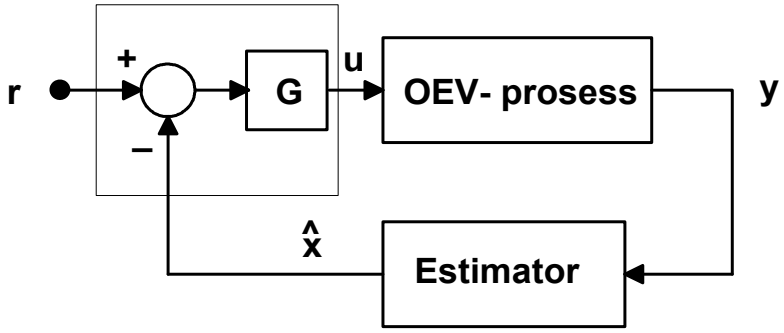
Figur 4.5.2.1.: Regulatorens struktur.



Estimatoren kan bruke informasjon om bølgeforhold til oppdatering av K- matrisen for å gi best mulig resultat under forskjellige forhold. Informasjon om bølgeforhold kan observeres av fartøyets fører eller fåes fra bølgevarsel på radio. En bedre måte er å la en egen estimator for bølgedata som ut i fra fartøyets bevegelser i forhold til målt overflateposisjon (z_1, z_2, z_3) beregne de aktuelle bølgeforholdene.

Skyvekraft, T, kobles fra referansevektor (r_2) direkte til pådrag u_1, slik at skyvekraft kan styres manuelt i simuleringsprogrammet fra referansevektoren.

Figur 4.5.2.2.: Regulator, blokkskjema



r: Referansevektor
- r_1: h_{ref}
- r_2: Skyvekraft, T
- r_{10}: $\omega\theta_{ref}$

u: Pådragsvektor
y: Målevektor
x: Estimert tilstandsvektor

4.5.3. Optimal regulator

Optimal G

Teori for optimalregulering av multivariable stokastiske systemer beskriver metode for beregning av tilbakekoblingsmatrisen G.

Ved valg av optimalvekter og hvilke tilstander som skal optimaliseres, må en foreta en vurdering av fartøyets grunnleggende fysiske oppførsel. En kan ikke ukritisk velge optimalvekt for alle tilstander og tro at det går bra. Med utgangspunkt i spesifikasjoner for regulatoren og viten om fartøyets egenskaper velges optimalvekter for fire tilstander:

Zg: Fartøyets høyde.
ϕ: Fartøyets dreining om x- aksen.
ψ: Fartøyets dreining om y- aksen.
$\omega\theta$: Fartøyets rotasjonshastighet om z- aksen.

Når disse tilstandene holdes under kontroll er krav til regulering oppfyllt. Tilstanden Zg holder også Vz under kontroll, Vy er naturlig stabil. Rotasjonshastighet om x- og y- akse er kontrollert når tilhørende vinkler styres. For rotasjon om z- akse må en velge rotasjonshastigheten som optimalkriterie for å kunne svinge fartøyet. Ettersom det ikke er noen enkel metode for beregning av verdier for optimalvekt, velges noen tall som utgangspunkt. Generelt så gir høyere tallverdi et raskere system, mens stabilitetsmarginen blir dårligere.

Regulatoren skal utnytte utstyringsområdet for pådragsorganene best mulig. Dette er utgangspunktet for valg av kostnadsvektmatrise for pådragene til foilvinkler. Stor kostnadskoeffisient gir lite pådrag. For å unngå pådrag til skyvekraft velges en høy kostnad for denne.

Nå kan en beregne matrisen G ved å løse Riccati- ligningen. Dette gjøres i MatLab med funksjonen OPTOEV [Appendix B: OPTOEV.M].

G- matrisens elementer for pådrag til skyvekraft (u_1) og referanse fra Vx (x_2) nullstilles slik at regulatoren ikke får gi pådrag til skyvekraft, og ikke blir påvirket av fartøyets hastighet framover. Hvis optimalvekter og kostnadsvekter er valgt med omhu, skal ikke disse justeringene ha nevneverdig innvirkning på regulert systems egenverdier.

En bruker MathCad [Appendix A: OPTOEV.MCD, SUPEROEV.MCD] for å organisere arbeidet med beregning av G. Utregning av egenverdier for systemet er første indikasjon på hvor vidt en er på rett vei. En enkel simulering med lineær modell gir informasjon om innsvingningsforløp for regulert system. Kostnadsvekt og optimalvekt justeres til en får et bra resultat på kurvene i MathCad- simuleringen. Deretter importeres G- matrisen i regulatoren til simuleringsprogrammet for utprøving. Ved å studere fartøyets oppførsel og pådrag til foiler kan en justere optimalvekt og kostnadsvekt, og beregne en ny G. Denne prosessen gjentas til en er fornøyd med resultatet.

I Appendix A:OPTOEV.MCD vises løsning for en optimal regulator som fungerer. Her har en koblet ut de elementer i G- matrisen som har kobling fra foilvinkler. Dette ble gjort fordi en ikke har tilgang til måling av foilvinkler direkte.

Ytterligere justering av vekter og utvikling av en bedre estimator gir en langt bedre regulator. Denne er vist i filen SUPEROEV.MCD.

Adaptiv G

OEV- fartøyet utgjør en ulineær prosess, derfor er det ønkelig å få oppdatert G- matrisen kontinuerlig på grunnlag av tilstandsvektoren. En ser at dette fort blir en særdeles komplisert oppgave å løse. En måte er da å ta utgangspunkt i de ulineariteter som gir størst problemer, og justere G- matrisen slik at en får kompensert for disse.

Ulinearitet i krefter fra foiler, som er en kvadratisk funksjon av hastigheten, har stor innvirkning på fartøyets dynamikk. Dette kan en tildels kompensere for ved å justere pådraget, slik at den kraft foilens utslagvinkel gir tilsvarer det en ville få hvis hastigheten var lik designfart. En får da:

$$F_0 = a \cdot v_0^2 \qquad \text{[R: 1.]}$$

$$F = a \cdot k \cdot v^2 \qquad \text{[R: 2.]}$$

Matrisene L , M tvinger x_2- x_{10} til å følge y_1 - y_9:

$L_{1,1}$=1
$L_{11,11}$..$L_{14,14}$=1

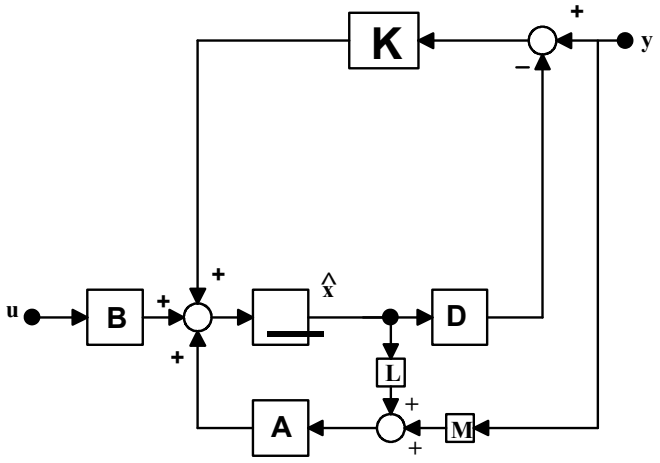
$$F_0 = F \Rightarrow k = (\frac{v_0}{v})^2 \qquad \text{[R: 3.]}$$

En ser at ved å skalere pådraget fra den opprinnelige G- matrisen med faktoren k oppnår en å få samme kraftpåvirkning ved et gitt beregnet pådrag fra G- matrisen over hele hastighetsområdet. Simulering vil vise om dette er en brukbar løsning.

4.5.4. Tilstandsestimator

En ønsker en tilstandsestimator der en kan benytte matriser fra analog (kontinuerlig tid) tilstandsmodell, derfor tar en utgangspunkt i en analog Kalman- estimator. Denne er beskrevet i kapittel 3.3.2.

Figur 4.5.4.: Estimator for Zg, α_2, α_1, α_2, α_3.



En antar at målestøyen for signalene y1 - y10 er lavere enn den feil en får ved å estimere disse tilstandene, derfor brukes disse målingene direkte, uten estimering eller filtrering. Da blir det mye enklere å få til en estimator som fungerer for Zg og foilvinkler, som ikke er direkte målbare. Ved å tvinge estimatorens x- vektor til å følge målingene fra y- vektoren blir feilen en får ved å bruke linearisert modell for A og B- matrise i estimatoren sterkt redusert.

Det er ønskelig å få estimater for foilvinkler til å følge det egentlige utslaget så eksakt som mulig.

Estimat for Zg skal være mest mulig støyfritt og samtidig følge sjøens overflate bra nok til at høyden kan holdes innenfor sikre grenser.

En benytter programvaren MathCad for å sette opp beregning av K- matrise for estimatoren, fil EST.MCD. Her foretas først en modal analyse av prosessmodellen, for å

kontrollere OEV- fartøyets observerbarhet og styrbarhet. Prosessmodell hentes fra linearisert OEV- prosess, som ble satt sammen i filen OEVMOD.MCD.

Kovariansmatrisen W for målestøy velges lik prosessmodellens E- matrise. Kovarians for prosesstøy, V, velges slik:

w32 Antar en midlere varians for vind. Dette gir elementene V_{11}, V_{22}.
w33 Antar en passende varians for bølger. Dette gir elementene V_{33}, V_{44}, W_{55}.

Løsning av Riccati- ligningen for beregning av K foretas i MatLab med funksjonen OEVEST2 [Appendix B: OEVEST2.M].

Egenvektorene for tilstandsestimatoren viser om estimatoren er stabil.

Videre finjustering av parametre foretas i simuleringsprogrammet. Ved å velge opsjonen Optimal K i estimatoren oppdateres K- matrisen automatisk. Større verdier i W-matrisen gir bedre filtrering og dårligere følging, mens større verdier i V- matrisen gir bedre følging.

For å få fartøyet til å følge bølgeformen ved lange bølgelengder og høye bølger kan en prøve å sette inn verdien h_s^2 (h_s= signifikant bølgehøyde) i kovariansmatrisen V, elementene V_{33}, V_{44}, V_{55}.

4.5.5. Implementering i programmet

Optimal tilbakekobling G og tilstandsestimator kobles sammen som i figur 4.2.2.2.

I OEV- regulatoren implementeres ulineær skalering av pådraget [OEV0.PAS, metode OEVRegulator.runproc]. Referansevektor element r_2 kobles over til pådragsvektor u_1., slik at en kan justere skyvekraft i r- vektoren. Elementet r_2 tilsvarer egentlig referanse for Vx, men denne har ingen kobling i G- matrisen og kan derfor brukes til dette formålet.

Standard estimator i programmet tar utgangspunkt i det analoge Kalman- filteret som diskretiseres , denne er beskrevet i kapittel 3.3.3.

Standard estimator endres slik at y- vektorens elementer y_1 - y_9 kopieres til x_2 - x_{10}. En har også justert for arbeidspunkt i linearisert tilstandsvektor slik at $y_{2,lin} = y_2 - Vx_0$.

Simulering av komplett reguleringssystem kan nå foretas ved å koble inn Regulator og Estimator mellom prosessens y- vektor og målevektor.

Ved valg av samplingsfrekvens må en ta hensyn til at tilstandsestimator og reguleringssløyfe har raskere egenverdier enn prosessmodellen alene. Samplingsfrekvens må økes for både estimator, regulator og prosess.

7. RESULTATER OG DISKUSJON

Resultater og vurdering av utførte aktiviteter.

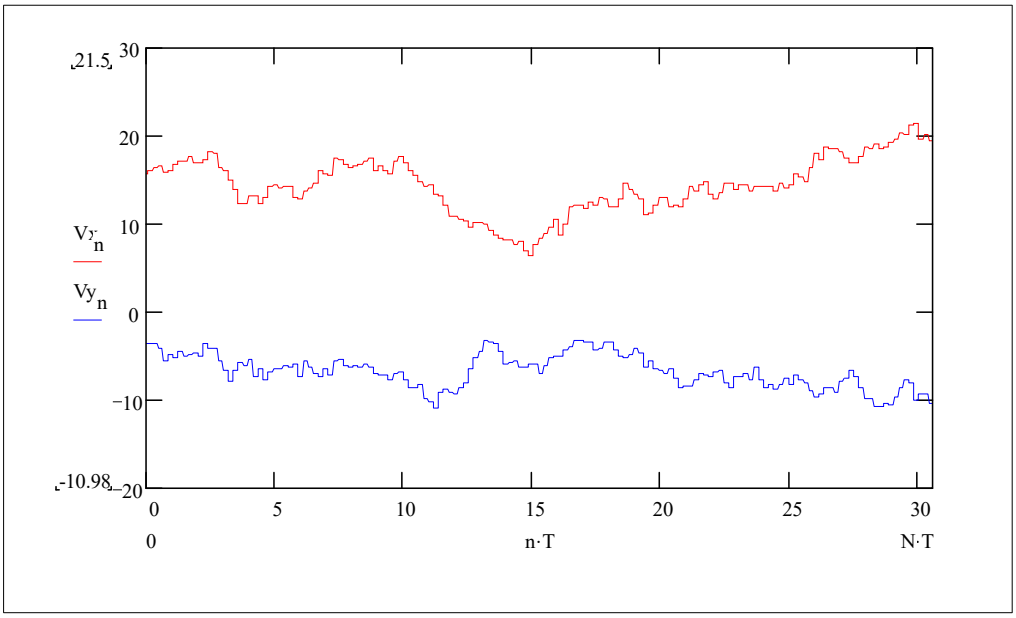
7.1. RESULTATER

Resultater fra simuleringer med simuleringsprogrammet OEV.EXE

7.1.1. Simulering av vind

Fartøyets hastighet :	Vx = 0 ms^{-1}.
Fartøyets retning :	θ = 0 rad.
Vindhastighet :	Vind, Vx_0 = 10 ms^{-1}.
Vind, variasjon x- retning :	ΔVx=4.
Vind, variasjon y- retning :	ΔVy=4.

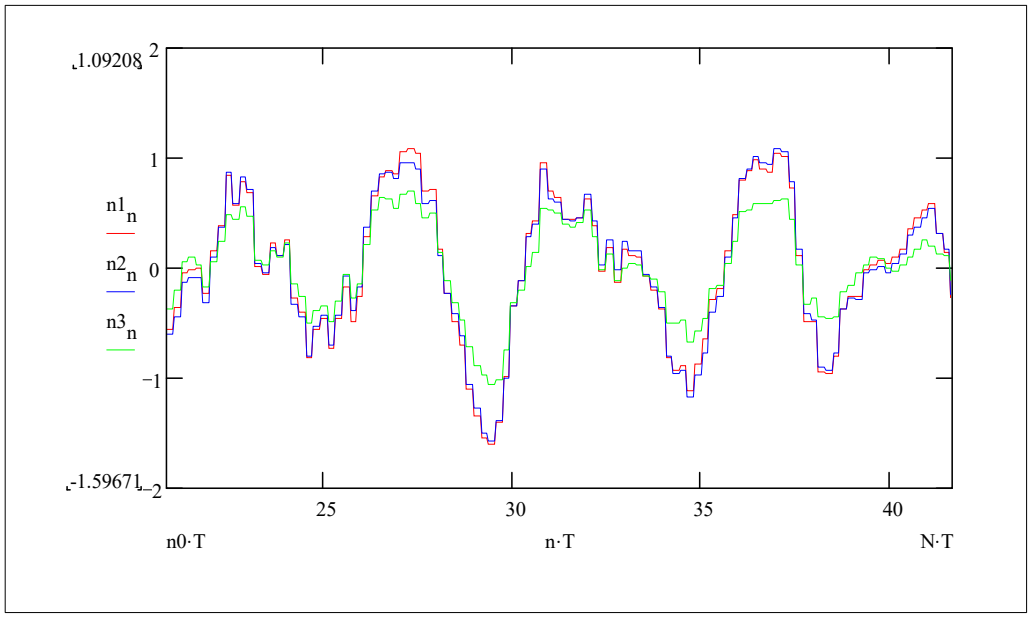
MathCad- plot av ASCII- domp fra logmodul:



7.1.2. Simulering av bølger

Fartøyets hastighet:	Vx = 10 ms^{-1}.
Fartøyets retning:	θ = 3 rad.
	(10 m/s mot bølgefronten)
Signifikant bølgehøyde	h_s = 1 m.
Signifikant bølgelengde	λ_s = 80 m.

MathCad- plot av ASCII- domp fra logmodul: Bølger ved fartøyets 3 foiler



7.1.3. Regulatoren OPTOEV

Regulator data

Regulator: OPTOEV.REG
Estimator: OEV2.EST
Prosessmodell: OEV.MOD

Samplingsfrekvens, regulator OPTOEV: 36 Hz.
Samplingsfrekvens, estimator OEV2: 72 Hz.
Samplingsfrekvens, prosess OEV: 36 Hz

Optimalvekten Q, Kostnadsvekten P, Tilbakekoblingsmatrisen G:
Se Appendix A: OPTOEV.MCD

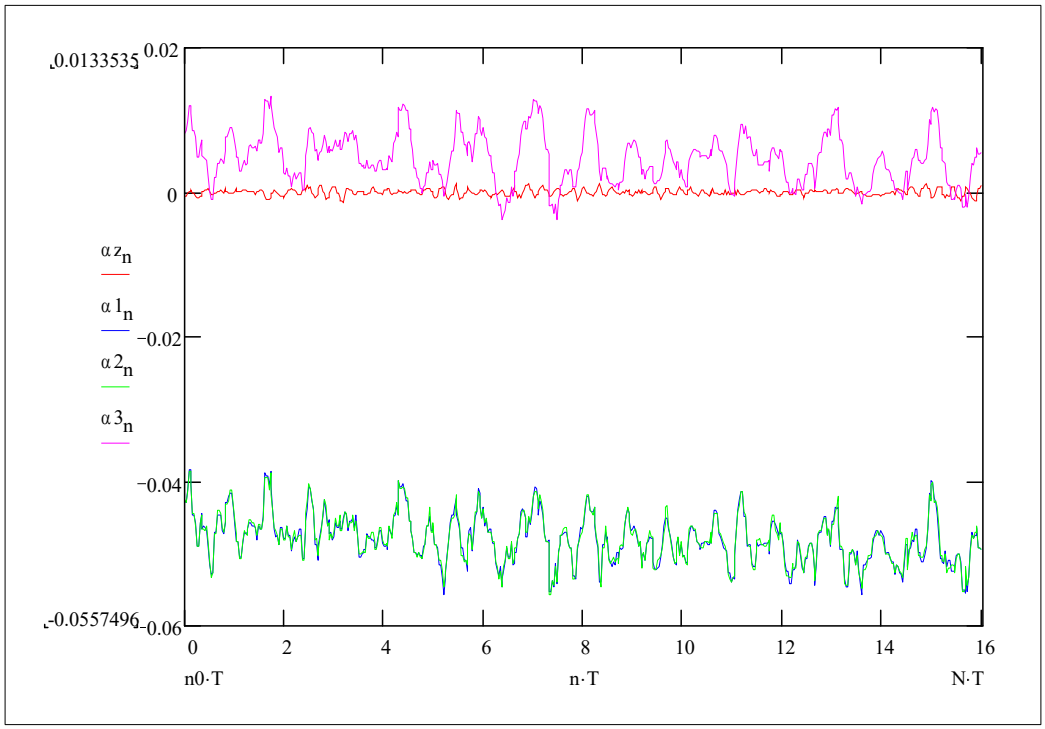
Estimator kovariansmatrise W, Estimator kovariansmatrise V, Estimator stasjonær K- matrise:
Se Appendix A: EST.MCD

Bølgeforhold: små bølger

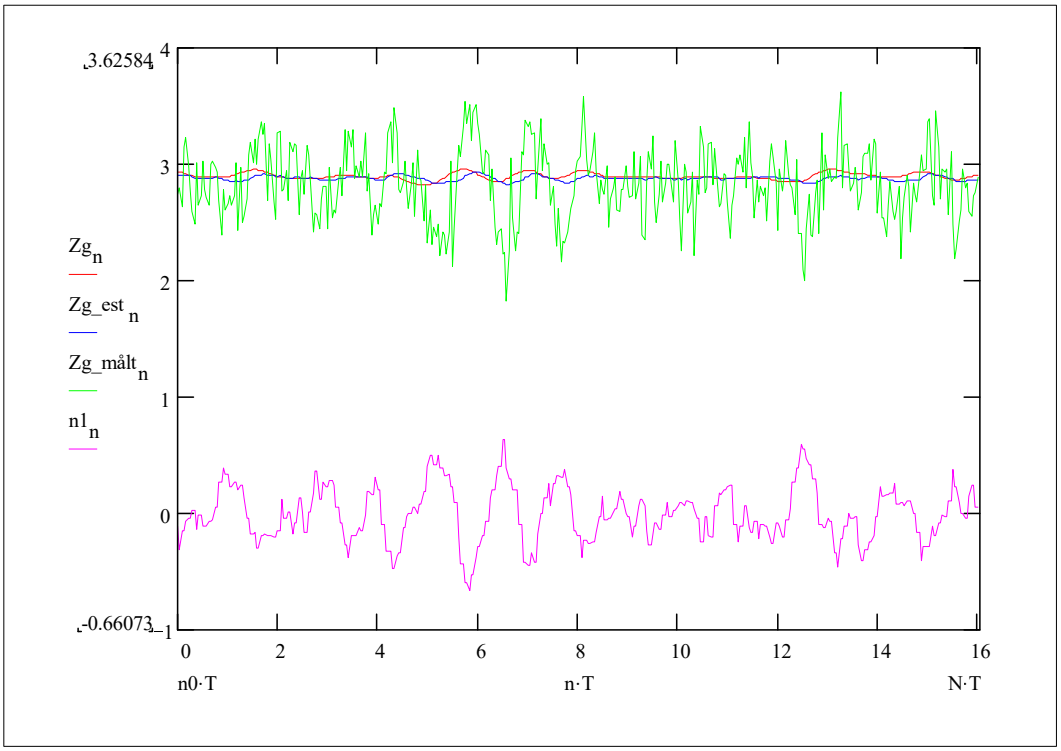
Fartøyets hastighet: $Vx = 40\ ms^{-1}$.
Fartøyets retning: $\theta = 3$ rad.
Signifikant bølgehøyde $h_s = 0.6$ m.
Signifikant bølgelengde $\lambda_s = 60$ m.

Skyvekraft: $T=7*10^4$ N.
Referansehøyde $Zg_{ref} = 2.9$ m.

MathCad- plot av ASCII- domp fra logmodul: Rorutslag



MathCad- plot av ASCII- domp fra logmodul: Høyder

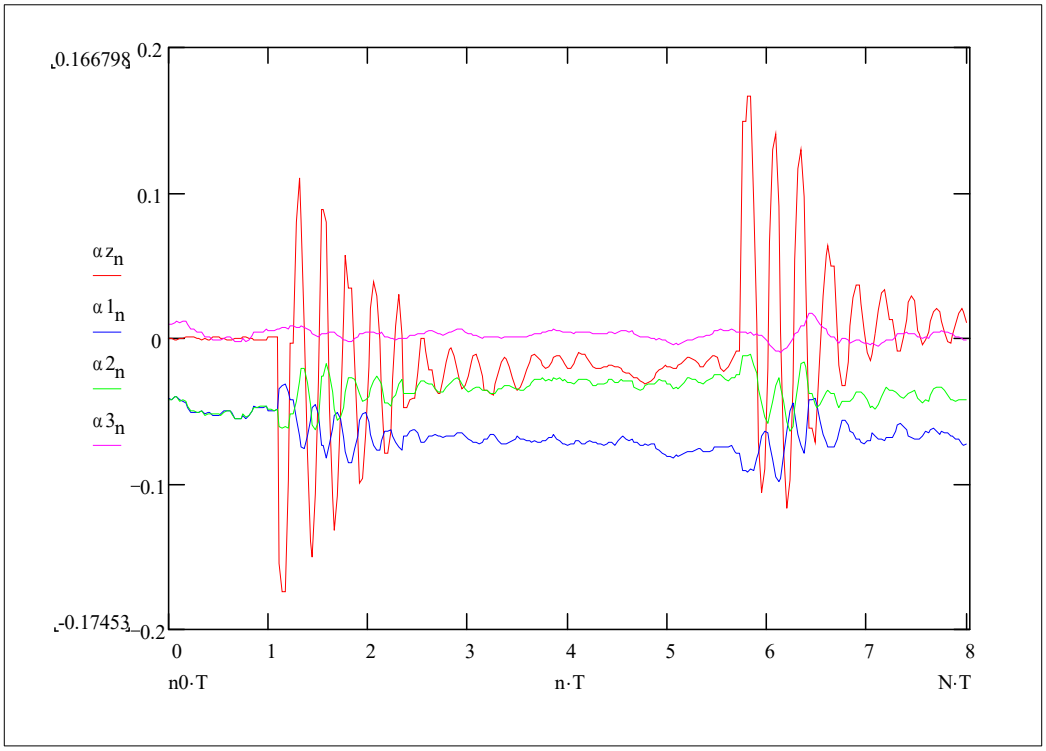


OEV i sving

Fartøyets hastighet: $Vx = 40\ ms^{-1}$.
Fartøyets retning: $\theta = 3$ rad.
Signifikant bølgehøyde $h_s = 0.6$ m.
Signifikant bølgelengde $\lambda_s = 60$ m.

Skyvekraft, $T=7*10^4$ N.
Referansehøyde $Zg_{ref} = 2.9$ m.
Referanse rotasjonshastighet $\omega\theta = 0.1\ rads^{-1}$.

MathCad- plot av ASCII- domp fra logmodul: Rorutslag



Fra nedsenket til luftbåren

Skyvekraft: $T=1.4*10^5$ N.
Starthastighet: $Vx=10\ ms^{-1}$.
Starthøyde: $Zg=0.9$ m.

Observert resultat: Regulator gir liten stabiliserende effekt. Alle høyderor står maksimalt opp. Akkselerasjon stanser ved 28 knop. Høyde uforandret.

7.1.4. Regulatoren SUPEROEV

Regulator data

Regulator: SUPEROEV.REG
Estimator: SUPEROEV.EST
Prosessmodell: SUPEROEV.MOD

Samplingsfrekvens, regulator SUPEROEV: 72 Hz.
Samplingsfrekvens, estimator SUPEROEV: 144 Hz.
Samplingsfrekvens, prosess SUPEROEV: 144 Hz

Optimalvekten Q, Kostnadsvekten P, Tilbakekoblingsmatrisen G:
Se Appendix A: SUPEROEV.MCD.

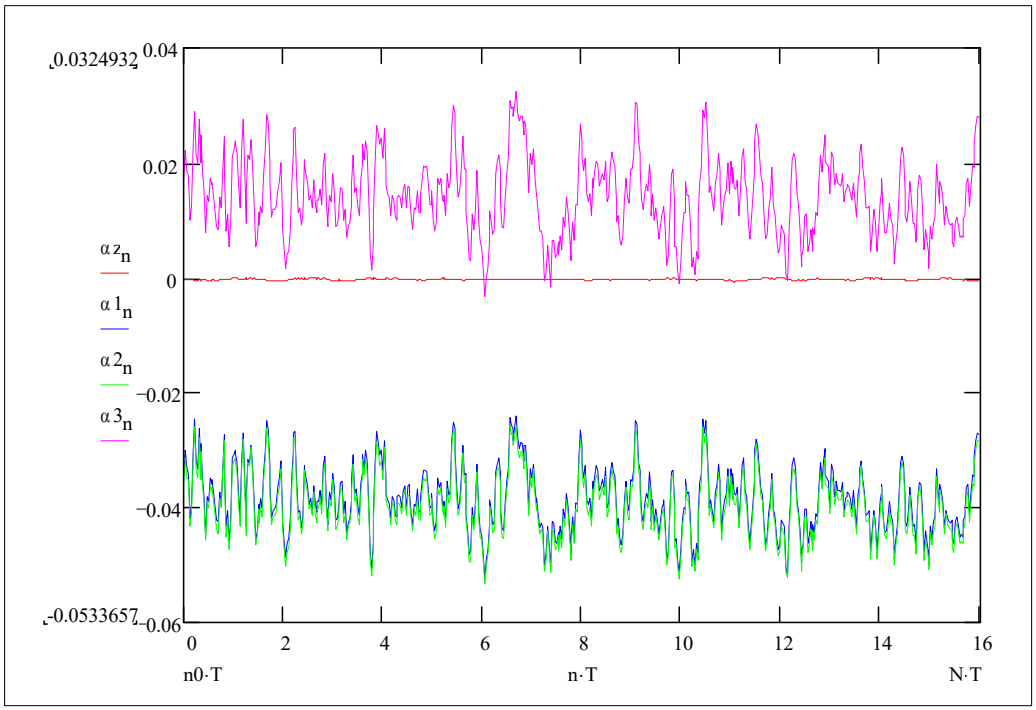
Estimator kovariansmatrise W, Estimator kovariansmatrise V, Estimator stasjonær K- matrise:
Se Appendix A: EST.MCD.
Elementer for målestøy er justert: $W_{10,10}=1$, $W_{11,11}=1$, $W_{12,12}=1$
Elementer for prosesstøy er justert: $V_{3,3}=1$, $V_{4,4}=1$, $V_{5,5}=1$
Optimal K er beregnet i estimatoren med opsjonen Optimal K.

Bølgeforhold: små bølger

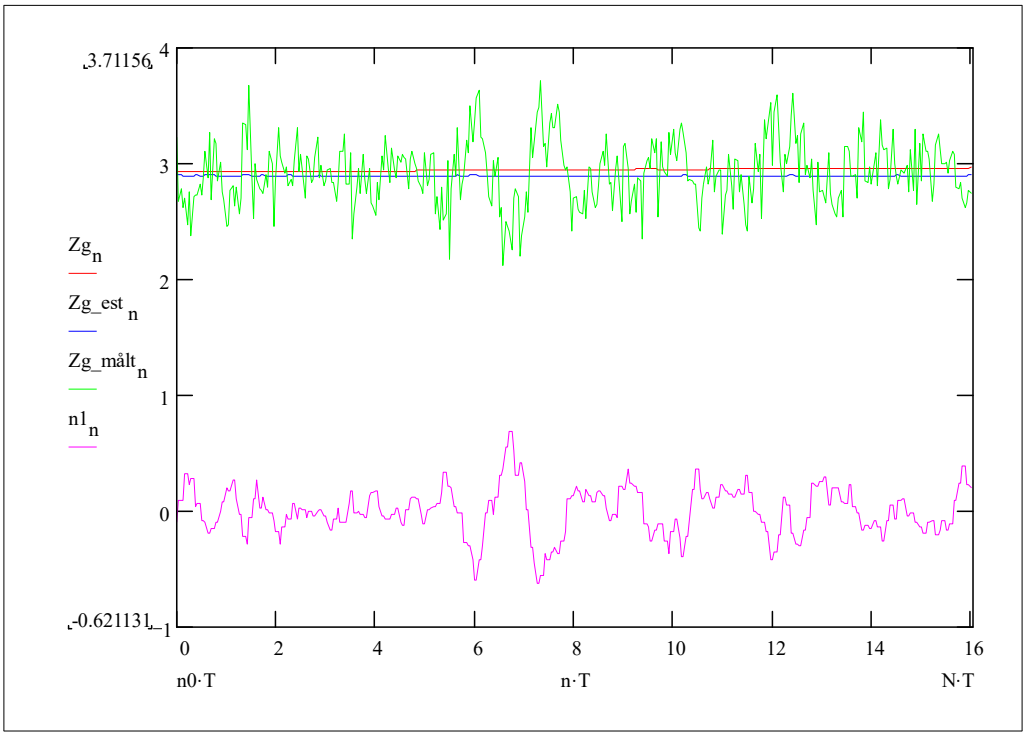
Fartøyets hastighet: $Vx = 40\ ms^{-1}$.
Fartøyets retning: $\theta = 4$ rad.
Signifikant bølgehøyde $h_s = 0.6$ m.
Signifikant bølgelengde $\lambda_s = 60$ m.

Skyvekraft $T=7*10^4$ N.
Referansehøyde $Zg_{ref} = 2.9$ m.

MathCad- plot av ASCII- domp fra logmodul: Rorutslag



MathCad- plot av ASCII- domp fra logmodul: Høyder

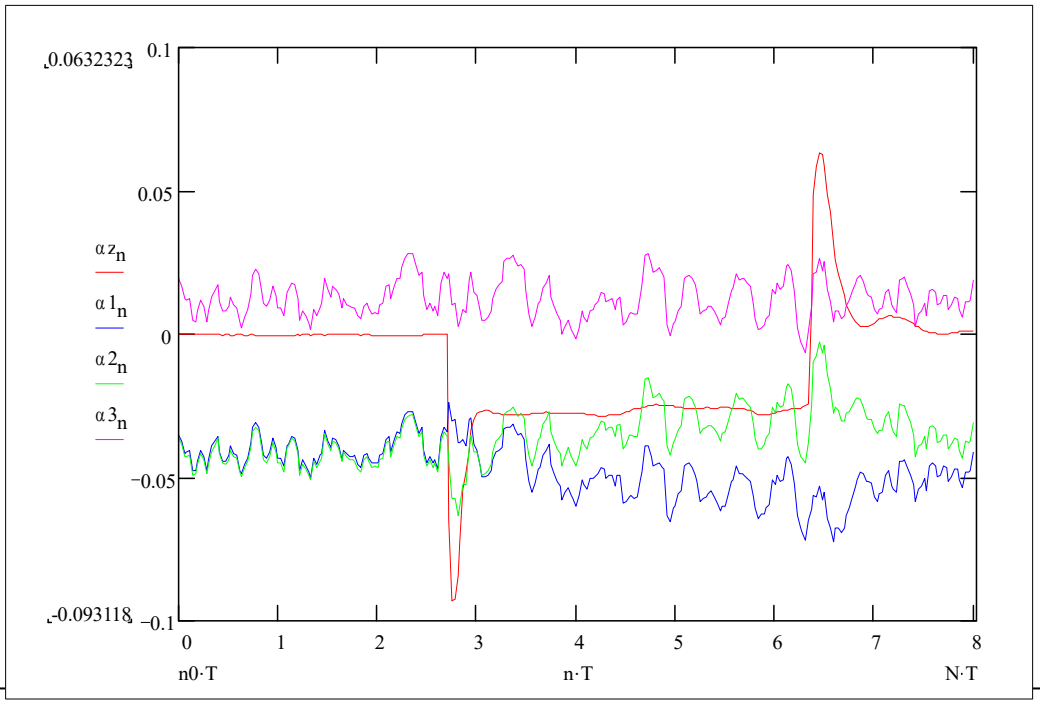


OEV i sving

Fartøyets hastighet:	Vx = 40 ms^{-1}.
Fartøyets retning:	θ = 4 rad.
Signifikant bølgehøyde	h_s = 0.6 m.
Signifikant bølgelengde	λ_s = 80 m.
Skyvekraft	T=7*10^4 N.
Referansehøyde	Zg_{ref} = 2.9 m.

Referanse rotasjonshastighet ωθ= 0.1 $rads^{-1}$.

MathCad- plot av ASCII- domp fra logmodul: Rorutslag



Fra nedsenket til luftbåren

Skyvekraft: $T=1.4*10^5$ N.
Starthastighet: $Vx=10$ ms^{-1}.
Starthøyde: $Zg=0.9$ m.

Observert resultat: Regulator gir en viss kursstabiliserende effekt. Alle høyderor står maksimalt opp. Akselerasjon stanser ved 28 knop. Høyde uforandret.

Bølgeforhold: digre bølger

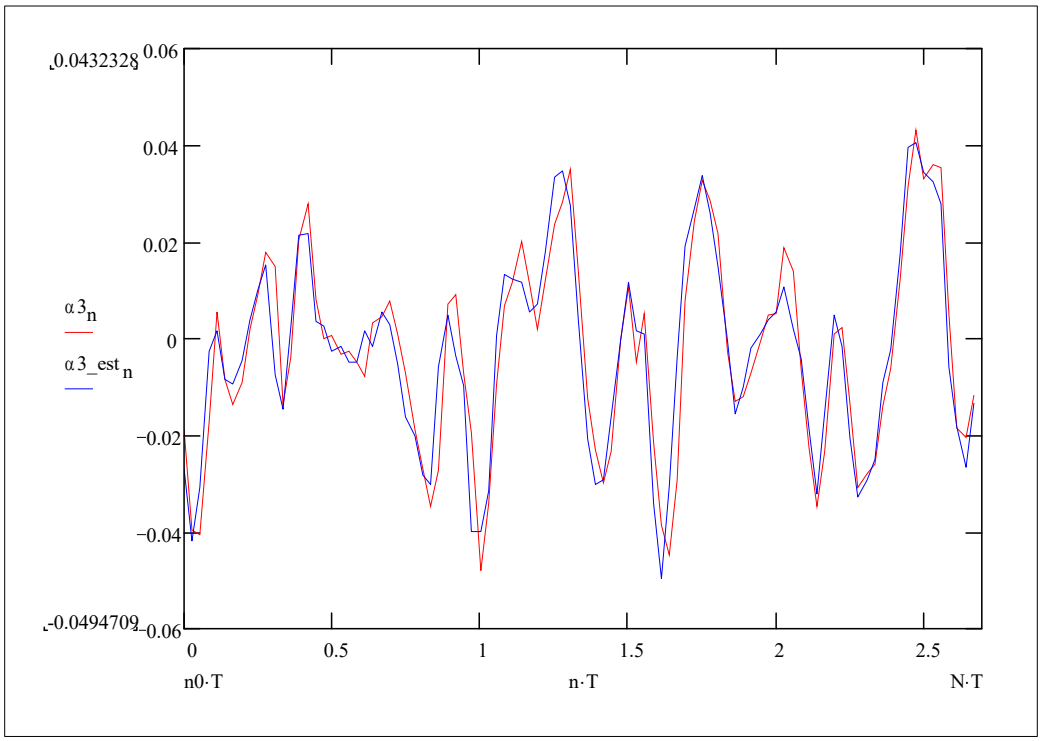
Fartøyets hastighet: $Vx = 40$ ms^{-1}.
Fartøyets retning: $\theta = 4$ rad.
Signifikant bølgehøyde $h_s = 4$ m.
Signifikant bølgelengde $\lambda_s = 200$ m.

Justerte kovariansmatriser for estimatoren:

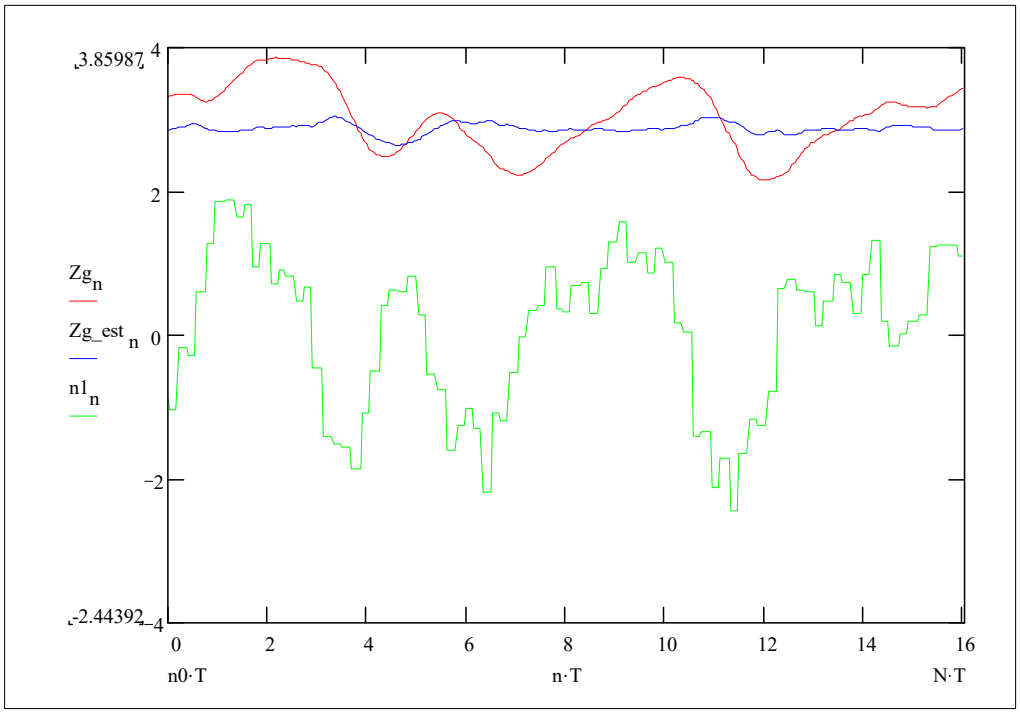
Elementer for målestøy er justert: $W_{10,10}=0.1$, $W_{11,11}=0.1$, $W_{12,12}=0.1$
Elementer for prosesstøy er justert: $V_{3,3}=4$, $V_{4,4}=4$, $V_{5,5}=4$
Ny K beregnes i estimatoren med opsjonen Optimal K.

Skyvekraft $T=7*10^4$ N.
Referansehøyde $Zg_{ref} = 2.9$ m.

MathCad- plot av ASCII- domp fra logmodul, utsnitt: Rorutslag, foil3 med estimat



MathCad- plot av ASCII- domp fra logmodul: Høyder



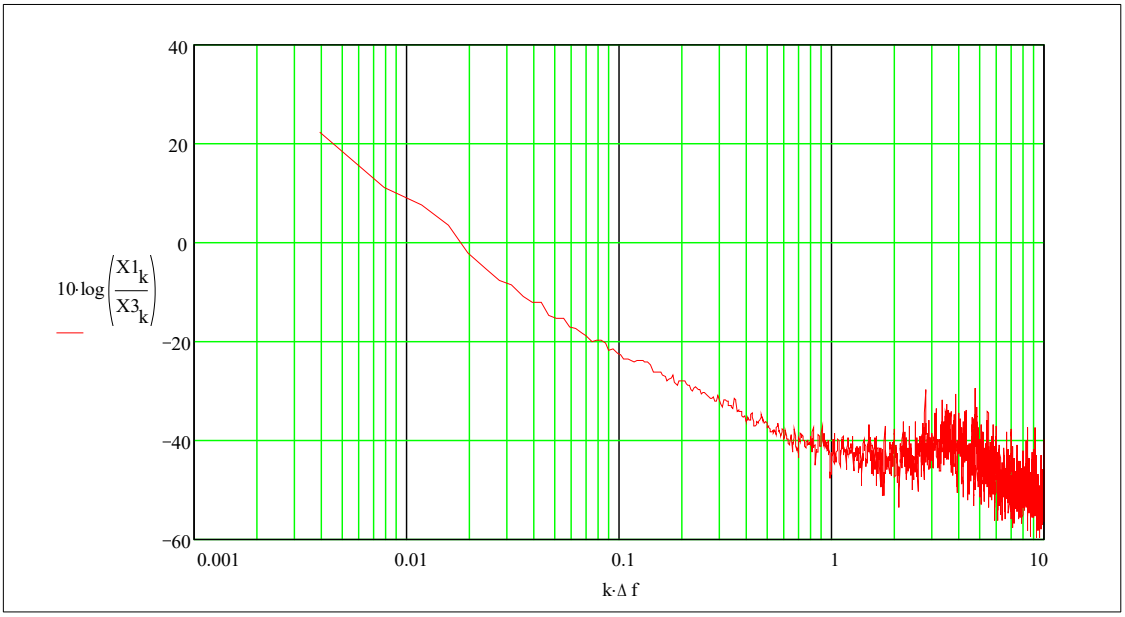
Spektralanalyse

Regulator data som ved simulering: SUPEROEV, små bølger.

Fartøyets ferd over tid fra log- data: SUPERSIM.PRN er analysert i MathCad. Effektspekter for fartøyets høyde og sjøens overflate gir transferfunksjon for fartøyets høyde i forhold til sjøoverflaten [Appendix A: SUPERSIM.MCD].

Sampelets totale tid: 256 s.

Samplingsfrekvens: 24Hz.



Figur 7.1.4.1.: Overføringsfunksjon for fartøyets høyde, funnet ved spektralanlyse.

7.2. DISKUSJON: MODELLERING

Vurdering av modelleringsarbeidet og grad av måloppnåelse for modellen.

7.2.1. Omgivelser

Utprøving av modell for vind i simuleringsprogrammet viser at modellen virker som forutsatt. Det er ikke sikkert at modellen har helt korrekt utseende med hensyn til spektralfordeling, men det er ikke så viktig i denne anvendelsen. Med de muligheter en har for å manipulere med signalet i programmet, kan en simulere realistiske vindforhold for sjøgående fartøyer på en bra måte.

Totalt sett er en fornøyd med bølgemodellen, en får simulert realistiske bølgeforhold for OEV- fartøyet.

Bølgemodellen gir ikke alltid riktig bølgehøyde. Det viser seg at høyden avhenger av fartøyets retning i forhold til bølgefronten og valgt bølgelengde. I simuleringsprogrammet kan en da overvåke bølgehøyden kontinuerlig og justere signifikant høyde slik at en får de bølgeforhold en ønsker seg.

Når fartøyet kjører i bølgeretning med hastighet nær bølgeformens hastighet er det liten variasjon på overflaten. Ettersom bølgespekteret består av flere enkeltbølgekomponenter med forskjellig amplitude, bølgelengde og fart, er det grunn til å anta at modellen her gir feil representasjon.

En ulempe med bølgemodellen er at de tre signalene for overflateoffset er beregnet ut i fra fysiske data på OEV- fartøyet. Hvis fartøyets dimensjoner endres, må en revidere implementasjonen av bølgemodellen og rekompilere programmet.

7.2.2. OEV- fartøy

En ser at arbeidet med modellering av OEV- fartøyet er omfattende. Selv om en har gjort betydelige forenklinger for modell av foiler, og valgt enkle løsninger for fartøyets dynamikk når skroget er i kontakt med sjøen, er omfanget av ligninger og koeffisienter betydelig. Dette øker risikoen for feil. Underveis fant man stadig nye feil i modellen, og det er tidkrevende å oppdatere modellen ved endring av fartøyets fysiske egenskaper. Det som skapte størst problemer var oppdatering av matriseelementer i tilstandsmodellen [OEVMOD.MCD] ved endring av fartøyets dynamiske modell [SKROG.MCD]. En kunne ha benyttet ett dokument for oppsett av både dynamisk modell og tilstandsmodell, men da ville dette blitt stort og uhåndterlig.

Oppdatering av den ulineære modellen i simuleringsprogrammet var litt enklere. Koeffisienter som inngår endres i programfilens begynnelse, endring i struktur lar seg lett implementere i metoden OEVProsess.calcgFmatrix [OEV0.PAS]. Programmet må rekompileres, dette kan være et problem hvis en ikke har tilgang til personer med kunnskap om programmering i TurboPascal.

En god løsning for modellering ville være å utvide simuleringsprogrammet til å håndtere modellering i tillegg til simulering. Programmet kunne da oppdatere ulineær modell selv, og ha mulighet for lagring av matriser til fil for import i andre programmer.

En er fornøyd med modellens oppførsel når normalbetingelsene er oppfyllt. Representasjonen av fartøyets dynamikk når skroget er i kontakt med sjøen er ikke like god, men den holder til å få med effekt fra bølger som slår opp i skroget når fartøyet er luftbåren. Hvis fartøyet står stille og påvirkes av bølger ser en at simuleringen blir ustabil. Dette kan skyldes at matriseelementene i modellen endrer seg for raskt i forhold til samplingstiden.

Det kunne vært ønskelig med en bedre modell for fartøyets oppførsel ved lave hastigheter, når mesteparten av løftet er statisk og dynamisk løft fra skroget. En føler seg ikke sikker på modellens gyldighet ved simulering fra nedsenket drift i lav fart opp til luftbåren drift.

7.3. DISKUSJON: SIMULERINGSPROGRAM

Vurdering av arbeidet med utvikling av simuleringsprogram og programmets anvendbarhet.

Ved å simulere et lineært 5. ordens system med kjent respons [Fil: FJEDRING.ZIP] fikk en kontollert programmets simuleringsalgoritmer. En ser at programmet simulerer dette systemet riktig, og føler seg trygg på at programmet regner rett.

Programmet er brukervennlig å arbeide med. Funksjoner kan opereres fra både tastatur og peker. En kan lett observere tilstander, målinger og pådrag med Logmodulene, matriser kan importeres fra andre programmer via ASCII- format filer. Det grafiske grensesnittet gir en behagelig atmosfære å arbeide i.

En ulempe med implementasjon under MS- DOS med Turbo- Pascal er at programmet ikke kan portes til andre plattformer. I fremtiden vil en nok se at arbeidsstasjoner med UNIX/ X- Windows / Motif får større innpass i slike anvendelser, en slik maskin har regnekraft i en helt annen klasse enn dagens 486- PC'er. OEV- programmet vil nok kunne kjøres på en slik maskin, fordi en har brukt dokumenterte interrupt- og operativsystemkall, men for å få utnyttet kapasiteten best mulig burde en programmere i C++ under det grensesnitt maskinen kjører på. En ser også at MS- DOS / intel- miljøet er strekt til aller ytterste grense. Ved kjøring med alle moduler åpne bruker en opp alt DOS- memory (den magiske 640K- grensen...), og simuleringer går utriveligt tregt. Ulineær OEV- simulering går ikke i sann tid, maskinen greier rett og slett ikke å regne raskt nok. Typisk så går simulert tid 6- 10 ganger saktere enn reell tid ved simulering av regulert OEV- fartøy.

Ustabile simuleringer vil føre til overflow i flyttallsprosessoren. Programmet har ingen egen interrupt- handler for flyttallshasarder, slik at dette umiddelbart fører til ukontrollert terminering av programmet. Hvis en kjører under OS/2 går dette bra, det eneste som skjer er at OEV- programmet lukkes ned. Ved kjøring under MS- DOS og Windows 3.1 går det også som regel bra. Hvis programmet går ned ved kjøring fra MS- DOS direkte, går det aldri bra. Da er det bannskap og reset neste.

Simuleringer over lengre tid krever mye XMS- memory, dette er et problem ved kjøring under MS- Windows. Under arbeidet med utvikling av regulatorer kjørte en ofte med både simuleringsprogram, MatLab, MathCad og enkelte små applikasjoner åpne samtidig. Da fikk en erfart hvor elendig Windows håndterer memory- allokering og de såkalte 'systemressursene'. Det hjelper ikke at maskinen vår har 20Mb RAM. Applikasjoner går ned, og til slutt havarerer hele stasen. Da er det en befrielse å arbeide med OEV- programmet under OS/2. En kan tildele programmet så mye XMS-minne en ønsker, og en trenger aldri være redd for systemhavari hvis programmet går ned på grunn av flyttallsfeil.

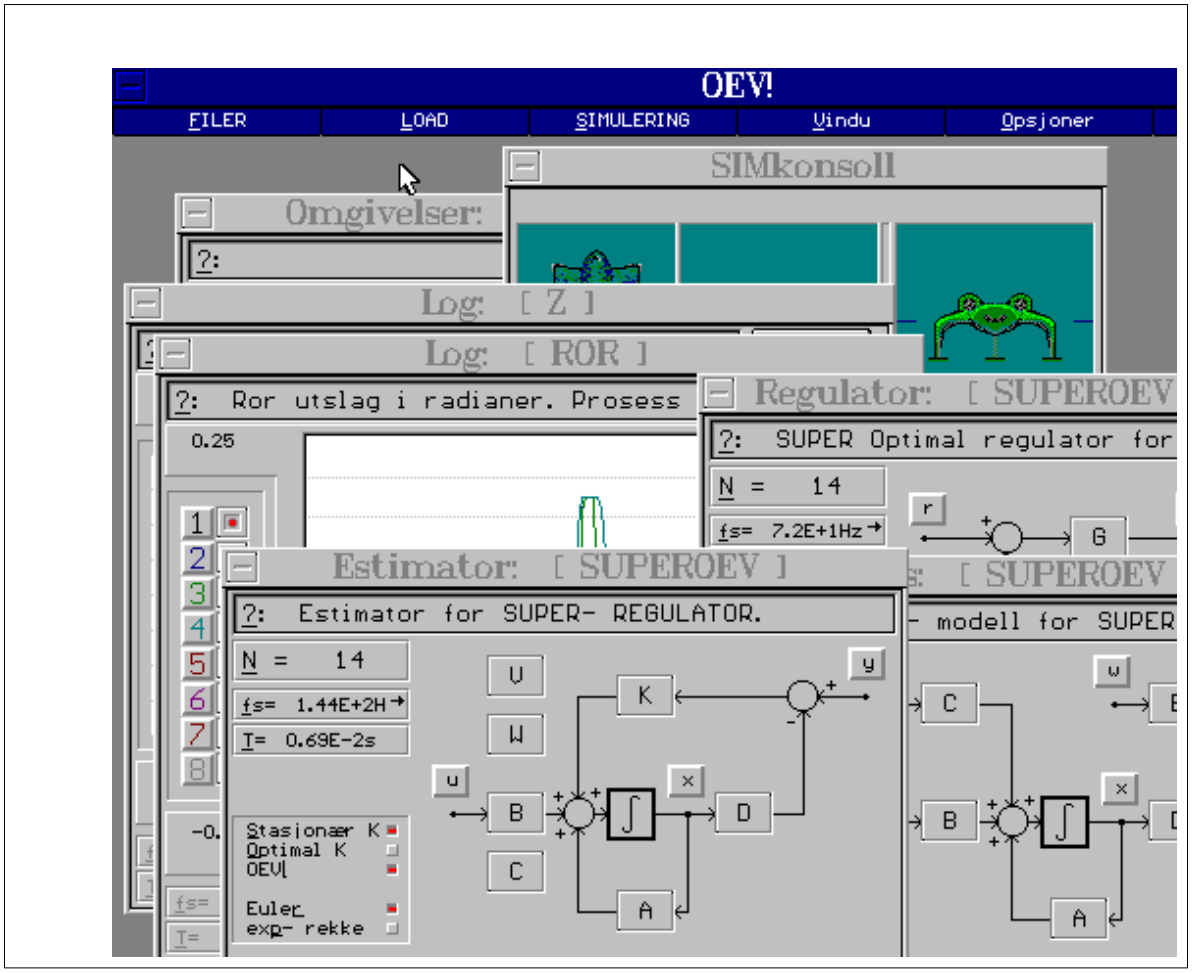
En ser stor nytte av å kunne implementere en ulineær modell for simulering. Bare det at en får med begrensninger for pådragsorganenes utstyringsområde, medfører at en får en mye mer realistisk simulering enn det er mulig med en linearisert approksimasjon. Dette minsker gapet mellom teoretisk modell og praktiske utprøvinger.

Utvikling av programmet har vært en særdeles tidkrevende prosess. En vurderer det slik at dette er vel anvendte ressurser, programmet er nødvendig som verktøy for utvikling av gode regulatorer for OEV- fartøyet. Alternativene ville være å bruke ferdige programpakker for simulering. En slik simuleringsmodell kan ikke implementeres i MathCad. En antar det er mulig å implementere systemet i MatLab, dette ble ikke prøvd. I alle tilfelle vil det bli en nokså komplisert oppgave hvis en skal ha samme muligheter for grafisk visning av tilstander.

Det ble brukt en del tid i prosjektets startfase til å fikse systemkjernen, som håndterer grensesnittet, slik at denne fungerer feilfritt. Kjernen er et komplekst rot av objekter, funksjoner og globale variable, som en ikke har full oversikt over. Lenger opp i systemets hierarki, når en kommer opp på plan som håndterer knapper, vinduer, menyer er det mer oversiktlig. Her brukes objektorientert tankegang fullt ut.

Hele OEV- programmet består av 18621 linjer kildekode. Et slikt system hadde nok vært særdeles uoversiktlig med en klassisk programmeringsstruktur, i hvertfall ville en ikke fått det til å fungere med de ressurser og tid som var til rådighet for dette prosjektet.

Figur 7.3.: Typisk skjermbilde i OEV- programmet



7.4. DISKUSJON: REGULERING

Vurdering av resultater oppnådd med regulering av OEV- fartøy.

7.4.1. Innledende eksperimenter

Før arbeidet med utvikling av regulator startet, hadde en prøvd å simulere OEV-fartøyet ved å styre det manuelt (sette inn tallverdier i pådragsvektoren **u** i programmet). Dette er vanskelig, og får som oftest et lite heldig utfall. En ser at fartøyet er ustabilt og vanskelig å styre.

Den første regulatoren som ble utprøvd var en enkel G- matrise innkoblet i linearisert modell, der tilbakekoblingen hentes fra prosessmodellens **x**- vektor. Simuleringsresultatet fra denne er ikke særlig interessant, det eneste reelle en kan få ut av en slik simulering er om fartøyet er stabilt i arbeidspunktet.

De første skikkelige simuleringene med ulineær modell ble foretatt med en fast G-matrise tilbakekobling. Dette virket bra så lenge en holdt designfart, 80 knop. Når hastigheten senkes ned mot 60- 70 knop blir reguleringen farlig treg og får dårlig følging. Økes hastigheten over 85- 90 knop, blir fartøyet ustabilt og havarerer. På dette stadiet ble det observert tendens til rotasjonsustabilitet om fartøyets x- akse. Det viste seg at dette skyldtes en feil i implementeringen av ulineær modell i programmet. Denne feilen ble rettet, og ustabiliteten forsvant.

En regulator med adaptiv G ble utviklet og utprøvd. Denne ga gode resultater over et stort hastighetsområde, typisk 55- 100 knop. Også denne simuleringen er urealistisk fordi en koblet tilbakekoblingen rett fra prosessmodellens **x**- vektor, som ikke er målbar i et virkelig OEV- fartøy.

7.4.2. Regulatoren OPTOEV

Da denne regulatoren ble laget var det uklart hvordan estimatoren skulle bli. Derfor ble G- matrisens elementer som har kobling til foilvinkler satt til null fordi en ikke var sikker på å få tilgang til disse. Dette går bra så lenge regulatoren ikke er for rask i forhold til tregheten i pådragsmekanismene.

En ser at denne regulatoren stabiliserer fartøyet og holder bevegelser på et tolerabelt nivå. Høyden holdes fast nær referanseverdien, avvik grunnet estimeringsfeil og det faktum at regulatoren mangler integratorer er ubetydelig. Innføring av integratorer vurderes som lite heldig, da forverres nemlig systemets dynamiske egenskaper, dessuten er det unødvendig å holde høyden mer eksakt enn det en oppnår uten.

Brukbart hastighetsområde for denne regulatoren er 55- 100 knop. Hastigheter over 100 knop medfører skyvekraft over maksimalverdi og er derfor umulig, under 55 knop brukes løftet fra foiler opp til å holde fartøyet luftbåren, slik at en ikke har reguleringsområde igjen til å holde fartøyet rotasjonsstabilt om x- og y- akser.

Ved brå svingmanøvrer ser en at sideroret gjør noen utrivelige transiente utslag. Faktisk er en her på grensen til ustabilitet. Ellers holdes fartøyet fremdeles plant når en får påvirkning fra sidekrefter. Det er viktig å holde sidekrefter på et moderat nivå i sving, ellers risikerer en at reguleringsområdet for foilene brukes opp til å holde fartøyets rotasjon om x- aksen.

Regulatoren greier ikke å ta seg inn i igjen hvis fartøyet kolliderer med bølger eller foiler letter fra sjøen. Dette medfører at det er direkte hasardiøst å kjøre OEV under forhold der det er risiko for å støte på bølger høyere enn 3 m.

En gjorde forsøk på å starte fra stillstand og gå opp til luftbåren drift. Dette viser seg å være umulig med regulatoren innkoblet. Skyvekraftens angrepspunkt er over vannlinjen, slik at nesen tvinges ned. Dette fører til så stor vannmotstand at en aldri greier å få opp stor nok fart til å ta av. En alternativ metode er å styre fartøyet manuelt i startfasen, og deretter koble inn regulator når skroget løftes opp. Dette gjøres ved å stille fremre foiler på maksimalt løft, og aktre foil maksimalt nedover. Da oppnås etterhvert stor nok fart til å ta av. Ved 35- 40 knop løftes skroget opp, men farten er enda for liten til at regulatoren kan kobles inn. Foreløpig konklusjon blir at fartøyet har for

lite løfteareal i foilene. En bedre modell for fartøyets oppførsel med skroget i sjøen vil gi sikrere informasjon om hvor vidt fartøyet i det hele tatt kan klare å ta av.

Denne regulatoren er et eksempel på hvordan den endelige løsningen kunne ha blitt uten ulineær simulering av systemet. Her ser en at et bedre resultat kanskje kan oppnås ved å utnytte utstyringsområdet for foilene bedre.

7.4.3. Regulatoren SUPEROEV

Revidering av optimalvekter og kostnadsvekter gir en ny G- matrise. En bedre estimator som gir estimat for foilvinkler medfører at denne regulatoren får raskere respons, bedre stabilitet og bedre følging enn den beskrevet under punkt 7.4.2.

Brukbart hastighetsområde og oppførsel ved overgang nedsenket- luftbåren er som under punkt 7.4.2.

Ved brå svingmanøvrer ser en at siderorutslag nå er kontrollert og fint.

En ser at fartøyets bevegelser er betydelig redusert, så lenge bølgene ikke slår opp i skroget vil fartøyets bevegelser være så godt som umerkbar for kaffedrikkende passasjerer.

Regulatoren greier som regel å ta seg inn i igjen hvis fartøyet kolliderer med bølger eller foiler letter fra sjøen. Bevegelsene kan bli nokså voldsomme, men fartøyet er fremdeles stabilt.

Ved å endre verdier i estimatorens kovariansmatriser, W og V, oppnår en at fartøyet følger sjøen når forholdene er slik at plattform- kjøring ikke er mulig. Filtrering i estimatoren blir dårligere, slik at dempning av fartøyets bevegelser blir dårligere. En ser at fartøyet nå kan greie å følge sjøen, men estimatoren innfører fasedreining slik at fartøyets høyde er forsinket i fase i forhold til havoverflaten. Når farten er stor fører dette til at fartøyet dundrer inn i bølgene og totalhaverer. En ser også at rorutslag for foiler fort overskrides, dette fører til ustabilitet og havari ved at fartøyet tipper over rundt x- eller y- akse. Det er nødvendig å endre både verdier for bølgers kovarians i V-matrisen og høydemålingens kovarians i W- matrisen. W- matrisen må endres fordi en i utgangspunktet har valgt en for høy verdi for målestøyen, slik at en får for liten vekt på høydemålingene.

Spektralanlyse av fartøyets høyde viser at påvirkning fra bølger dempes kraftig. Fartøyets bevegelse har en resonansfrekvens rundt 3.5Hz. Denne byr ikke på problemer da signalet er dempet med mer enn 40 dB i området rundt resonansen, og Q- faktor ser ut til å være under kontroll. Spektralanalysen er usikker for lave frekvenser, under 0.02Hz, grunnet kort simuleringstid og for liten variasjon i bølgeforhold. At fartøyets bevegelser under 0.01- 0.02 Hz ser ut til å være større enn overflaten antas å skyldes usikkerhet ved analyse av så korte samples (Δf=0.004 Hz er minste oppløsning her). En annen mulighet er at estimatoren har en lavfrekvente resonansfrekvens med dårlig dempning. Simulering over lang tid viser at fenomenet skyldes estimatorens innsvingningforløp og et statisk estimeringsavvik som avhenger av fortidige hendelser. Estimeringsfeil skyldes at estimatorens modell ikke er en eksakt representasjon av fatrøyet. Typisk feil er mindre enn 10 cm., men når pådragsorganene overstyres kan den bli langt større, og avviket rettes ikke opp igjen over tid. Problemet kan løses ved å ha et overliggende system som overvåker høydemålinger og oppdaterer kovariansmatrisene V og W med lavt samplingsintervall. Da kan en samtidig få automatisk følgemodus ved store bølgehøyder. Hvis en velger kovarians for målestøy = 0.2 gir estimatoren rett høydeestimat, men støyen er i største laget.

Sett under ett er en fornøyd med de resultater er har oppnådd med regulering av fartøyet. Det er den fysiske konstruksjonen som begrenser ytelsen.

8. KONKLUSJON

Oppsummering av konklusjoner som kan trekkes.

Modellering

Modellering av OEV- fartøy er arbeidskrevende. Omfanget av ligninger er stort, det er nødvendig å være nøyaktig og systematisk for å unngå feil. Når modellen er brakt over på tilstandsform er det imidlertid klare fordeler med metoden. Regning foretas på et høyere abstraksjonsnivå, med enkle matriseligninger som representerer systemet.

Simuleringsprogram

En ser at programmet simulerer kjente systemer riktig, og en føler seg derfor trygg på at programmet regner rett.

Programmet strekker grensene for MS- DOS / intel- miljøet til det aller ytterste. For simulering av enda større systemer må en vurdere å bruke kraftigere maskiner og andre operativsystemer.

En ser stor nytte av å kunne implementere en ulineær modell for simulering. Det medfører at en får mer realistisk simulering enn med en linearisert approksimasjon, slik at gapet mellom teoretisk simulering og praktiske utprøvinger minker.

Utvikling av programmet har vært en særdeles tidkrevende prosess. En vurderer det slik at dette er vel anvendte ressurser, programmet er nødvendig som verktøy for utvikling av gode regulatorer for OEV- fartøyet.

Regulering

Regulatoren SUPEROEV stabiliserer fartøyet og gjør det mulig å styre manuelt.

Fartøyet har for lite foilareal til at kontrollert kjøring kan foretas under 55 knop. Dette medfører problemer ved start fra stillstand. En bedre modell for fartøyets oppførsel med skroget i sjøen vil gi sikrere informasjon om hvor vidt fartøyet i det hele tatt kan klare å ta av.

Ved brå svingmanøvrer ser en at fartøyet holdes stabilt og siderorutslag er kontrollert og fint.

En ser at fartøyets bevegelser i høyderetning er moderate, så lenge bølgene ikke slår opp i skroget vil fartøyets bevegelser være så godt som umerkbar for kaffedrikkende passasjerer.

Regulatoren greier som regel å ta seg inn i igjen hvis fartøyet kolliderer med bølger eller foiler letter fra sjøen. Bevegelsene kan bli nokså voldsomme, men fartøyet er fremdeles stabilt.

Totalt sett er en fornøyd med de resultater er har oppnådd med regulering av fartøyet. Det er den fysiske konstruksjonen som begrenser ytelsen.

Samlet konklusjon

En ser at formålet med oppgaven er løst. Simuleringsprogrammet, modell av OEV-fartøyet med omgivelser og regulatoren SUPERREG kan simulering regulert OEV-fartøy med påvirkning fra bølger og vind.

9. SAMMENDRAG

Kortfattet sammendrag av de viktigste momentene.

Problemet

En OEV er et hurtiggående sjøgående fartøy som utnytter både aerodynamisk løft fra skrog og hydrodynamisk løft fra tre nedsenkede hydrofoiler for å løfte skroget klar av vannflaten. Oppgaven er å konstruere en regulator som kontrollerer fartøyet ved å justere angrepsvinkelen for foilene. For å få til dette må en gå veien om modellering av OEV- fartøy med omgivelser og simulering for utprøving av mulige løsninger.

Modell av vind og bølger

Det ble laget stokastiske modeller for vind og bølger. Vind implementeres som integrert hvit støy, en såkalt Brownsk prosess. Bølger lages ved å ta utgangspunkt i Bretschneiders bølgespekter. Ved å filtrere hvit støy i et filter som tilnærmer denne spektralfordelingen, fåes en modell som gir et signal med frekvensfordeling lik det virkelige bølger har.

En modell av OEV- fartøyet

Det ble satt opp en matematisk modell av OEV- fartøyet. Med utgangspunkt i kraftbalanse får en et sett med første ordens differensialligninger. Tilstander defineres, slik at en kan sette opp en 14.- ordens tilstandsmodell. Da er systemet på standard tilstandsrom form, slik at generell teori for regulering kan anvendes.

Det ble laget to tilstandsmodeller. En linearisert tilnærmelse beskriver dynamikken i arbeidspunktet, denne representasjonen benyttes for beregning av regulatoren. En ulineær modell beregnet på implementasjon i simuleringsprogram gir en bedre beskrivelse av fartøyet over et vidt hastighetsområde.

En gikk gjennom mulige løsninger for instrumentering. Dette gir et sett av målinger til regulatoren.

Simuleringsprogram

For å kunne simulere fartøyet mest mulig realistisk ble det laget et program som kan simulere fartøyet. Ulineær modell og stokastiske modeller for vind og bølger gir en god representasjon av OEV- fartøyet. Regulator, estimator, prosessmodell og omgivelser er implementert som helt selvstendige enheter som kan simuleres hver for seg. Et brukervennlig grafisk grensesnitt gjør programmet lett og behagelig å arbeide med.

Grafisk visning av tilstander i kurveplot og bilde av OEV- fartøyet gir rask tilgang til informasjon om tilstander i systemet.

En objektorientert arkitektur medfører bedre oversikt enn klassiske programmeringsmåter. Fleksibilitet oppnås ved å bygge videre på allerede eksisterende moduler, programmet kan lett utvides og det er lett å foreta endringer i koden.

Regulator

Teori for optimalregulering av stokastiske systemer og Kalman- filter er grunnlaget for utvikling av en regulator.

En kom fram til en løsning med en optimal tilbakekoblingsmatrise, G, der målesignalet til denne hentes fra en Kalman- estimator. Estimatoren finner informasjon om tilstander en ikke kan måle direkte, og filterer vekk støy.

Oppnådde resultater

Med simuleringsprogrammet som verktøy kom en fram til en regulator som kontrollerer fartøyer så bra som det er mulig å få til med de fysiske begrensninger som er innebygd i konseptet. En ser at så lenge bølgehøyde ikke overskrider 3 m., kan fartøyet kjøre i 80 knop uten at bevegelser vil merkes noe særlig for passasjerene. Med bølgehøyde på 1 m. holdes høydevariasjon innenfor 1.5 cm.