Universitetet i Stavanger

DET TEKNISK-NATURVITENSKAPELIGE FAKULTET

BACHELOROPPGAVE

Studieprogram/spesialisering: Automatisering og elektronikkdesign, bachelor i ingeniørfag	Vårsemesteret, 2021 Åpen / Konfidensiell
Forfattere: Markus Lægreid Haldorsen Edmond Baloku Vigre	Markus L. Haldorsen (signatur Markus Lægreid Haldorsen) Edmond Baloku (signatur Edmond Baloku Vigre)
Fagansvarlig: Morten Tengesdal Veileder(e): Morten Tengesdal	
Tittel på bacheloroppgaven: Motor, styrings- og reguleringssystem for fjernstyrt undervannsfartøy og manipulatorarm Engelsk tittel: Motor, control technology and systems on remotely operated vehicle and manipulator arm	
Studiepoeng: 2x20	
Emneord: Motor, motorkontroller, styringssystem, reguleringssystem, manipulator, manipulatorarm, ROV, MATE, UiS Subsea	Sidetall: 161 + vedlegg/annet: Programkode, Simuleringsfiler, Fremdriftsplan Stavanger, 14.05.2021

UiS Subsea - Motorstyrings- og reguleringssystem

Motor, styrings- og reguleringssystem for fjernstyrt undervannsfartøy og manipulatorarm

Forfattere:

Navn	Studentnummer
Markus Lægreid Haldorsen	249248
Edmond Baloku Vigre	249250

Universitetet i Stavanger

Bacheloroppgave

Institutt for data- og elektroteknlogi (IDE)

14.05.2021

Forord

Det settes stor pris på å ha fått være med på å utvikle et undervannsfartøy for UiS Subsea. Det har på mange måter vært et ganske stressende semester med lange dager og frustrerende arbeid, men i ettertid er det kun positive ting å se tilbake på. Dette gjelder både for læringsutbyttet vi har fått som vil hjelpe mye på når vi skal ut i arbeidslivet, og ikke minst vennskapene som har blitt dannet. Vi ønsker derfor å takke UiS Subsea for et minneverdig semester, i et godt arbeidsmiljø.

Koronapandemien har selvsagt spilt en stor rolle for gjennomførsel av labarbeid, bruk av verksted og sosiale sammenkomster med gruppen. Det gis derfor en ekstra stor takk til Universitetet i Stavanger, som har sørget for tildeling av egne arbeidsplasser på lab, og gode smitteverntiltak generelt på hele universitetet, som har ført til at vi har kunnet gjennomføre oppgaven så og si som normalt. Det takkes også for at universitetet gir mulighet for å holde studentorganisasjoner som UiS Subsea gående, både ved økonomisk støtte og tildeling av eget grupperom.

Takk også til alle sponsorer som har bidratt med midler som har gjort det mulig å bygge et så avansert fartøy.

Vi ønsker sist, men ikke minst, å gi en stor takk til vår veileder, Morten Tengesdal, som har vært veldig behjelpelig gjennom hele oppgavens forløp.

Sammendrag

UiS Subsea er en studentorganisasjon ved Universitetet i Stavanger, som har et årlig mål om å utvikle et undervannsfartøy for å delta i den internasjonale Marine Advanced Technology Education (MATE) ROV[1]-konkurransen [1]. Det stilles i den sammenheng en mengde krav til vekt, utforming, sikkerhet og lignende, som må tas hensyn til ved planlegging, funksjonskartlegging og utvikling.

Denne rapporten omfatter utvikling av et robust og driftssikkert styrings- og reguleringssystem på et sett med valgte motorer og motorkontrollere, for å best mulig kunne gjennomføre oppgavene i MATE ROV konkurransen. I korte trekk gjelder det:

- Valg av motorer og motorkontrollere for fremdrift av ROV'en og for styring av en manipulatorarm, samt beregning av nødvendig tverrsnitt og valg av kontakter.

- Utvikling av et intuitivt og brukervennlig styringssystem, som gir en pilot mulighet for å styre ROV-ens bevegelse i alle ønskelige frihetsgrader[2] ved hjelp av en håndholdt kontroller.

- Utvikling av et intuitivt og brukervennlig styringssystem for manipulatorarm[3], som gir piloten mulighet for å styre samtlige av manipulatorarmens ledd individuelt, ved hjelp av en håndholdt kontroller.

- Drøfting om hvilke frihetsgrader som skal automatisk reguleres, samt utvikling av et raskt, presist og robust reguleringssystem for disse. Matematisk modellering blir benyttet for å analysere fartøyets oppførsel i vann, først og fremst i de frihetsgradene som skal reguleres.

Arbeidet gir et godt grunnlag for et solid og driftssikkert styrings- og reguleringssystem. Manipulatorarmen er testet og fungerer etter sin hensikt. Videre er kode for styring og regulering, bestående av nødvendige beregninger og sending og mottak av styresignaler og sensordata, testet hver for seg. Styrings- og reguleringssystemet er dessverre ikke testet sammensatt på en fysisk ROV, da den ikke er klar for komplett montering ved innlevering av bacheloroppgaven.

Videre arbeid består av å fullføre nødvendig testing av systemet på ferdigmontert ROV i basseng, og justering av parametrer for å oppnå best mulig resultat for styrings- og reguleringssystem. Dette er planlagt fullført innen to uker etter leveringsdato.

[1] Fjernstyrt undervannsfartøy (Remotely Operated Vehicle)
[2] Frihetsgrader refererer til bevegelsesfriheten til et legeme i et tredimensjonalt rom
[3] En fjernstyrt arm som monteres på ROV-en

Ordliste

API Application Programming Interface. Et grensesnitt i en programvare som lar spesifikke deler av denne bli aktivert fra en annen programvare.

ARR AutoReload register. Tallet en timer skal telle opp til før den enten ruller rundt til 0, eller snur telleretning.

BEMF Back Electromotive Force.

BFF Body Fixed Frame.

BLDC Brushless Direct Current motor (Børsteløs likestrømsmotor).

CAD Computer Aided Design (Dataassistert konstruksjon). Konstruksjon og teknisk tegning som utføres ved hjelp av datamaskinbaserte programvarer og redskaper.

CCR Capture/compare register. I capture modus kan CCR registeret lagre timeren sin teller i respons på ett eksternt signal. I compare modus brukes CCR registeret til å sette duty-cycle på ett utgangssignal.

Datatips Punkter for avlesning av data i matlab- og simulinkfigurer.

DC Likestrøm.

Elektronikkhus Vanntett kapsling på fartøyet som inneholder alt av ikkevanntett elektronikk.

FOC Field Oriented Control.

Manipulator Fjernstyrt arm montert på fartøyet som brukes for å plukke opp, holde, flytte og plassere gjenstander.

MATE Marine Advanced Technology Education. Organisasjonen som arrangerer MATE-konkurransen.

NED North East Down.

PFM Pulsfrekvensmodulasjon.

PID-regulator Proporsjonal Integrasjon Derivasjon. En algoritme som brukes for å regulere ROV i ønskede frihetsgrader.

PWM Pulsbreddemodulasjon.

ROV Remotely Operated Vehicle (fjernstyrt undervannsfartøy).

RPM Rotasjoner per minutt.

SPI Serielt perifert grensesnitt. En synkron seriekommunikasjonsgrensesnittspesifikasjon som brukes for kortdistansekommunikasjon, først og fremst i integrerte systemer.

Thruster Motor, propeller og motorhus for fremdrift.

Timer En logisk krets som teller oppover for hver klokkesyklus.

Toppside Et overflatesystem med brukergrensesnitt for pilot, kontrollere for styring, visuell presentasjon av forskjellige data og lignende.

Innhold

1 Innledning **1**
- 1.1 Bacheloroppgaver . . . 1
- 1.2 ROV . . . 3
  - 1.2.1 Frittsvømmende ROV med navlestreng . . . 3
  - 1.2.2 Krypende ROV . . . 4
  - 1.2.3 Strukturelt avhengig ROV . . . 4
  - 1.2.4 Hymir . . . 5
- 1.3 UiS Subsea . . . 5
- 1.4 MATE - Marine Advanced Technology Education . . . 7
  - 1.4.1 MATE ROV Competition . . . 7
- 1.5 Overordnet system . . . 17
  - 1.5.1 Blokkskjema . . . 17
  - 1.5.2 Bildegjenkjenning og autonom kjøring . . . 18
  - 1.5.3 Kraftfordelingssystem . . . 18
  - 1.5.4 Kommunikasjonssystem . . . 18
  - 1.5.5 Motorstyrings- og reguleringssystem . . . 18
  - 1.5.6 Sensorsystem og elektronikkhus . . . 20
  - 1.5.7 Design og kontroll av ROV manipulatoren . . . 21
  - 1.5.8 Design og montering av ROV-ramme, og ytelsesanalyse av motorer . . . 21
  - 1.5.9 Mikro-ROV . . . 21

2 Motorer **22**
- 2.1 Elektrisk motor . . . 22
- 2.2 Ulike typer motorer . . . 23
  - 2.2.1 Børstemotor . . . 23
  - 2.2.2 Børsteløs motor . . . 24
  - 2.2.3 Stegmotor . . . 27
- 2.3 Valg av thrustere . . . 28
  - 2.3.1 Ulike thrustere . . . 28
  - 2.3.2 Konklusjon . . . 30
- 2.4 Valg av manipulatormotorer . . . 30
  - 2.4.1 Ulike manipulatormotorer . . . 31
  - 2.4.2 Konklusjon . . . 32

3 Motorkontrollere **34**
- 3.1 Styring av børsteløse motorer . . . 34
  - 3.1.1 Hall-effektsensor . . . 34
  - 3.1.2 BEMF . . . 35
  - 3.1.3 FOC . . . 36
- 3.2 Styring av stegmotor . . . 37
- 3.3 Krav til motorkontrollerne . . . 39
- 3.4 Valg av motorkontrollere . . . 40
  - 3.4.1 Thrustere . . . 40

3.4.2 Manipulator . . . 43
3.4.3 Konklusjon . . . 45

4 Kontakter og tverrsnitt 46
4.1 Kontakter . . . 46
4.1.1 Alternativer til kontakter . . . 46
4.1.2 Valg av kontakter . . . 48
4.2 Tverrsnitt . . . 50
4.2.1 Internt i ROV-en . . . 51
4.2.2 Eksternt i ROV-en . . . 53
4.3 Konklusjon . . . 56

5 Manipulator 57
5.1 Design av manipulatorarmen . . . 57
5.1.1 Nedre arm . . . 58
5.1.2 Øvre arm . . . 58
5.1.3 Klype . . . 59
5.1.4 Rotasjon av klypen . . . 59
5.2 Konklusjon . . . 60

6 Oppsett av mikrokontroller 61
6.1 Innledning . . . 61
6.2 Konfigurering . . . 62
6.2.1 Timere . . . 62
6.2.2 SysTick . . . 62
6.2.3 Pulsbreddemodulasjon . . . 62
6.2.4 Pulsfrekvensmodulasjon . . . 62
6.2.5 SPI . . . 63
6.3 Generelt oppsett av kode . . . 63
6.3.1 Fil- og programstruktur . . . 63
6.4 Verifisering . . . 68
6.5 Konklusjon . . . 72

7 Styringssystem 74
7.1 Innledning . . . 74
7.2 ROV-ens posisjon og dens bevegelser i vann . . . 74
7.3 Utregning av motorpådrag . . . 75
7.4 Oppsett av styrestikke . . . 78
7.5 Implementering av styredata . . . 79
7.5.1 Styring av thrustere . . . 85
7.5.2 Styring av manipulator . . . 86
7.6 Implementering på mikrokontroller . . . 87
7.6.1 Styring av thrustere . . . 91
7.6.2 Styring av manipulator . . . 94

8 Modellering 97
8.1 Hvilke frihetsgrader som skal reguleres . . . 97
8.1.1 Behovet for regulator til manipulator . . . 98
8.2 Viktige momenter for modelleringen . . . 98

8.3 Dybdeprosessen . . . 100
8.3.1 Matematisk modell av dybdeprosessen . . . 102
8.3.2 Simulering av dybdeprosessen . . . 105
8.4 Rull- og stampprosessene . . . 108
8.4.1 Matematisk modell av rull- og stampprosessen . . . 113
8.4.2 Simulering av rullprosessen . . . 120
8.4.3 Konklusjon for rull- og stampprosessen . . . 121

9 Reguleringssystem 123
9.1 Valg av reguleringsmetode . . . 123
9.2 Regulering av dybdeprosessen . . . 125
9.2.1 Konklusjon av dybderegulator . . . 135
9.3 Regulering av rull- og stampprosessen . . . 135
9.3.1 Regulering av rullprosessen . . . 136
9.3.2 Regulering av stampprosessen . . . 138
9.3.3 Konklusjon av rull- og stampprosessen . . . 139
9.4 Implementering på mikrokontroller . . . 140
9.4.1 Implementering av dybde-, rull- og stampprosessen . . . 143
9.4.2 Konklusjon av implementering på mikrokontroller . . . 145

10 Testing og tuning 146
10.1 Testing av manipulatorarm . . . 146
10.2 Testing av styrings- og reguleringssystem . . . 148

11 Diskusjon 150
11.1 Motorer og motorkontrollere . . . 150
11.2 Styringssystem . . . 150
11.3 Modellering og regulering . . . 151
11.3.1 Modell og regulator for dybdeprosessen . . . 151
11.3.2 Modell og regulator for rull- og stampprosessen . . . 152
11.4 Mikrokontroller . . . 153
11.5 Videre arbeid . . . 153

12 Konklusjon 154

13 Vedlegg 160

Kapittel 1

Innledning

Innhold

1.1 Bacheloroppgaver 1
1.2 ROV 3
1.2.1 Frittsvømmende ROV med navlestreng 3
1.2.2 Krypende ROV 4
1.2.3 Strukturelt avhengig ROV 4
1.2.4 Hymir 5
1.3 UiS Subsea 5
1.4 MATE - Marine Advanced Technology Education 7
1.4.1 MATE ROV Competition 7
1.5 Overordnet system 17
1.5.1 Blokkskjema 17
1.5.2 Bildegjenkjenning og autonom kjøring 18
1.5.3 Kraftfordelingssystem 18
1.5.4 Kommunikasjonssystem 18
1.5.5 Motorstyrings- og reguleringssystem 18
1.5.6 Sensorsystem og elektronikkhus 20
1.5.7 Design og kontroll av ROV manipulatoren 21
1.5.8 Design og montering av ROV-ramme, og ytelsesanalyse av motorer 21
1.5.9 Mikro-ROV 21

Innledningsvis presenteres årets bacheloroppgaver og definerer hva en ROV er. Deretter introduseres studentorganisasjonen UiS Subsea og MATE ROV-konkurransen vi skal delta i. Til slutt vil det overordnede systemet for ROV-en representeres med et blokkdiagram og kort oppsummering av ansvarsområdet til hver bachelorgruppe.

Dette kapittelet er skrevet av gruppen ansvarlig for Sensorsystem og elektronikkhus [2].

1.1 Bacheloroppgaver

I år er vi totalt 16 elektro- og maskin-studenter som deltar i studentprosjektet til UiS Subsea. Målet er å bygge en vellfungerende ROV som skal hevde seg i den internasjonale MATE ROV-konkurransen (beskrevet i delkapittel 1.4.1) som avholdes

i USA om sommeren. Prosjektet går samtidig ut på å ha et godt tverrfaglig samarbeid, hvor man ønsker at alle får erfaring i det å delta i en større prosjektoppgave. Samtlige av årets studenter skal skrive bacheloroppgave[1] på prosjektet, derfor har vi valgt å dele oppgavene og gruppene som følgende:

- **Elektro**
  - **Bildegjenkjenning og autonom kjøring** - Bjørnar Wiik og Jens Trydal
  - **Kraftfordelingssystem** - Andrine Pedersen og Anniken Hjelm
  - **Kommunikasjonssystem** - Martin Hausken og Oliver Langvik Veland
  - **Mikro-ROV** - Geir Arne Solland Kindingstad og Mikael Rodal Helgesen
  - **Motorstyrings- og reguleringssystem** - Edmond Baloku og Markus Haldorsen
  - **Sensorsystem og elektronikkhus** - Daniel Vasshus og Espen Myrset

[1] Oppgavebeskrivelsen for hver av oppgavene finnes i kapittel: 1.5.1

- **Maskin**
  - **Design og produksjon av ROV-manipulator** - Sindre Fjermedal og Joachim Merenyi Skjervik
  - **Design og montering av ROV-ramme, og ytelsesanalyse av motorer** - Alexander Falch Voerman og Sigvart Daniel Rodriguez Høien

Til slutt vil alle oppgavene bli satt sammen til en mikro- og en vanlig ROV.

1.2 ROV

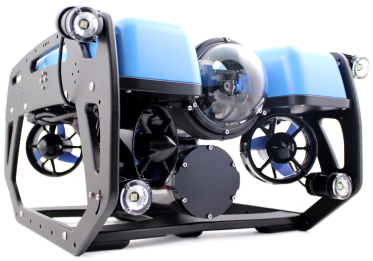
En ROV er et fjernstyrt ubemannet fartøy, som ofte omtales som ROV[2]. Disse finnes i mange størrelser og former, de minste kan være på noen få kilo, mens de største[3] kan være på størrelsen av et lite hus og veie flere tonn.

De minste ROV-ene brukes ofte til vitenskaplig forsking i form av opplæring, overvåke plante- og dyreliv, prøvetaking og gi mulighet for å inspisere plasser hvor det er uforsvarlig å sende dykkere. Større ROV-er brukes til reparasjoner, vedlikehold og inspeksjoner av konstruksjoner under vannoverflaten. Ofte vil ROV-ene være utstyrt med kamera, lys, og manipulator, men det er ikke uvanlig å utvide bruksområdet ved å installere spesialdesignet tilleggsutstyr for gitte arbeidsoppgaver.

ROV-er kan deles inn i underkategorier, og videre vil vi kort oppsummere egenskapene til disse.

1.2.1 Frittsvømmende ROV med navlestreng

Denne typen ROV er den mest vanlige og gjenkjennes med at den har en navlestreng mellom styrekonsoll og fartøy. BlueROV fra Bluerobotics som vist i figur 1.1 er en frittsvømmende ROV med navlestreng. Denne og tilsvarende fartøy blir ofte utstyrt med kamera, lys, manipulator eller spesialtilpassede verktøy for gitte arbeidsoppgaver.



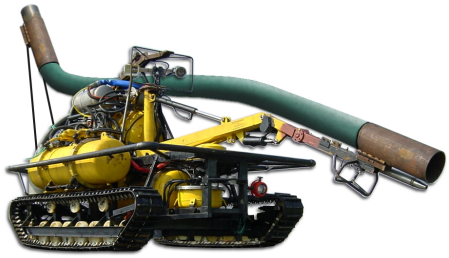
Figur 1.1: ROV med navlestreng av typen BlueROV2 fra BlueRobotics. Bilde hentet fra [3]

[2] **ROV**: Fjernstyrt undervannsfartøy (Remotely operated vehicle)

[3] "UT-1" en av verdens største er 7.8 m lang, 7.8 m bred, 5.6 m høy og veier 60 tonn. Denne brukes til å installere kabler på havbunnen

1.2.2 Krypende ROV

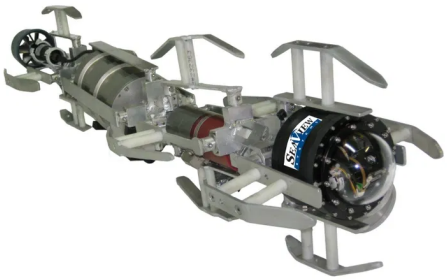
I figur 1.2 vises en krypende ROV[4]. Denne brukes til å krype langs sjøbunnen eller gjennom rør. Ofte graver den ned rør og kabler i sjøbunnen, men kan også utføre inspeksjoner eller brukes til gruvedrift på havbunnen.



Figur 1.2: Krypende ROV av typen Seabed dredger fra Seascape. Hentet fra [4]

1.2.3 Strukturelt avhengig ROV

En strukturelt avhengig ROV[5] som vist i figur 1.3 er primært brukt for å inspisere og vaske konstruksjonen den er festet til. For å bevege seg langs konstruksjonen den er festet til, brukes blant annet hjul, kabel, skinner, taljer eller hydrauliske løsninger.



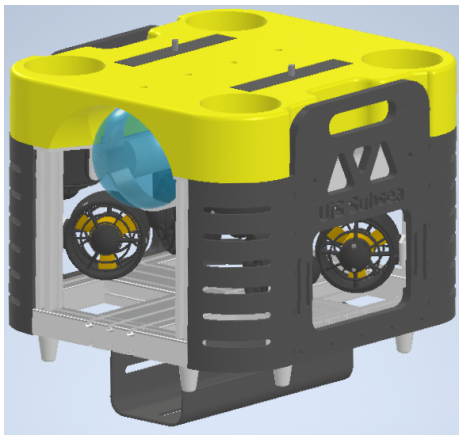
Figur 1.3: Strukturelt avhengig ROV av typen Serpent av Seaview. Hentet fra [5]

[4]På engelsk: Crawling ROV
[5]På engelsk: Structurally Reliant ROV

1.2.4 Hymir

I UiS Subsea i år skal vi bygge en ROV med navn Hymir, denne er frittsvømmende ROV med navlestreng som blir styrt av en ROV-pilot ved et kontrollpanel på land. Hymir skal brukes til å utføre diverse arbeidsoppgaver i basseng, men vi ønsker også å ha mulighet for å bruke den i ferskvann og sjø. I tillegg skal vi bygge en mikro-ROV som er festet til undersiden av ROV-en, denne skal kunne inspisere og hente ut objekter i rør helt ned til en størrelse på 6 tommer.

Hele kostnaden for prosjektet blir dekket av studentorganisasjonen UiS Subsea.



Figur 1.4: 3D modell av Hymir, med dokkingstasjon for mikro-ROV på undersiden

1.3 UiS Subsea

UiS Subsea er en studentorganisasjon ved Universitetet i Stavanger. Organisasjonen har siden 2013 hatt som mål å motivere studenter til å delta i større studentprosjekt, hvor man ønsker å skape et miljø for innovasjon, utvikle tekniske ferdigheter og vise kreativ tilnærming til problemstillinger som dukker opp. Samtidig er det svært viktig for UiS Subsea å styrke samarbeidet mellom universitetet og næringslivet. Tidligere år har det blitt utviklet både ROV og AUV-er i forbindelse med bacheloroppgaver for data-, elektro- og maskinstudenter.

Figur 1.5: Logo UiS Subsea

Vi har i løpet av overtakelsen av UiS Subsea gjort store endringer på organisasjonsstrukturen for å gjøre det mer bærekraftig for fremtidig rekruttering og fordeling av arbeid. I den sammenheng valgte vi å skille mellom styreoppgaver og prosjektoppgaver. Styret har ansvar for:

- Organisasjonen UiS Subsea
- Økonomi
- Sponsoravtaler
- Markedsføring
- Rekruttering av nye studenter
- HMS

Noen av oppgavene til prosjektledelsen er:

- Prosjektfremgang
- Finne en passende konkurranse laget skal delta i
- Passe på at prosjektet blir forsvarlig styrt
- Ansvar for at alle studentene er med og drar i riktig retning
- Påse at alle krav og retningslinjer følges

For å fordele ansvar og imøtekomme kravet til MATE ROV-konkurransen om entreprenørskap har vi fordelt rollene som følger:

- Styret
  - **Styreleder:** Daniel Vasshus
  - **Nestleder:** Geir Arne Solland Kindingstad
  - **Økonomiansvarlig:** Edmond Baloku
  - **Markedsføring:** Martin Hausken og Oliver Langvik Veland
  - **Sponsoransvarlig:** Sigvart Daniel Rodriguez og Bjørnar Wiik
  - **Styremedlemmer:** Espen Myrset og Mikal Rodal Helgesen
- Prosjektledelse
  - **Prosjektleder:** Jens Trydal

- **Teknisk ansvarlig:** Sindre Fjermedal
- **Lagleder elektro:** Markus Haldorsen
- **Lagleder maskin:** Joachim Merenyi

1.4 MATE - Marine Advanced Technology Education

Vi vil i dette delkapittelet presentere MATE ROV-konkurransen, oppgavene som skal utføres, poengfordelingen og tekniske krav til ROV-en.

MATE Center er en nasjonal organisasjon i USA som samarbeider med skoler, forskningsinstitutt, myndigheteter, militær, og institutt for marin teknikk. Formålet til organisasjonen er å bruke marin teknologi til å utfordre og inspirere studenter ved å løse problemstillinger fra den virkelige verden. For å løse problemstillingene ønsker de at studentene har en kreativ tilnærming til forskning, teknologi og ingeniørfag.

Figur 1.6: MATE logo. Hentet fra [6]

For å oppnå formålet til organisasjonen blir det årlig arrangert en internasjonal ROV-konkurranse hvor samtlige lag skal løse en bestemt problemstilling. Konkurransen har fått navnet "MATE ROV Competition"

1.4.1 MATE ROV Competition

Hvert år presenterer konkurransen nye kunder med forskjellige problemstillinger som skal løses, og i årets utgave har vi "verdenssamfunnet" som kunde. Oppgaven som skal løses er å bygge en ROV som skal takle plastproblemet i sjøen, klimaendringens påvirkning på korallrev og konsekvensen med dårlig miljøpraksis på våre vannveier. I konkurransen skal vi ikke gå inn på hva vi må gjøre for å adressere den faktiske årsaken til problemstillingen, kun løse oppgavene som er gitt.

UiS Subsea stiller i utforsker klassen[6] og er den vanskeligste klassen i MATE ROV Competition.

[6] I konkurransen heter klassen "Explorer"

Figur 1.7: Mate ROV-konkurranse logo. Hentet fra [7]

Konkurransen gir poeng for å løse tre forskjellige praktiske oppgaver, samtidig gis det poeng for størrelse og vekt. Forutenom det, får man poeng for dokumentasjon, presentasjon og sikkerhet. Mer om poengfordeling i delkappittel 1.4.1. Vi skal nå ta for oss de tre praktiske oppgavene:

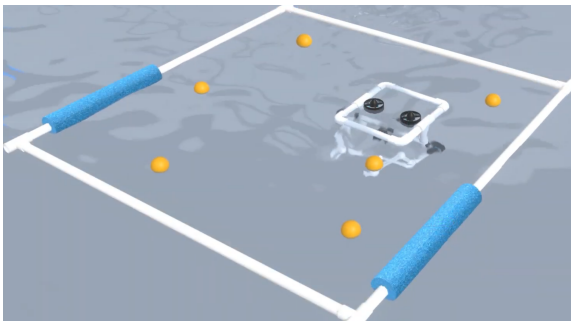
Oppgave 1: Problemet med plastforurensing (Totalt 90 poeng)

- **Søppelbøtte til sjøs - "Rydde opp havet, en havn om gangen"**
  - Koble fra gammel strømkontakt til en nyinstallert søppelbøtte - 5 poeng
  - Fjerne tidligere fangstpose av netting fra søppelbøtte - 10 poeng
  - Installere ny fangstpose av netting i søppelbøtta - 10 poeng
  - Koble til en ny strømkontakt[7] til den nyinstallerte søppelbøtta - 20 poeng

Figur 1.8: Ved at ROV-en fjerner støpselet fra søppelbøtta vil lyset slukkes. Videre i oppgaven fjerner man fangstposen med objekter i, for å så returnere med en ny fangstpose. Hentet fra [8]

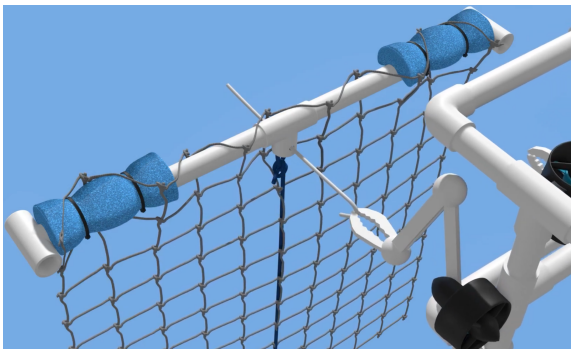
[7]Strømkontakten skal selv lages

- **Utbedring: Fjerne plastavfall fra topp til bunn**
  - Fjerne flytende plastavfall fra vannoverflaten
    - Fjerne alle 6 plastballene - 15 poeng
    - Fjerne 3 til 5 plastballer - 10 poeng
    - Fjerne 1 til 2 plastballer - 5 poeng
    - Fjerne 0 plastballer - 0 poeng



Figur 1.9: Fjerne flytende plastavfall fra vannoverflaten. Hentet fra [8]

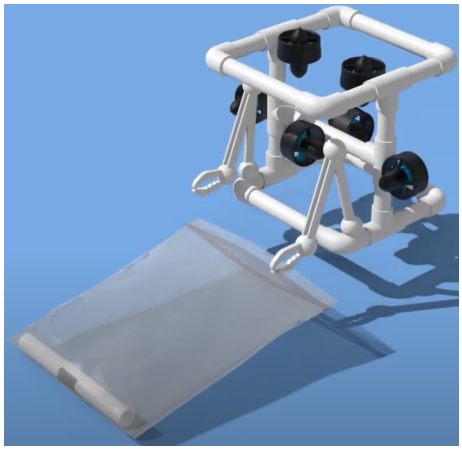
  - Fjerne et spøkelsesnett[8] fra vannet
    - Trekke ut en pinne for å simulere det å kutte spøkelsesnettet løst - 10 poeng
    - Fjerne spøkelsesnettet fra vannet - 10 poeng



Figur 1.10: Fjerne spøkelsesnett fra vannet. Hentet fra [8]

[8]Spøkelsesnett representerer et fiskenett som er forlatt eller mistet av fiskere.

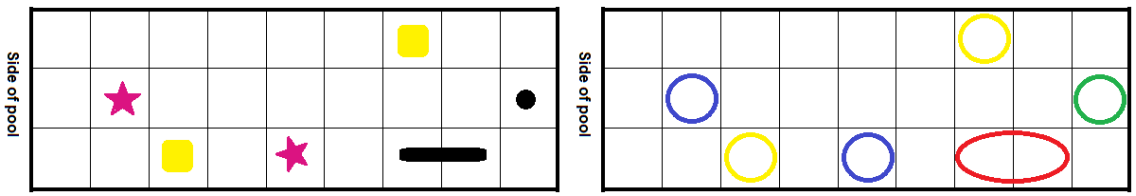
- Fjerne plastrester fra bunnen av Marianergropen[9] - 5 poeng for hver, maks 10 poeng



Figur 1.11: Fjerne plastrester fra havbunnen. Hentet fra [8]

Oppgave 2: Den katastrofale virkningen av klimaendringen på korallrev (Totalt 90 poeng)

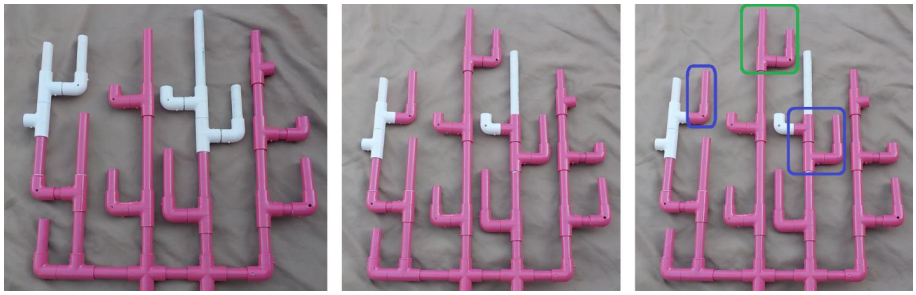
- **Kjøre i en transektlinje[10] over et korallrev og kartlegge interessepunkt**
  - Kjøre i en transektlinje over et korallrev
    - Kjøre transektlinjen autonomt - 15 poeng
    - Kjøre transektlinjen manuelt - 5 poeng
  - Kartlegge interessepunkt i korallrevet
    - Automatisk kartlegging på en dataskjerm - 10 poeng
    - Manuell kartlegging på en dataskjerm - 5 poeng



Figur 1.12: Kjøre over korallrevet til venstre i figuren. Her er det to lokasjoner for å plante ut korall-fragmenter (gul firkant) i , to sjøstjerner (lilla stjerner), en korallkoloni (svart rektangel) en svamp (svart prikk). Høyre: Fargede sirkeloverlegg i riktige ruter for vise interessepunkt og organismene på revet. Hentet fra [1]

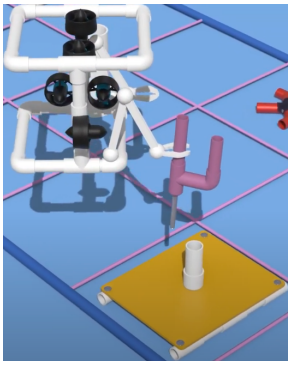
[9]Marianergropen er verdens dypeste havsgrop og ligger vest i Stillehavet og øst for Marianene
[10]En transektlinje er en linje på tvers av et habitat, eller deler av et habitat

- **Bruke bildegjenkjenning for å bestemme helsen til en korallkoloni ved å sammenligne nåværende tilstand med tidligere data**
  - Bruke bildegjenkjenning for å bestemme helsen til korallkolonien
    - Alle endringer er identifisert - 20 poeng
    - Minst en, men ikke alle endringene er identifisert - 10 poeng
    - Ingen endringer er identifisert - 0 poeng
  - Bruke en håndbok for å bestemme helsen til korallkolonien - 5 poeng



Figur 1.13: Venstre: Korallkolonien for ett år siden. Senter: Korallkolonien slik den er i dag. Høyre: Korallkolonien med alle områder identifisert, et område med vekst og to områder med gjenoppretting fra bleking/flekking. Hentet fra [1]

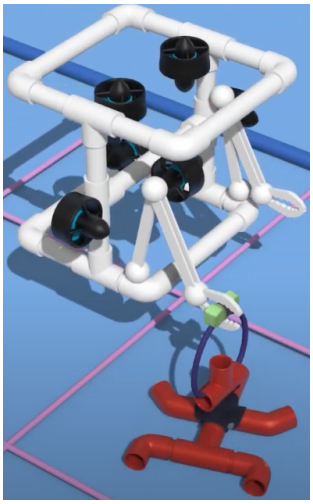
- **Gro koraller på rev**
  - Fjerne korall-fragmenter fra planteskolen[11] - 5 poeng for hver, maks 10 poeng
  - Plante ut korall-fragmenter på bestemte områder i revet - 5 poeng for hver, maks 10 poeng



Figur 1.14: Gro koraller på rev. Hentet fra [8]

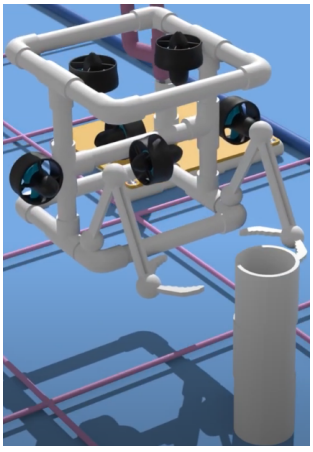
[11] Gartneri der det dyrkes planter som siden plantes ut

- **Begrense et utbrudd av tornekronesjøstjerne**[12] - 5 poeng for hver, maks 10 poeng



Figur 1.15: Begrense utbrudd av tornekronesjøstjerne. Hentet fra [8]

- **Samle prøver av svampearter for farmasøytisk forskning**
  - Hente en prøve fra en svamp - 10 poeng
  - Returnere prøven til overflaten - 5 poeng



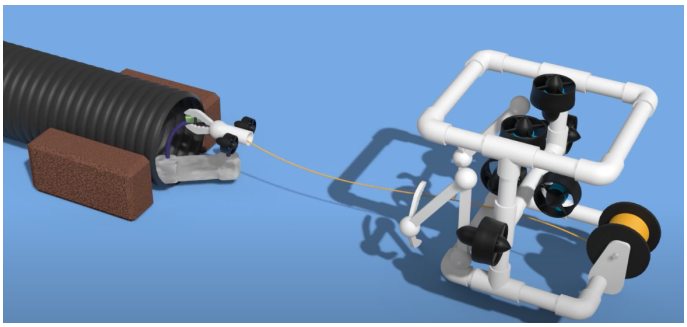
Figur 1.16: Hente prøve fra svampearter. Hentet fra [8]

[12]Tornekronesjøstjerne er en av verdens største sjøstjerner og er beryktet for å skade koraller i tropiske farvann

Oppgave 3: Vedlikeholde sunne vannveier. Del II: Delawarebukten og elva (Totalt 90 poeng)

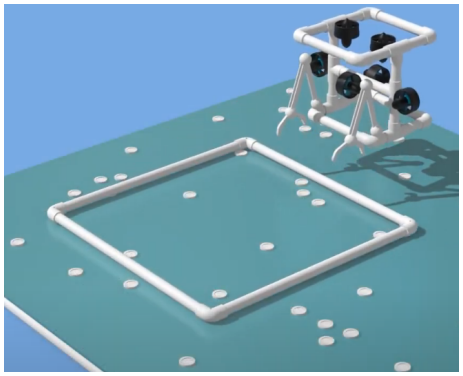
Delarwarebukten er en dyp bukt og er en forlengelse av elven Delaware på USAs nordøstlige kyst. Ferskvannet renner ut i Atlanterhavet.

- **Hente ut en sedimentprøve fra innsiden av et avløpsrør for å analysere forurensing**
  - Utplassere en enhet i røret for å hente sedimentprøven - 25 poeng
  - Returnere sedimentprøven til overflaten - 10 poeng
  - Bestemme hvilken type forurensing(er) som er tilstede i sedimentprøven - 5 poeng



Figur 1.17: Hente ut en sedimentprøve fra et 6 tommers rør. Hentet fra [8]

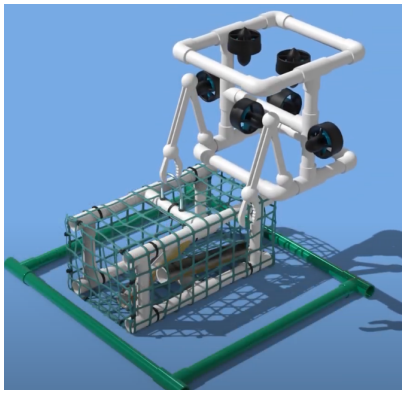
- **Estimere antall blåskjell i en blåskjell-klynge**
  - Utplassere en kvadrant og telle antall blåskjell i kvadranten - 5 poeng
  - Estimere antall blåskjell og hvor mye vann som filtreres av blåskjellene - 5 poeng



Figur 1.18: Estimere antall blåskjell

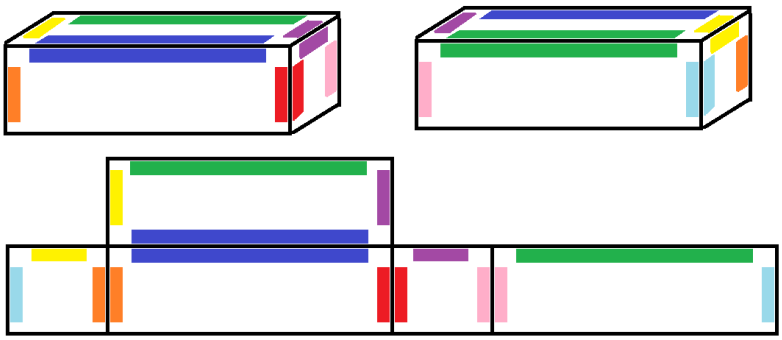
- **Restaurere ålebestanden**

- Fjerne en full åleteine fra et bestemt område - 10 poeng
- Plassere en tom åleteine i et bestemt område - 10 poeng



Figur 1.19: Fjerne og plassere ut åleteine. Hentet fra [8]

- **Lage fotomosaikk[13] av en nedsenket togvogn for å skape et kunstig rev**
  - Autonomt - 20 poeng
  - Manuelt - 10 poeng



Figur 1.20: Topp: Diagram med fargesammensetting på togvognen. Bunn: Fotomosaikk av togvognen som vist på dataskjermen. Hentet fra [1]

[13]Fotomosaikk er en samling av bilder satt sammen til ett bilde

Poengfordeling

For å kåre en vinner av konkurransen, vil samtlige poeng man har samlet, legges sammen, og laget med flest poeng blir kåret som vinner. I listen nedenfor har vi en oversikt på alle delene vi kan hente poeng på.

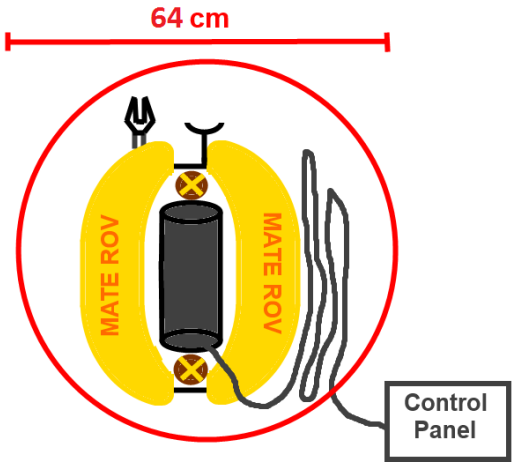
Produktdemonstrasjon	
Produktdemonstasjon (oppgaver)	270 + tidsbonus
Størrelse- og vektrestriksjoner	20
Produktdemonstrasjon av organisasjonseffektivitet	10
Prosjektering og kommunikasjon	
Teknisk dokumentasjon	100
Produktpresentasjon	100
Markedsføring	50
Selskapets spesifikasjonsark	10
Bedriftsansvar	20
Sikkerhet	
Sikkerhet og dokumentasjon gjennomgang	20
Sikkerhetsinspeskjon	30
Sikker jobb analyse (SJA)	10
Sum poeng:	**650 + tidsbonus**

Tekniske krav til ROV-en

Konkurransen (beskrevet i delkapittel 1.4.1) stiller strenge krav til ROV-ene som skal delta. Disse kravene setter føring på hvordan vi ønsker å løse våre prosjektoppgaver. I produktmanualen til konkurransen [1] er det en full oversikt over krav man må følge. Her vil vi gi en kort oversikt på de aller viktigste punktene. Spesifikke krav som angår enkelte grupper blir belyst i bacheloroppgavene hvor det er relevant.

Fysiske krav

I konkurransen er den fysiske størrelsesgrensa for å kunne delta, satt til å være maks 92 cm i diameter og kan ikke veie mer enn totalt 35 kg. Hele ROV-en, med tilleggsutstyr og navlestreng skal passe innenfor en ring som vist i figur 1.21 og veies.



Figur 1.21: ROV med verktøy og navlesteng kveilet opp ved siden av ROV-en. Hentet fra [1]

Videre får man ekstrapoeng for å være innenfor bestemte størrelse og vektklasser, som vist i tabellen under.

Størrelse	**Poeng**	**Vekt (på land)**	**Poeng**
<64 cm diameter	+ 10 poeng	<20 kg	+ 10 poeng
65.1 til 75 cm diameter	+ 5 poeng	20.01 kg til 28 kg	+ 5 poeng
75.1 til 92 cm diameter	0 poeng	28.01 til 35 kg	+ 0 poeng

Tekniske løsninger

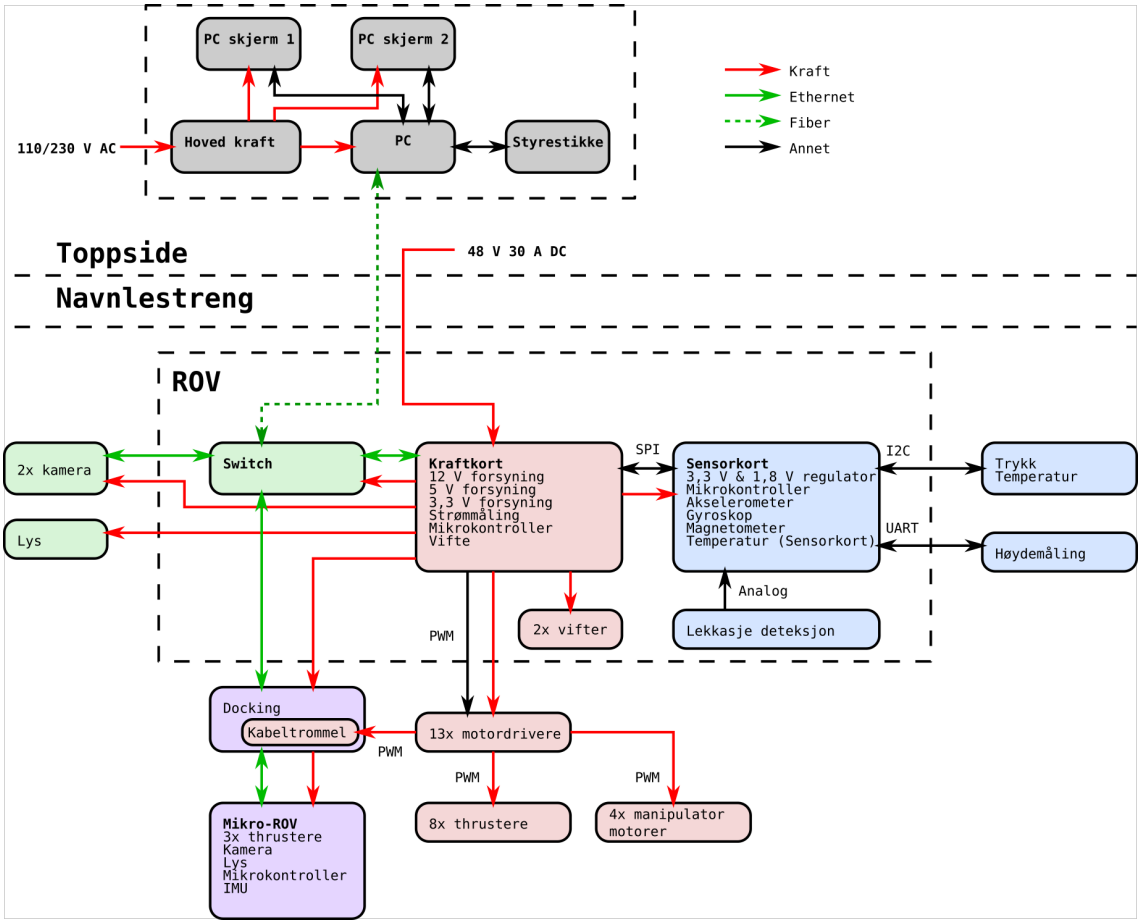
Det er kun lov å bygge én ROV til å utføre oppgavene i bassenget, men i oppgaven hvor man skal hente ut sedimentprøven (beskrevet i delkapittel 1.4.1) har man lov å bygge en mikro-ROV. Videre skal ROV-en kunne kjøre i et rent klor-basseng med en temperatur mellom 15 og 30 °C. Dybden på bassenget er maks 7 meter og samtlige oppgaver foregår innenfor 10 meter fra bassengkanten. Styrekonsollen blir plassert nær bassengkanten (maksimalt 3 meter), og navlestrengen må være lang nok til å utføre alle oppgavene.

"MATE ROV Competition" disponerer 48 V og 30 A likespenning ved styrekonsollen som skal forsyne ROV-en. Det er først lov å endre spenningsnivået på innsiden av ROV-en. Konkurransen stiller ekstra strenge krav til pneumatikk, hydraulikk og laser hvis man velger å bruke dette. Tas dette i bruk må man følge spesifikasjonene og dokumentere utstyret i henhold til konkurransemanualen [1].

1.5 Overordnet system

I dette delkapittelet vil vi presentere det overordnede systemet for ROV-en med et blokkdiagram og gi en kort oppsummering over ansvaret til hver av gruppene.

1.5.1 Blokkskjema



Figur 1.22: Blokkskjema av ROV

1.5.2 Bildegjenkjenning og autonom kjøring

Oppgaven båstår av to hoveddeler, bildebehandling og brukergrensesnitt. Den første delen går ut på å lage algoritmer som løser oppgavene om bildebehandling- og autonom kjøring i MATE ROV-konkurransen. Den andre delen går ut på å lage et nettbasert styringsprogram / brukergrensesnitt til ROV-en. I tillegg inneholder oppgaven litt om prosjektledelse.

1.5.3 Kraftfordelingssystem

Oppgaven består av å utvikle og konstruere et kraftfordelingskort for ROV, og en navlestreng for krafttilførsel fra overflatesystemet til ROV-en. Kraftfordelingskortet skal kunne forsynes med en spenning på 48 - 56 V DC med på maksimalt 30 A strøm.

Tilførsel av kraft skjer ved hjelp av navlestrengen, som også vil bli brukt til kommunikasjonsoverføring (fiberkabel). De ulike komponentene i ROV-en forsynes med ulik spenning, derfor vil det være behov for å regulere spenningen ned til 12 V, 5 V og 3,3 V på kraftfordelingskortet. For å kjøle de varmeste delene på kretskortet blir det tatt i bruk kjøleribber og vifter.

1.5.4 Kommunikasjonssystem

Oppgaven går ut på å utvikle kommunikasjonssystemet til ROV og mikro-ROV. Her tar vi av oss all kommunikasjon som skal gå mellom ROV og toppside på land. Kommunikasjonen inneholder blant annet styringsverdier og verdier fra forskjellige sensorer på ROV-en. Samtidig inneholder oppgaven valg av kamera og belysning til ROV-en.

1.5.5 Motorstyrings- og reguleringssystem

ROV-en består av 12 motorer, som må velges. For disse skal det utvikles et robust og driftssikkert styrings- og reguleringssystem, samt et robust og driftssikkert styringsystem for manipulator, for å best mulig kunne gjennomføre oppgavene i MATE ROV konkurransen. Frihetsgradene som skal reguleres er rull, stamp og hiv. Vi vil i resten av rapporten gjennomgå følgende kapitler:

- **Kapittel 2 Motorer**: Krav til motorer for fremdrift og manipulatorarm, og valg av disse.
- **Kapittel 3 Motorkontrollere**: Krav for motorkontrollere i henhold til valg av motorer, og valg av disse.
- **Kapittel 4 Kontakter og tverrsnitt**: Valg av kontakter for sammenkobling av komponenter på utsiden og innsiden av elektronikkhuset, samt beregning av tverrsnitt for kabler.
- **Kapittel 5 Manipulator**: Manipulatorarmens funksjoner, bakgrunn for design og funksjonene til de valgte motorene.

- **Kapittel 6 Oppsett av mikrokontroller**: Beskrivelse av de relevante modulene og funksjonene til mikrokontrolleren brukt til årets styrings- og reguleringssystem, samt verifisering av at disse fungerer som de skal.

- **Kapittel 7 Styringssystem**: Gjennomgang av styresystemets oppbygning, nødvendige beregninger, hvordan systemet fungerer og hvilke funksjoner det har, samt implementering på mikrokontroller.

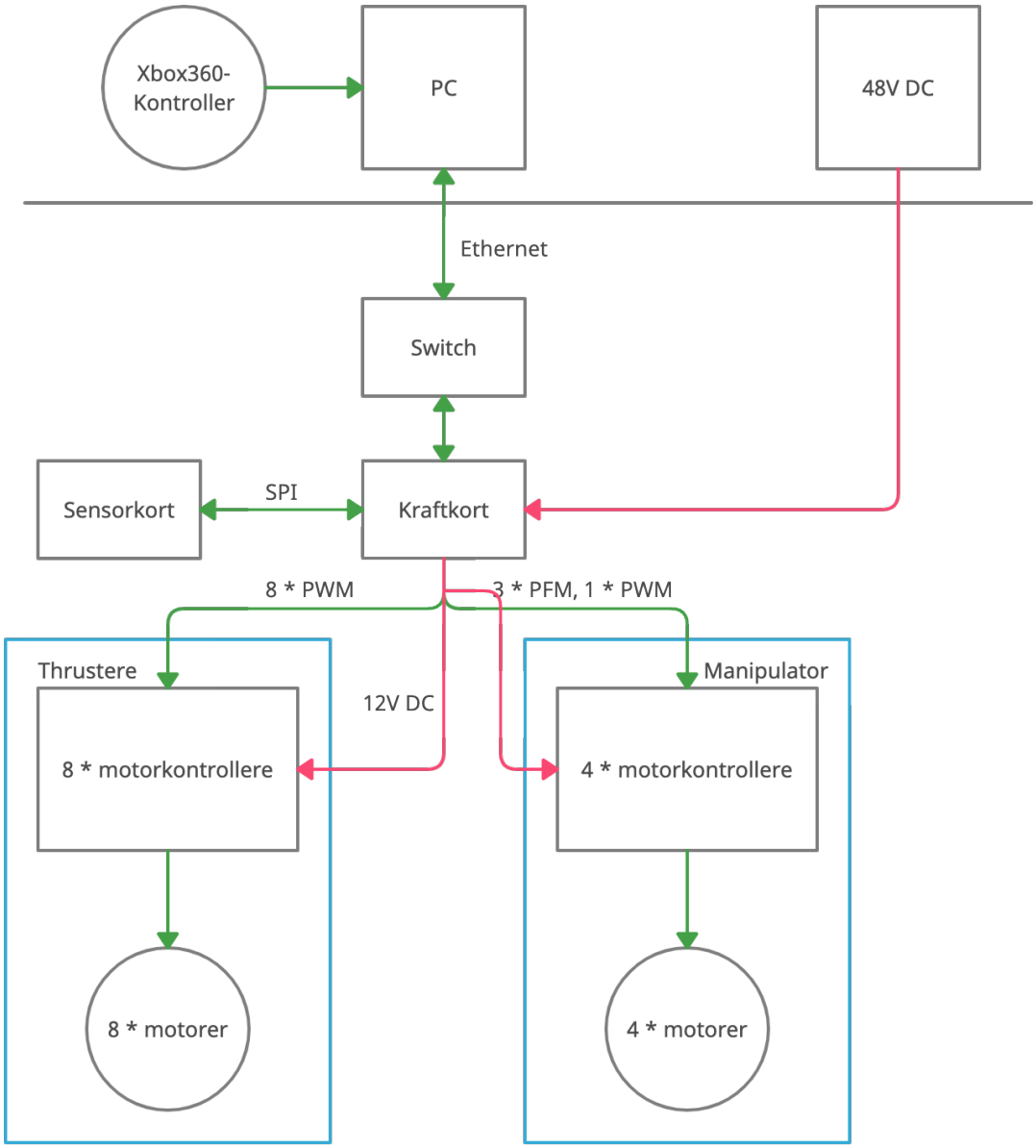
- **Kapittel 8 Modellering**: Analysering av fartøyets oppførsel under vann, og simulering av relevante prosesser for bedre forståelse av disse.

- **Kapittel 9 Reguleringssystem**: Drøfting om valgt regulator, beregning av reguleringsparametrer og implementering av reguleringssystem på mikrokontroller.

- **Kapittel 10 Testing og tuning**: Gjennomgang av hvordan systemet skal testes og tunes når alle systemene til ROV-en er klare for dette.

- **Kapittel 11 Diskusjon og konklusjon**: Diskusjon rundt resultatene av bacheloroppgaven, videre arbeid og konklusjon.

Overordnet blokkskjema for systemet som skal utvikles i henhold til denne oppgaven er dermed som vist under:



Figur 1.23: Overordnet blokkskjema av systemet som skal utvikles for Motorstyring og regulering.

1.5.6 Sensorsystem og elektronikkhus

Oppgaven består av å utvikle og konstruere sensorsystem og elektronikkhus til ROV-en. Sensorsystemet består av flere forskjellige sensorer som skal hjelpe til med å styre, overvåke og informere om status til ROV-en i vannet. Målingene som mottas kalibreres og brukes til utregninger før det kan brukes av de andre delsystemene. Elektronikkhuset inneholder alt av elektronikk, og har konnektorer for å koble sammen utstyr på utsiden med elektronikken på innsiden.

1.5.7 Design og kontroll av ROV manipulatoren

Oppgaven består av å utvikle og konstruere en manipulator samt klype for ROV-en. Manipulatoren skal være i stand til å utføre gitte oppgaver i konkurransen, samt være modulær. Det skal også lages ulike simuleringer av manipulatoren for å øke forståelsen av de ønskede bevegelser og laster.

1.5.8 Design og montering av ROV-ramme, og ytelsesanalyse av motorer

Rammegruppen konstruerer et rammedesign basert på *Design for Assembly* konseptet. Det er modulært tilpasset slik at andre komponenter kan plasseres på ROV-en. ROV-en skal ha et lavt *Center of mass* og høyt *Center of buoyancy* for å ha stabil og ha høy manøverabilitet. Alle rammekomponenter styrkeanalyseres og det foretas en thrusteranalyse for plassering av thrustere for å få geometrisk optimalisert design basert på vektklasse og størrelse. Samtidig blir Autodesk benyttes til å beregne ROV-ens faktisk bevegelighet i vann og vektoranalyser av bevegelsene i vann.

1.5.9 Mikro-ROV

For å hente ut et objekt fra et 6 tommers rør utvikles det en Mikro-ROV som skal dokkes til hoved-ROV-en som igjen forsyner den med strøm og signaler via en navlestreng. Oppgaven består av maskinvare, kretskort, programvare, design, fysikk og mekanisk arbeid.

Kapittel 2

Motorer

Innhold

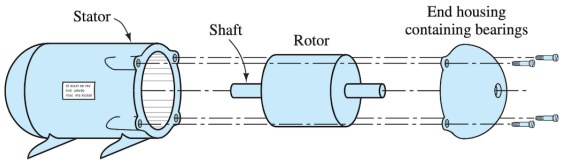
2.1 Elektrisk motor 22
2.2 Ulike typer motorer 23
2.2.1 Børstemotor 23
2.2.2 Børsteløs motor 24
2.2.3 Stegmotor 27
2.3 Valg av thrustere 28
2.3.1 Ulike thrustere 28
2.3.2 Konklusjon 30
2.4 Valg av manipulatormotorer 30
2.4.1 Ulike manipulatormotorer 31
2.4.2 Konklusjon 32

Dette kapittelet tar for seg teori om ulike elektriske motortyper, krav vi stiller til motorene og valg av motorer til både fremdrift og manipulator.

Teori er stort sett hentet fra [9], [10] og [11].

2.1 Elektrisk motor

En elektrisk motor er en maskin som omsetter elektrisk energi til mekanisk energi. Figur 2.1 viser en elektrisk motor som er bygget opp av en stillestående stator, en roterende rotor og en aksling på rotoren som kobles til den mekaniske delen. Rundt akslingen finner vi også, i noen motortyper, en kommutator. Det finnes flere ulike typer elektriske motorer som har flere ulike komponenter i tillegg.



Figur 2.1: Figuren viser oppbygning av en typisk elektrisk motor, hentet fra [9]

En elektrisk motor omsetter energi ved hjelp av elektromagnetisk induksjon. Elektromagnetisk induksjon er et fenomen som produserer elektrisk strøm ved hjelp av et

varierende magnetfelt [12]. I de fleste elektriske motorer finner vi kobberviklinger i rotoren som induserer spenning. En vikling kalles også for en spole som er en komponent som kan skape et magnetfelt. I statoren finner vi enten permanentmagneter eller feltviklinger, avhengig av hvilke type motortype det er snakk om. Ved å tilføre elektrisk strøm til motoren vil det danne seg et magnetfelt rundt viklingene i rotoren som tiltrekker magnetene i statoren, som fører til at rotoren begynner å rotere. Avhengig av type motor kan komponentene i stator og rotor være ulike.

2.2 Ulike typer motorer

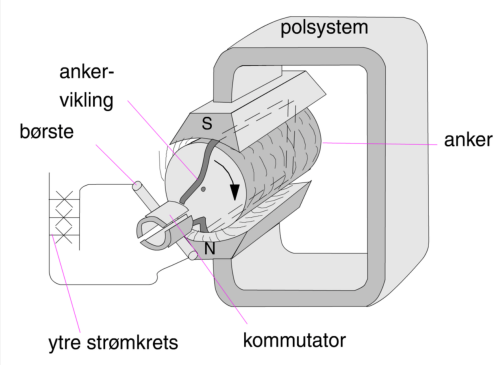
Med dagens teknologi finnes det et stort utvalg av elektriske motorer, som både kan kjøres på likestrøm og vekselstrøm. Fordelen med likestrømsmotorer over vekselstrømsmotorer er blant annet:

- Enkel installasjon
- Raskere responstid og akselerasjon
- Høyere oppstartskraft og dreiemoment
- Lineær hastighet- til dreiemomentkurve

En likestrømsmotor sine egenskaper er derfor mest egnet for bruk i ROV. Videre finnes det i hovedsak to typer likestrømsmotorer: børste- og børsteløse likestrømsmotorer. Børseløse likestrømsmotorer opererer på mange måter på samme måte som vekselstrømsmotorer, men kan ved hjelp av en motorkontroller bli styrt av likestrøm. De har dermed noen andre egenskaper enn børstemotorer, alt etter hvordan programvaren i motorkontrolleren er bygd opp. Børste- og børsteløse likestrømsmotorer har flere fordeler og ulemper, og det må derfor drøftes om hvilke motor som passer best til prosjektets formål.

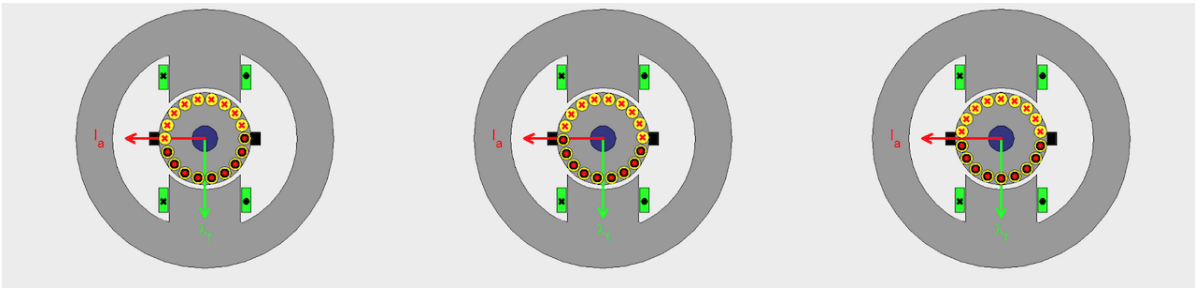
2.2.1 Børstemotor

En børstemotor er bygget opp som vist i figur 2.2 og figur 2.3. I statoren finner vi et polsystem med magneter eller feltviklinger, og på rotoren (anker) finner vi viklinger, kommutatoren og en børste. Børsten har hele tiden fysisk kontakt med rotoren, og dens oppgave er å lede strøm fra ytre strømkrets videre til viklingene.



Figur 2.2: Figuren viser oppbygningen til en børstemotor. På figuren omtales rotoren som anker og stator som polsystem, hentet fra [10].

For at rotoren skal fortsette å rotere endres retningen til den induserte spenningen i lederne, mens strømretningen i den ytre strømkrets beholdes. Dette gjøres ved hjelp av kommutatoren, og kalles kommutering. Ved å hele tiden endre den induserte spenningens retning slik at tiltrekningskraften til magnetene alltid er stor vil rotoren fortsette på rotere.

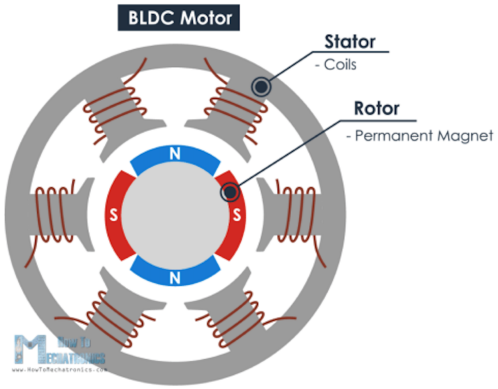


Figur 2.3: Enkel illustrasjon som viser kommutering. Den ytre grå delen er statoren, med grønne markeringer som indikerer magneter. De gule og svarte sirklene illustrerer magnetfeltet rundt rotoren der polaritet endres når de passerer kommutatoren. Kommutatoren er markert med svarte firkanter på hver sin side av rotoren. Bildet er hentet fra [10].

Den største ulempen med en børstemotor er at det kreves vedlikehold av børsten siden den hele tiden er i fysisk kontakt med statoren, og vil med tiden bli slitt ned. Dette fører til at en enten må bytte børsten eller kjøpe ny motor. Dette gjør også vanntetting av en originalt ikke-vanntett børstemotor til en stor utfordring.

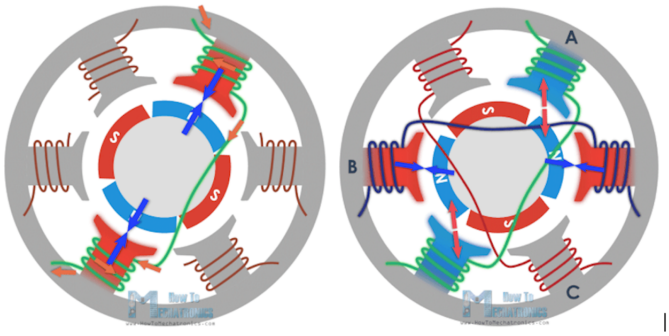
2.2.2 Børsteløs motor

En børsteløs motor er bygget som vist i figur 2.4. I statoren er det kobberviklinger og i rotoren er det permanentmagneter. Istedenfor en børste som leder strøm til viklingene, er den ytre strømkretsen koblet direkte på viklingene i statoren. Siden det ikke er en børste vil det aldri være fysisk kontakt mellom stator og rotor. Det kreves dermed ikke vedlikehold på samme nivå som børstemotoren, og levetiden er mye lengre.



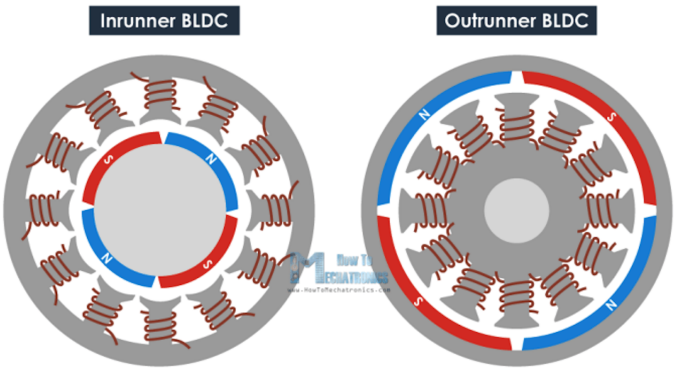
Figur 2.4: Figuren viser oppbygningen til en børsteløs motor. Ytterst er statoren med kobberviklinger, og innerst er rotoren med permanentmagneter. Bildet er hentet fra [11].

Ved å danne viklingpar mellom motstående viklinger, som vist til venstre i figur 2.5, vil en doble tiltrekningskraften. Det er i dette tilfellet da generert seks poler på statoren med kun tre faser. Videre kan effektiviteten økes ved å aktivere to spoler samtidig , som vist til høyre i figur 2.5 (markert med rødt på statoren), som fører til at en spole tiltrekkes rotoren og en annen spole fraskyves rotoren (henholdsvis viklingparene B og A).



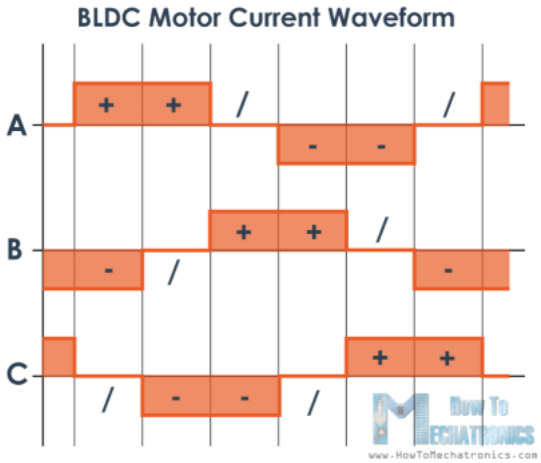
Figur 2.5: Figuren viser hvordan fasene er koblet opp til viklingparene og viser hvordan kommutering skjer i en børsteløs motor, hentet fra [11].

Det finnes to ulike oppbygninger av børsteløse motorer: 'Inrunner' og 'Outrunner'. Forskjellen er som vist på figur 2.6. Figuren viser at en 'Inrunner' har rotoren på innsiden av statoren, mens en 'Outrunner' har rotoren på utsiden av statoren. Ved å se på to motorer av samme størrelse, ser man at en 'Outrunner' vil påføre kreftene til en større diameter enn en 'Inrunner'. En større diameter vil føre til mer moment, da kreftene blir påført fra en større avstand, som betyr at motoren får en lenger momentarm. Dette betyr i praksis at en 'Inrunner' gir mulighet for høyere rotasjonshastighet, mens en 'Outrunner' har mer moment vet lik kraft og størrelse.



Figur 2.6: Figuren viser en 'inrunner' til venstre hvor rotoren er på innsiden av statoren, og en 'outrunner' til høyre hvor rotoren er på utsiden av statoren. Bildet er hentet fra [11]

Da en børsteløs motor ikke har fysisk kommutator og en børste, må kommutering skje på en annen måte. Dette kan bli gjort på både enkle og mer avanserte måter. Hovedprinsippet er at en motorkontroller aktiverer og deaktiverer fasene fortløpende. Basismetoden vises i figur 2.7, der de seks nødvendige stegene, eller intervallene, som er nødvendig for å rotere en sekspolet BLDC motor 360 grader.



Figur 2.7: Figuren viser de seks nødvendige intervallene for å rotere en sekspolet BLDC-motor 360 grader. Bildet er hentet fra [11].

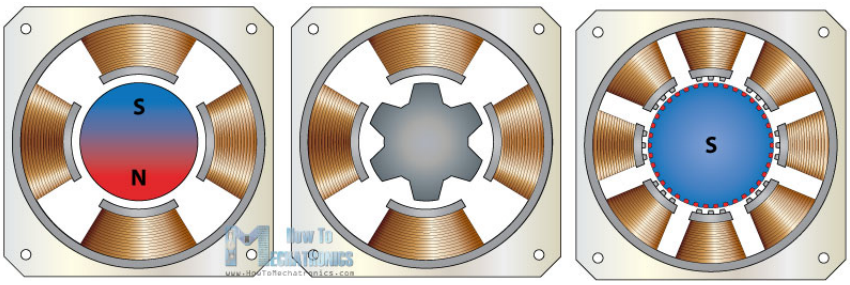
Motorkontrolleren må på en eller annen måte vite posisjonen til statoren for å vite når den skal aktivere og deaktivere fasene. Dette kan gjøres på flere måter, og blir forklart mer detaljert i kapittel 3.

Å vanntette en børsteløs motor er lettere enn en børstemotor siden det ikke er fysisk kontakt mellom rotor og stator. Dette gjør også at en børsteløs motor krever mindre vedlikehold, spesielt når den er vanntettet, som igjen fører til at den har lenger levetid. Da det er ønskelig med så lang levetid og så lite vedlikehold som mulig etter at ROV-en er satt sammen, er børsløse motorer bedre egnet for bruk i ROV enn børstemotorer.

2.2.3 Stegmotor

En stegmotor er en type børsteløs motor og er kjent for å kunne kjøre med stor nøyaktighet. Med hjelp av en motorkontroller som styrer antall pulser som blir sendt til viklingene i statoren, kan motoren rotere i en retning med et bestemt antall steg. Hvor store disse stegene er kan variere fra 0.72° (500 steg per omdreining) til 15° (24 steg per omdreining) avhengig av stegmotoren. En stegmotor blir for eksempel brukt til en laserprinter ettersom nøyaktigheter er så stor som 3% per steg. En stegmotor vil generelt sett ikke kunne oppnå samme hastighet som en børstemotor eller en børsteløs motor, men den vil ha et høyere moment ved lave hastigheter og den har også et mye høyere holdemoment[1]. Man kan også oppnå mer nøyaktighet ved å benytte seg av kjøremetoden kalt mikrosteg (microstepping), som blir forklart mer detaljert i 3.2.

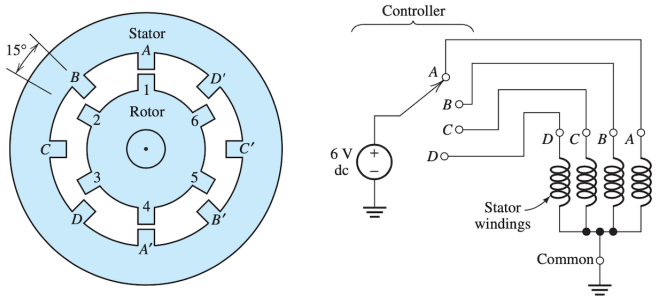
Det finnes flere ulike typer stegmotorer. De mest vanlige er permanentmagnet stegmotor som har permanentmagneter i rotoren, variabel reluktans stegmotor som har en jernrotor med et visst antall tenner, eller hybrid synkron stegmotor som er en blanding av de to forrige med permanentmagneter og flere tenner på rotoren.



Figur 2.8: Figuren viser de ulike typene steg motorer som er beskrevet ovenfor. Fra venstre: permanentmagnet-, variabel reluktans- og hybrid synkron stegmotor. Hentet fra [13].

I figur 2.9 under vises det en enkel variabel reluktans stegmotor der statoren har fire viklingpar, altså fase A er koblet til vikling A og A'. Rotoren har seks poler. Polene står 15° forskjøvet i forhold til viklingene i statoren. Hvis en sender inn en puls til vikling A vil rotoren fortsette å stå i ro siden pol 1 ligger ovenfor A. Hvis en tar vekk pulsen fra A og sender puls til B, vil pol 2 på rotoren flytte seg 15° med klokken slik at den ligger rett ovenfor vikling B. Ved å justere hvor ofte man endrer pulsene styrer man hastigheten til motoren. Det er da viktig at pulsen varer lenge nok til at rotoren får rotert til en stabil posisjon.

[1]Holdemoment er det maksimale momentet som en stegmotor kan belastes før akslingen ikke klarer å holde den stasjonære posisjonen sin.



Figur 2.9: Figuren viser et eksempel på en enkel reluktans stegmotor, til venstre vises stator med viklinger og rotor med seks poler. Til høyre vises det en spenningskilde som er koblet på en motorkontroller som er koblet videre via fire faser til fire viklingpar, hentet fra [9].

Utregning av steg per omdreining eller stegvinkel θ kan gjøres med formel 2.1

$$\text{Steg per omdreining} = \frac{360^\circ}{\theta} \tag{2.1}$$

2.3 Valg av thrustere

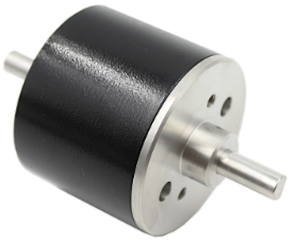
Basert på våres funn om mengde vedlikehold, vanntetting, presise styremetoder, effektivitet og varmgang, samt erfaringer fra tidligere år, har vi valgt å bruke fire børsteløse motorer for horisontal bevegelse og fire børsteløse motorer for vertikal bevegelse av ROV-en. Videre ønsker vi å finne motorer som allerede er vanntette, da det fra tidligere år har vist seg å være tidkrevende å gjøre dette selv. Kravene vi da stiller til valg av motor er:

- Må være børsteløs likestrømsmotor
- Bør være vanntett
- Må kunne kjøre på 10-48 V
- Bør være en rimelig pris innenfor UiS Subsea sitt budsjett

2.3.1 Ulike thrustere

Alternativ 1 - M200 - BlueRobotics

Motoren **M200** [14] fra figur 2.10 er en allerede vanntett motor som er designet for fremdrift av ROV. Motoren opererer på 7-20V og har maksimal effekt på 640W/32A. Motorene har en vekt på 176g. Prisen er 550 kr + eventuell toll, og sendes fra USA, så leveringstiden kan være lang. Med M200 må vi designe thrusterhus og propeller som vi må optimalisere til våres behov.



Figur 2.10: BlueRobotics - M200, hentet fra [15]

Alternativ 2 - T200 - BlueRobotics

Thrusteren **T200** [15] fra figur 2.11 er som M200 motoren ovenfor, med allerede optimalisert design for thrusterhus og propeller Thrusterhuset har en diameter på 100mm, lengde på 113mm og totalvekt på 344g. Thrusteren kan som sagt kjøres på 7-20V med maks effekt på 640W/32A. Prisen er ca. 2000 kr per thruster, og kan sendes fra JMRobotics i Kristiansand som gir en veldig overkommelig leveringstid.

Figur 2.11: BlueRobotics - T200, hentet fra [14]

Alternativ 3 - P1000 - THRUSTME

Thrusteren **THRUSTME** [16] fra figur 2.12 er en allerede vanntett thruster som opprinnelig er designet for fremdrift av kajakk og mindre fritidsbåter. THRUSTME er et nytt lokalt selskap som allerede selger produkter til flere deler av Europa. Thrusterhus med propeller følger med, og er optimalisert i forhold til selve motoren. Dette gjør at thrusteren med thrusterhus får en diameter på 121mm, lengde på 144mm og totalvekt på 800g. Thrusteren kan kjøres på 12-25.2V og har en maksimal effekt på 1000W/40A. Ifølge datablad [16] har den en skyvekraft på $16kg = 156.8N$ thrust under vann, som i praksis betyr at én thruster vil kunne akselerere et 16kg objekt med hele $a = \frac{156.8N}{16kg} = 9.81\frac{m}{s^2}$. Thrusteren er også trykktestet til 3000m. Ulempen med denne motoren er den høye prisen på hele 4500kr.

Figur 2.12: THRUSTME - P1000, hentet fra [16].

Tabell 2.1 viser en oppsummering og sammenligning av de tekniske spesifikasjonene til de ulike motorene. Merk at spesifikasjonene for M200 er uten thrusterhus og propeller som gir fysiske mindre størrelse enn T200 og P1000.

Navn	**Vanntett**	**Vekt**	**Diameter**	**Lengde**	**Volt**	**Maks effekt**	**Maks strøm**
M200	Ja	176g	36mm	62mm	7-20V	645W	32A
T200	Ja	344g	100mm	113mm	7-20V	645W	32A
P1000	Ja	800g	144mm	121mm	9-25V	1000W	40A

Tabell 2.1: Tekniske spesifikasjoner av de alternative motorene

2.3.2 Konklusjon

Et av målene til UiS Subsea 2021 var å lage en ROV med kraftigere motorer enn tidligere år. prøve Som beskrevet over oppfyller alle disse thrusterene det ønsket. Videre er pris en viktig faktor, noe en skulle tro ville gi M200 og T200 en fordel. Tvert imot viste det seg at P1000 var det mest økonomiske valget, da vi var så heldige å få tilbud om å få sponset hele 8 thrustere og tilhørende motorkontrollere av THRUSTME. Den eneste ulempen med å velge P1000 er den betydelig høyere vekten på $8 \cdot 0.8Kg = 6.4Kg$. Dette ble diskutert med resten av gruppene, og sammen konkluderte vi med å bruke P1000.

2.4 Valg av manipulatormotorer

Likens med fremdriftsmotorene ønsket vi å kjøpe inn motorer til manipulatoren som allerede var vanntette, da det vil spare oss for mye tid. Totalt skal manipulatoren ha fire ledd, og trenger dermed fire motorer.

Etter diskusjon med manipulatorgruppen [17] ble det enighet om å bruke motorer med holdemoment, for å kunne holde igjen leddene til manipulatoren i en bestemt posisjon.

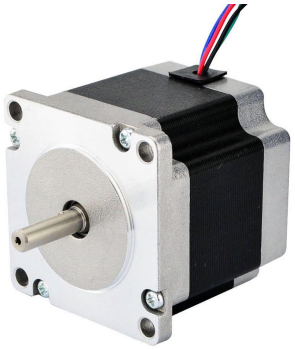
Etter mye leting etter vanntette motorer fant vi veldig få alternativer som var in-

nenfor våres kriterier og budsjett. Dette førte til at vi begynte å se på gamle motorer til manipulatorer som var laget tidligere. Av tre tilgjengelige manipulatorer var det kun den ene som hadde stegmotorer, ellers var det børsteløse motorer. Da vi ønsket å bruke stegmotorer, så vi mer på disse.

2.4.1 Ulike manipulatormotorer

Nema 23HM22-2804S

Nema 23HM22-2804S opererer på 24-48 VDC og har et strømtrekk på 2.8A per fase. Motoren har et holdemoment på 1.26 Nm. Den har oppgitt en stegvinkel på 0.9°. Fra formel 2.1 kan en regne at det er 400 steg per omdreining. Denne motoren har en veldig stor størrelse og den veier 1.271 kg med den vanntette kapselen. [18]

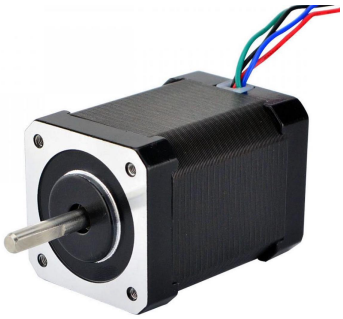


(a) Viser Nema 23 uten vanntetting, hentet fra [19].

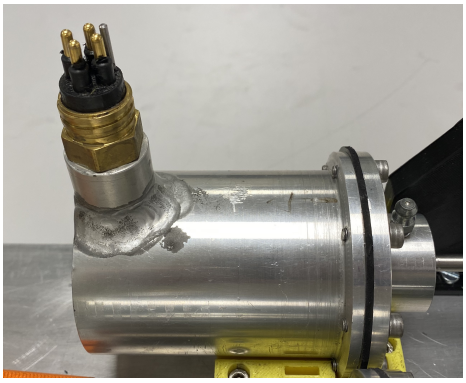
(b) Viser Nema 23 med vanntett kapsel.

Nema 17HS24-2104S

Nema 17HS24-2104S opererer på 12-24 VDC og har et strømtrekk på 2.10A per fase. Motoren har et holdemoment på 0.65 Nm. Den har opgitt en stegvinkel på 1.8°. Fra formel 2.1 kan en regne at det er 200 steg per omdreining. Motoren med den vanntette kapselen får en vekt på 0.870 kg. Vi har to slike motorer med tilhørende vanntette kapsler i beholdning. [20]



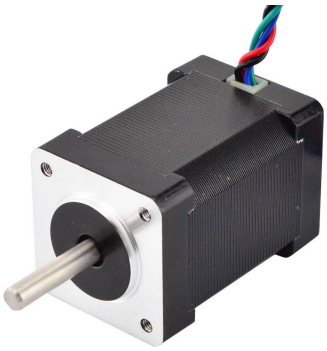
(a) Viser Nema 17 uten vanntett kapsel, hentet fra [21].



(b) Viser Nema 17 med vanntett kapsel.

Nema 14HS20-1504S

Nema 14HS20-1504S opererer på 12-24 VDC og har et strømtrekk på 1.5A per fase. Motoren har et holdemoment på 0.40 Nm. Den har oppgitt en stegvinkel på 1.8°. Fra ligning 2.1 kan en regne at det er 200 steg per omdreining. Motoren med den vanntette kapselen får en vekt på 0.636 kg. Vi har en slik motor i beholdning. [22]



(a) Viser Nema 14 uten vanntett kapsel, hentet fra [23].

(b) Viser Nema 14 med vanntett kapsel.

Multistar Elite 3508-268KV

Multistar Elite 3508-268KV er en børsteløs motor med 12 poler og en maks effekt på 330W. Den er relativ liten og veier kun 96g. Denne motoren vil trenge et ormgir for å kunne ha et tilstrekkelig holdemoment. Denne motoren er brukt i tidligere års UiS Subsea ROV-er, og erfaring fra tidligere bachelorstudenter tilsier at det er en god motor. Vi har flere av denne motoren ferdig vanntettet fra tidligere år i beholdning.

(a) Viser Multistar Elite 3508 uten vanntetting.

(b) Viser vanntettet Multistar Elite 3508.

2.4.2 Konklusjon

Hovedgrunnen til valget av manipulatormotorer er beholdning, og om de oppfyller våres krav om holdemoment og vanntetting. Det ble også diskutert med manipulatorgruppen hvilke motorer som egnet seg best til bruk [17]. Konklusjonen ble dermed å gjenbruke en de vanntette kapslingene for Nema 14, to Nema 17 og en ferdig vanntettet Multistar Elite 3508 for årets manipulator. For å unngå å bruke eldre

motorer som har vært i en lagerbeholdning over et par år, ble det bestilt inn helt nye stegmotorer av lik type. Nema 14 motoren har ansvar for klypfunksjonen, Nema 17 motorene har ansvar for nedre- og øvre arm og Multistar Elite 3508 motoren har ansvar for rotasjon til kloen. Beskrivelse av bakgrunn for design, funksjoner og motorenes roller, kan leses mer om i kapittel 5.

Kapittel 3

Motorkontrollere

Innhold

3.1 Styring av børsteløse motorer 34
3.1.1 Hall-effektsensor 34
3.1.2 BEMF 35
3.1.3 FOC 36
3.2 Styring av stegmotor 37
3.3 Krav til motorkontrollerne 39
3.4 Valg av motorkontrollere 40
3.4.1 Thrustere 40
3.4.2 Manipulator 43
3.4.3 Konklusjon 45

Dette kapittelet tar for seg teori om ulike metoder en motorkontroller styrer en motor, hvilke krav som stilles til motorkontrollerne og konklusjon om valgte motorkontrollere. Ettersom det har blitt bestemt å benytte 8 sensorløse børstemotorer fra THRUSTME for fremdrift, tre stegmotorer og en sensorløs børstemotor for manipulatoren, er det behov for totalt 12 motorkontrollere.

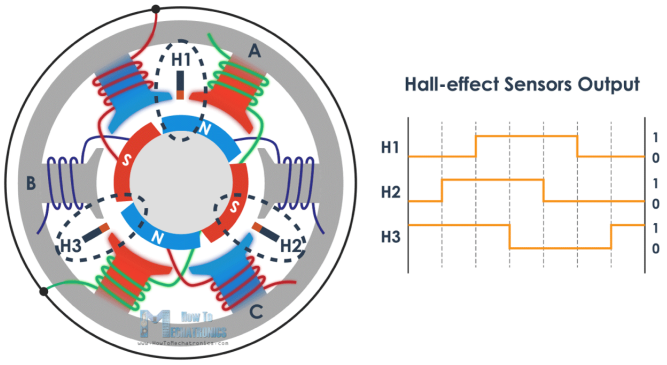
3.1 Styring av børsteløse motorer

I en børsteløs motor er det aldri kontakt mellom rotor og stator. Dette gjør det vanskelig å vite posisjonen til rotoren i forhold til statoren. En må dog vite posisjonen til rotoren i forhold til statoren for at kommuteringen kan skje. Dette er i bunn og grunn hovedsaken til at en børsteløs motor trenger en motorkontroller for å styres. En motorkontroller brukes også til å kunne endre både hastighet og retning til en motor. Det finnes flere måter en motorkontroller kan styre en motor på, ettersom det finnes BLDC-motorer både med og uten sensorer. Alle de ni børsteløse motorene vi har valgt er sensorløse. Dette kapittelet vil derfor i hovedsak ta for seg styring av sensorløse motorer. Noe teori om sensorer for BLDC-motorer blir likevell gjennomgått, for å gi et bedre bilde av mulighetene som finnes.

3.1.1 Hall-effektsensor

En Hall-effektsensor er det som er mest brukt i industrien som posisjon- og hastighetssensor på en børsteløs motor. Det er ofte en til tre sensorer som er plassert på statoren, med 60°, 120° eller 180° mellomrom. Figur 3.1 viser tre hall-effektsensorer (H1,H2,H3) med 120° mellomrom. Når en permanentmagnet på rotoren passerer

sensoren, vil sensoren sende et logisk høyt for en positiv magnet og et logisk lavt for negativ magnet (kan være motsatt) tilbake til motorkontrolleren slik at motorkontrolleren vet hvor rotoren er i forhold til statoren, og kan på den måten vite hvilke viklinger som skal kommuteres slik at rotoren kan fortsette å rotere.



Figur 3.1: Figuren til venstre viser tre hall-effektsensorer plassert med 120° mellomrom. Til høyre vises det hva sensoren sender tilbake til motorkontrolleren, hentet fra [11].

Det er mange fordeler med å bruke hall-effektsensorer. Blant annet er de omtrent immune mot miljøet rundt seg, i forhold til fuktighet, støv og vibrasjon. Sensoren er også veldig nøyaktig for å måle posisjonen og hastigheten, og har lang levetid. Ulemper med hall-effektsensorer er at høy temperatur og forstyrrelser i magnetfelt kan påvirke sensitiviteten til sensoren. I tillegg kommer det en ekstra kompleksitet ved en eventuell vanntetting av motoren på grunn av flere kabler en skal ta hensyn til. Siden vi kun har sensorløse børsteløse motorer vil vi ikke gå mer inn på bruk og implementasjon av hall-effektsensorer. [11][24][25]

3.1.2 BEMF

Teori er stort sett hentet fra [26] [27] [28].

Når en kjører en børsteløs motor vil statorviklingene bli spenningssatt. Når rotoren roterer rundt statorviklingene, vil det ifølge Lenz lov [29] bli indusert en spenning som går imot spenningen som blir tilført viklingene. Dette utgjør et spenningspotensial som kan måles, og kalles 'Back Electromotive Force' (BEMF). Ved å vite hva dette spenningspotensialet er, kan en vite hvor rotoren er i forhold til statoren.

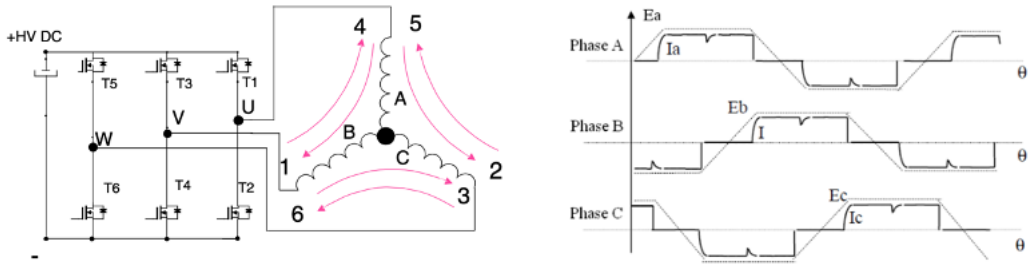
En kan finne spenningspotensialet med formel 3.1. Alle variablene utenom rotorens vinkelhastigheten er konstant, som forteller oss at spenningspotensialet er proporsjonalt med rotorens vinkelhastighet. Det vil si at rotoren er nødt til å rotere for at en kan måle BEMF.

$$\text{BEMF} = N \cdot l \cdot r \cdot B \cdot \omega \tag{3.1}$$

Hvor:
N = Antall viklinger i stator
l = Lengden av rotor
r = Radius til rotor

B = Den magnetiske felttettheten til rotor
ω = Vinkelhastigheten til rotor

BEMF spenningen som blir indusert vil være trapesformet. Figur 3.2 viser en stjernekoblet BLDC motor som styres ved hjelp av seks krafttransistorer (en motorkontroller). På høyre side vises en mer detaljert trapesformet BEMF. Figuren til høyre viser hvilke faser som blir spenningsatt, der strømmen 'I' følger en seks-trinns sekvens. Ved å benytte seg av nullpunktsgjennomgang til den fasen som ikke er tilført spenning, vil en kunne vite at spenningen til de to andre fasene er like, men med motsatt polaritet. Ved hjelp av nullpunktgjennomgangen kan en vite posisjonen til rotoren, og derfor vil krafttransistorene tilføre riktig spenning til de riktige viklingene.



Figur 3.2: Hentet fra [27].

I en seks-trinns sekvens vil kommuteringen ofte skje ved hver 60°. Tabellen under viser hvilke transistorer som er på i de ulike sekvensene, og hvilken polaritet de ulike fasene har. Ved å se på hvilke fase som er 0 vil en kunne bruke nullpunktsgjennomgang som forklart over.

Sekvens	Transistor på	Polaritet A-B-C
1	T1,T4	+,-,0
2	T1,T6	+,0,-
3	T3,T6	0,+,-
4	T3,T2	-,+,0
5	T5,T2	-,0,+
6	T5,T4	0,-,-

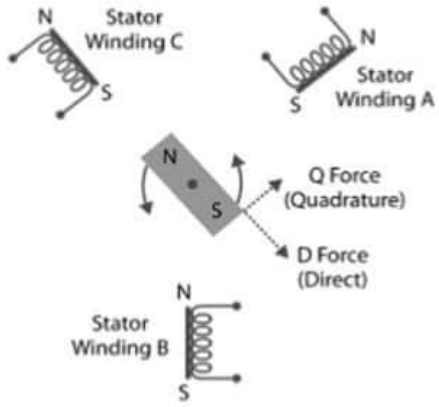
Tabell 3.1: Tabellen viser hvilke transistorer som blir skrudd på i de ulike sekvensene og hvilken polaritet de ulike fasene har ved gitt sekvens.

3.1.3 FOC

Teori er stort sett hentet fra [30], [31], [32].

FOC - 'Field oriented control' er en vektorbasert styremåte som benytter seg av strømmen i statorviklingene til å beregne posisjon og moment. Momentvektoren 'Q' og fluksvektoren 'D' som en ser på figur 3.3 er referansen som blir brukt for å styre motoren. En av fordelene med FOC er at man klarer å få mer moment per

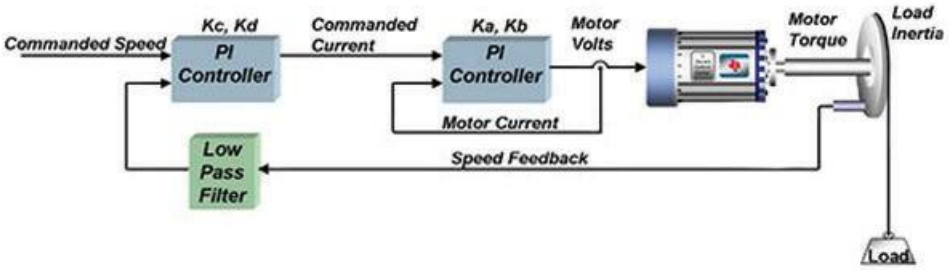
ampere. For å maksimere dette reguleres strømmen i statorviklingene, slik at fluksen produsert av rotormagnetene til en hver tid står vinkelrett på statorens magnetfelt.



Figur 3.3: Bildet viser momentvektoren Q og fluksvektoren D, hentet fra [30]

Denne metoden er moment-orientert, i motsetning til trapesformet BEMF styring som er en hastighets-orientert. Dette eliminerer store deler av momentrippelen en har ved seks trinns styring, og en vil få mye mykere og mer behagelig kjøring. Med denne metoden vil en også kunne oppnå høyere moment ved lavere hastighet. [32]

Et typisk system for styring av motor med FOC vises i figur 3.4. I de fleste algoritmene til FOC vil det være en PI-regulator som regulerer momentvektoren og fluksvektoren etter ønsket moment sendt fra bruker. Ved å øke momentvektoren Q og minske fluksvektoren D vil rotasjonshastigheten øke, og omvendt. Dersom momentet blir lavere enn ønsket moment brukeren har sendt inn, på grunn av motstand, vil regulatorene hjelpe til. De regulerer da ved å øke strømmen til motorene for å øke momentet.

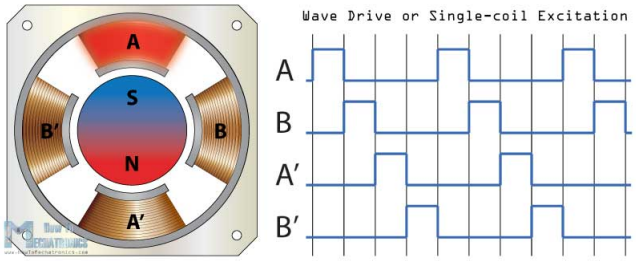


Figur 3.4: Bildet viser hvordan FOC styring blir gjort ved hjelp av PI regulator, hentet fra [30].

3.2 Styring av stegmotor

Single-Coil excitation

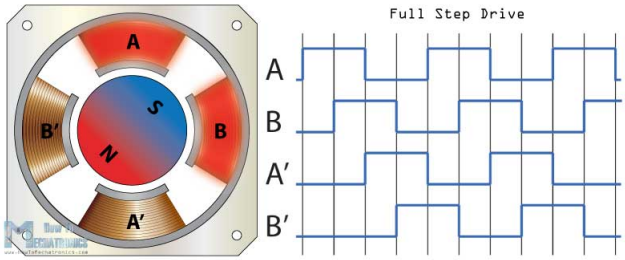
Denne metoden er lik som den som ble forklart i figur 2.9. Her aktiverer man en og en vikling. Figur 3.5 viser et eksempel på en stegmotor med fire viklinger, som dermed vil da gå en hel runde på fire sekvenser.



Figur 3.5: Figuren viser Single-Coil excitation, hentet fra [33].

Full-step mode

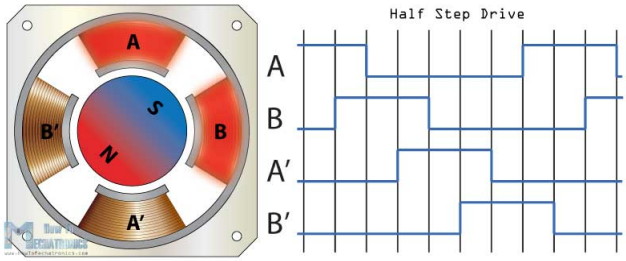
I denne styremåten har man alltid to aktive viklinger. Slik som figur 3.6 viser overlapper alltid en vikling den andre, slik at det alltid er to aktive viklinger. Dette gjør at en klarer å få høyere moment enn ved Single-Coil excitation. En hel runde er også her på fire sekvenser, som vil si at antall steg per omdreining ikke endres.



Figur 3.6: Figuren viser Full-step mode, hentet fra [33]

Half-step mode

For å øke antall steg per omdreining brukes styremåten Half-step mode. Dette er en kombinasjon av de to foregående styremåtene, men her aktiveres først en vikling, derretter to viklinger, så en vikling, deretter to viklinger, og så videre. En hel runde vil da være på åtte sekvenser, som gjør at antall steg per omdreining er doblet i forhold til foregående styremetoder med likt antall viklinger. Merk at dette vil føre til variabelt moment, da antall viklinger aktivert varierer.



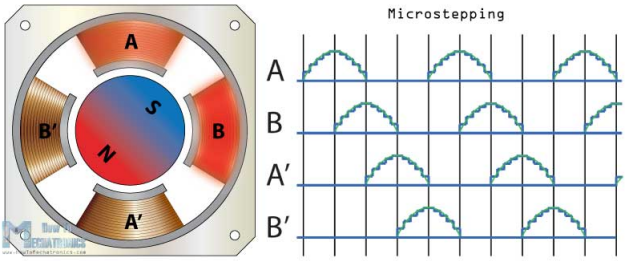
Figur 3.7: Figuren viser Half-step mode, hentet fra [33]

Microstepping

Denne styremåten er den mest vanlige for motorkontrollere idag. I denne styremåten deles hvert steg til enda mindre steg. For eksempel kan en motor med stegvinkel på 1.8° som tilsvarer 200 steg per omdreining bli delt opp 256 ganger, slik at stegvinkelen da blir $\frac{1.8°}{256} = 0.007°$. Dette tilsvarer $\frac{360°}{0.007°} = 51200$ steg per omdreining. Ved å gjøre dette vil kjøringen bli enda mykere og mer nøyaktig ved lavere hastigheter.

Microstepping blir styrt med PFM-signal, som vist i figur 3.8. Driveren tolker PFM-signalene, og sender to tilnærmede sinussignal, 90 grader forskjøvet fra hverandre, for å kontrollere strømmen i viklingene. Mens strømmen økes i den ene viklingen, minkes den i den andre. Ved å velge hvor mye strømmen inkrementeres eller dekrementeres, velger man hvor mange ganger en deler opp stegene. Med tilnærmede sinussignal menes det at driveren ikke kan produsere en helt genuin sinuskurve, men en tilnærming ved hjelp av stegvis inkrementering og dekrementering. En vil derfor ikke kvitte seg med all støy og rippel, men mye mer enn ved half-step mode og full-step mode. En kan ofte velge med fysiske brytere på en motorkontroller hvor mange steg man ønsker å dele opp i.

En ulempe med microstepping er at ved 256 oppdelinger, vil en kun oppnå omtrent 0.61% av holdemomentet sammenlignet med full-step mode, ifølge [34].



Figur 3.8: Figuren viser microstepping, hentet fra [33].

3.3 Krav til motorkontrollerne

Vi har valgt ni vanntette sensorløse BLDC motorer og tre vanntette stegmotorer. Til konkurransen er det spesifisert at motorkontrollerne må være basert på mikrokontroller, og ikke være bryter- eller relestyrt. For å spare plass i elektronikkhuset har vi et ønske om å plassere motorkontrollene på utsiden, på selve rammen til ROV-en. Dette er veldig gunstig med tanke på kjøling, da det oppgis av MATE-manualen at det maksimalt er 30° i vannet [1], mens kraftgruppen oppgir det vil være maksimalt 50° i elektronikkhuset [35]. Dette minker også plassbehovet og antall komponenter inne i selve elektronikkhuset.

I samarbeid med kraftgruppen [35] har vi fått tildelt inntil 16 A per thruster og 12 A per manipulatormotor. Det er derfor nødvendig med motorkontrollere som har mulighet til å forsyne dette.

For å oppsummere kravene vi stiller i forhold til motorene vi har:

BLDC motorene:

- Kjøre på 12-25 VDC
- Maksimalt strømtrekk på minimum 16A
- Mulighet for vanntetting / Allerede vanntettet
- Lav vekt og fysisk størrelse
- Styre sensorløse BLDC-motorer
- Mulighet for lav starthastighet

Stegmotorene:

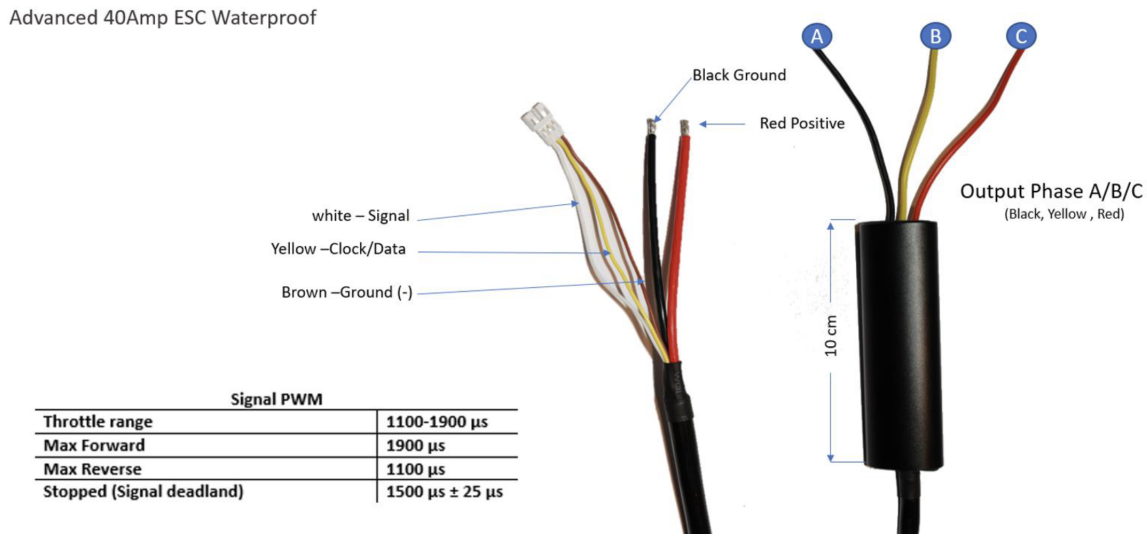
- Kjøre på 10-24 VDC
- Maksimalt strømtrekk på inntil 12A
- Mikrokontrollerbasert
- Lav vekt og fysisk størrelse
- Styre bipolar stegmotor

3.4 Valg av motorkontrollere

3.4.1 Thrustere

THRUSTME - 40Amp motor controller

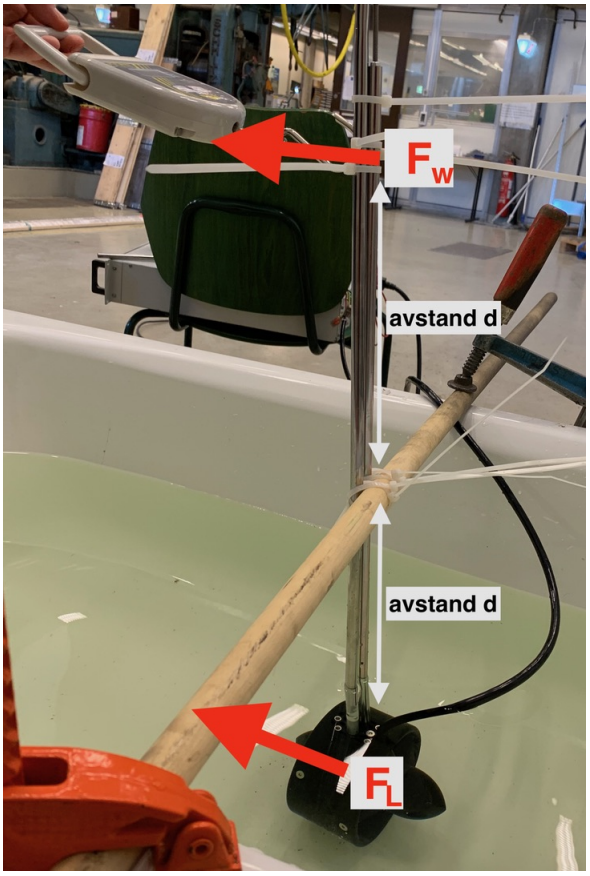
Når vi fikk en sponsoravtale med THRUSTME for åtte BLDC motorer, fikk vi også mulighet til å få sponset motorkontrollere som var optimalisert for motorene. Disse motorkontrollerne er ferdig vanntettet og kan brukes ned til 250 meters dybde, dette gir oss store fordeler da mye tid og ressurser blir spart. Motorkontrolleren kan kjøres på 9-25.2 V og opptil 40 A. Denne motorkontrolleren har også to kondensatorer på 390 μF, som er nyttig for å håndtere spenningsfall, samt forhindre støy. Sammen med kraftforsyningsgruppen ble det enighet om at dette var tilstrekkelig, ref. [35]. Motorkontrollerene benytter styremetoden FOC som gir oss fordeler ved å kjøre i lave hastigheter.



Figur 3.9: Illustrasjon av THRUSTME - 40 ampere motorkontroller, med forklarende tekst. Hentet fra datablad [16].

Vi ønsket å funksjonsteste motorkontrolleren på motoren for å få en indikasjon på om den oppfulgte kravene våre før vi tok en beslutning. Vi koblet opp motorkontrolleren til motoren og testet i et badekar på verkstedet på Universitetet i Stavanger. Det vi ønsket å teste var den laveste starthastigheten vi klarte å oppnå, både fremover og bakover. Dette var et av de viktigste kravene vi stilte. Vi ønsket også å se generelt hvordan thrusterne og motorkontrollerne oppførte seg ilag.

I bildet under viser testoppsettet som ble brukt. Det ble laget kode der PWM-signalet som sendes til motorkontrolleren kunne endres i sanntid, PWM-signalet er det som bestemmer hastigheten og retningen til motoren. PWM-signalet på motorkontrolleren har et område fra 1100 μs til 1900 μs, der 1500 er middelverdien hvor motoren står i ro. Vi kom frem til at når vi sendte 1525 μs fikk vi den laveste starthastigheten fremover hvor motoren ikke hakket og kjørte fint.



Figur 3.10: Bildet viser oppsett for måling av skyvekraft. F_L er skyvekraft fra motoren, og F_w er motkraften fra koffertvekten. Avstanden d er lik i begge retninger, som gir direkte utslag fra thrusteren til koffertvekten.

Vi ønsket å få en indikasjon på skyvekraften til thrusteren under vann, og brukte som vist i figuren en koffertvekt til å måle dette. Vi målte avstanden fra festepunktet til gjengestangen på motoren opp til det horisontale festepunktet. Koffertvekten ble plassert med lik høyde over festepunktet, som dermed ville gi en direkte måling av skyvekraft.

Ved lavest mulig starthastighet, som tidligere forklart er ved et PWM-signal med 1525 μs påtid eller $\frac{25}{400} = 6.25\%$ pådrag, fikk vi et utslag på 50 gram på koffertvekten. Vi tok deretter målinger med 5% økning i pådrag opp til 85%, og fant at stigningen var tilnærmet lineær. Ved å gi thrusteren 100 % pådrag, ble strømmen målt til å være omtrent 14 A.

Ved bruk av Newtons 2. lov, $F = m \cdot a$, har vi at:
- F = Thrust $\cdot$ Virkningsgrad, der Thrust er $0.05kg \cdot 9.81 = 0.49N$, og virkningsgraden er cos(thrusterens rotasjon fra kjøreretning)
- m = massen til ROVen

Fra rammegruppen har vi fått oppgitt at rotasjonsvinkelen på thrusterne vil være 35°, og at massen vil være ca. 35 kg [36].

Akselerasjonen for horisontal fremdrift kan da regnes ut til å være:

$$a = \frac{Thrust \cdot Virkningsgrad}{masse_{ROV}} \Longrightarrow a = \frac{0.49N \cdot cos(35°)}{35\ kg} = 0.011\ \frac{m}{s^2} \tag{3.2}$$

Vi kan kjøre ROV-en med enten to eller fire thrustere samtidig i hver horisontale retning. Med to thrustere får vi da $a = 0.011\ \frac{m}{s^2} \cdot 2 = 0.022\ \frac{m}{s^2}$, og med fire får vi $0.011\ \frac{m}{s^2} \cdot 4 = 0.044\ \frac{m}{s^2}$. I vertikal retning har vi fire motorer med 0 grader vinkel fra kjøreretning, som gir $a = \frac{0.49\ N \cdot cos(0°)}{35\ kg} \cdot 4 = 0.056\ \frac{m}{s^2}$. Tatt i betraktning at det er en del usikkerhetsmomenter som kan gi unøyaktige målinger, mener vi likevell at dette er en god nok indikasjon på at ROV-en kan kjøres med lav nok hastighet til finmanøvrering.

Vi har åtte vanntette motorkontrollere og testet derfor fire ulike for å se om vi fikk samme respons på de samme PWM-signalene, som vi gjorde. Vi skrudde av thrusterhuset og propell for å teste hva regulatoren i motorkontrolleren regulerer når vi gir motstand. Når vi ga motstand økte strømtrekket, men hastigheten ble redusert. Dette betyr at de ikke regulerer på hastighet eller effekt. Videre ble det målt på fasene til motoren, og funnet at fasestrømmene forble uendret. Dette er kjennetegnet til kjøremetoden FOC, og betyr i tillegg at motorkontrolleren regulerer momentet til motoren. Vi var ganske fornøyde med resultatet vi fikk med THRUSTME sin motorkontroller og valgte derfor å velge denne, siden den både er optimalisert for motoren, og ga gode testresultater.

3.4.2 Manipulator

Det finnes både analoge og digitale motorkontrollere til stegmotorer. Hva dette egentlig betydde var noe uklart for gruppen, og viste seg å være vanskelig å finne gode kilder på internett. Det ble derfor sendt en e-post til Leadshine [37], en produsent av motorkontrollerne for stegmotorer, for å høre om de kunne bistå med forklaring av dette, se vedlegg.

Analoge drivere er ifølge Leadshine realisert av en maskinvarekrets, mens en digital driver prosesserer og kontrollerer signalene ved en digital signalprosessor, som gir mye raskere behandling. Digitale motorkontrollere tilbyr også teknologi som gjenkjenner motorens egenskaper, og tilpasser seg til dem, genererer mindre varme, kjører mykere og en vil klare å få høyere moment med samme effekt som en analog motorkontroller. Problemet med digitale motorkontrollere er at de fleste opererer på over 24 V og alternativene på markedet er ikke alt for mange. Vi gjenbruker tre steg motorer fra 2015, og ønsket å sammenligne motorkontrollerne de brukte med andre på markedet.

Alternativ 1 - DM320T

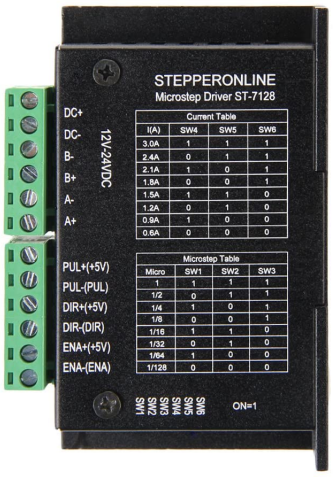
Denne digitale motorkontrolleren opererer på 10-30 VDC og har et strømtrekk på maks 2.2 A per fase. Den har også en smart funksjon som kalles 'soft start' som sørger for en myk oppstart. Styremåten er 'microstepping' hvor en kan stille hvor mange steg motoren skal ta per omdreining, fra 400-12800. den har også funksjon for å stille

maks strømtrekk en skal tillate, fra 0.3A til 2.2A per fase. Motorkontrolleren styres ved PFM-signaler. En positiv flanke vil føre til et steg, og hastigheten styres dermed ved å endre frekvensen. Motorkontrolleren tar inn styresignaler med en frekvens på inntil 60 KHz. [38].

Figur 3.11: Bildet viser DM320T med deksel. Vi ønsker å ta av dekselet for å spare plass og gjøre motorkontrolleren enda mindre. Bildet er hentet fra [39].

Alternativ 2 - ST-7128

På ROV-en fra 2015 ble den analoge motorkontrolleren ST-7128 brukt til å styre de ulike stegmotorene. Den opererer på 10-30 V, og maks 3 A per fase. Denne har også 'microstepping' som styremåte hvor en kan stille hvor mange steg motoren skal ta per omdreining, fra 400-25600. Den har også funksjon for å stille maks strømtrekk en skal tillate, fra 0.6 A til 3 A per fase. Motorkontrolleren tar inn samme type styresignaler som den digitale motorkontrolleren, med en frekvens på inntil 200 KHz. [40].



Figur 3.12: Bildet viser ST-7128 med deksel. Vi ønsker å ta av dekselet for å spare plass og gjøre motorkontrolleren enda mindre. Bildet er hentet fra [41].

3.4.3 Konklusjon

Til thrusterne var det hele tiden et ønske om å ta i bruk de vanntette motorkontrollerne sponset fra THRUSTME. Hovedgrunnen til dette er at de er designet til bruk av de valgte P1000 thrusterne, som gir thrusterne optimal oppførsel. Motorkontrollerne er også vanntettet opptil 250 meter [16], som er et stort bonus. I testingen som ble gjort var også resultatene gode, og vi fikk testet thrusterne sin oppførsel i ønsket arbeidspunkt. Det ble derfor konkludert med at det videre ville bli brukt motorkontrollerne 40Amp motor controller sponset av THRUSTME.

Til stegmotorene ble både DM320T og ST-7128 bestilt for å teste hvordan de ulike stegmotorene oppførte seg. Det var uten tvil at den digitale motorkontrolleren DM320T ga en mykere og jevnere oppførsel til stegmotorene i både lave og høye hastigheter. Når selve manipulatoren var ferdig konstruert ble det gjort en enkel test sammen med manipulatorgruppen [17]. Denne testen hadde intensjon om å finne hvilken mikrostegparameter som skulle brukes. Av de ulike mulighetene ble det konkludert med at 12800 $\frac{pulser}{omdreining}$ var best egnet til manipulatorens formål, da denne innstillingen ga mest moment til selve motorene, og mer en nok hastighet, mer om testing av stegmotorene og manipulatoren kan leses i kapittel 10. Det ble derfor konkludert med at DM320T var den mest optimale til styring av stegmotorene til manipulatoren.

Kapittel 4

Kontakter og tverrsnitt

Innhold

4.1 Kontakter 46
4.1.1 Alternativer til kontakter 46
4.1.2 Valg av kontakter 48
4.2 Tverrsnitt 50
4.2.1 Internt i ROV-en 51
4.2.2 Eksternt i ROV-en 53
4.3 Konklusjon 56

Dette kapittelet tar for seg kontakter som skal brukes til å koble sammen kabler fra mikrokontrolleren inne i elektronikkhuset, gjennom en bakplate og videre ut til motorkontrolleren som igjen går videre til motorene. Det er viktig å bestemme riktig tverrsnitt på kablene både inni og utenfor elektronikkhuset, for å unngå varmgang og spenningsfall.

4.1 Kontakter

Valg og oppsett av kontakter til ROV-ens elektronikkhus er veldig viktig. Gode kontakter sørger for solid vanntetting, lav vekt, enkel til- og frakobling og lite varmgang. Hvilke kontakter/penetrator vi velger baseres på hvilke krav vi stiller i disse kategoriene, og er som følger:

- Kontakene må være vanntette til minimum 100 meter.
- Kontakene må veie så lite som mulig.
- Kontaktene bør være enkle å koble på/av, da vi vil ha mulighet til å fjerne elektronikkhuset fra rammen enklest mulig.
- Kontaktene må ha et tverrsnitt som oppfyller krav til maksimalt strømtrekk.
- Kontaktene må være kompakte nok til at vi får plass på bakplaten til elektronikkhuset.

4.1.1 Alternativer til kontakter

Ved å se på løsninger på tidligere ROV-er, samt undersøkelse av alternativer til nye løsninger, kom vi frem til disse alternativene:

- **MacArtney**
  Felles for alle kontaktene fra MacArtney er at de har en oppgitt vanntetting på minst 300 bar, som tilsvarer 3000 meter under vann, og at kontaktene er svært brukervennlige for til- og frakobling.

  – **SubConn Micro Circular**
  SubConn Micro Circular kontakter er sirkulære kontakter med et areal på $\pi \cdot 7.75^2 = 188.59mm^2$. De har mulighet for 2, 3, 4, 5, 6 eller 8 pinner, og er spesifisert for inntil 10 A per pinne for 2-4 pinner, og 5 A per pinne for 5-8 pinner.

  – **SubConn Circular**
  SubConn Circular kontakter er sirkulære kontakter med et areal på $\pi \cdot 15.5^2 = 754.38\ mm^2$. De har mulighet for 6, 8 og 10 pinner, og er spesifisert for inntil 10 A per pinne.

Figur 4.1: SubConn Circular Series, hentet fra [42].

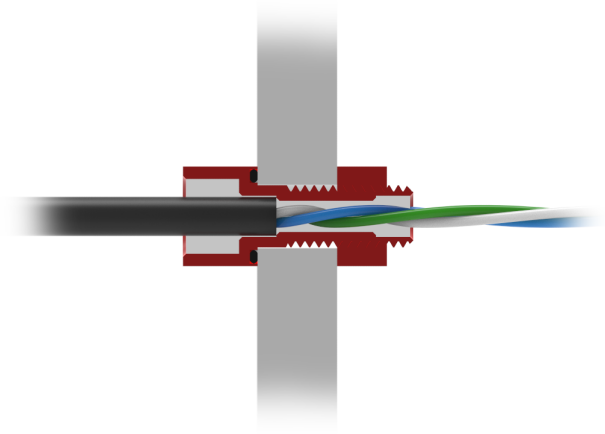
  – **SubConn Low Profile - 9 contacts**
  SunConn Low Profile - 9 contacts er rektangulære kontakter med vinklet tilkobling. De har et areal på $31.5 \cdot 44.5 = 1402\ mm^2$ og er spesifisert for inntil 10 A per pinne.

  – **SubConn Micro Low Profile - 9 contacts**
  SubConn Micro Low Profile - 9 contacts er rektangulære kontakter med vinklet tilkobling. De er utformet likt som SubConn Low Profile - 9 contacts, med et mindre areal på $25 \cdot 34.5 = 832.5\ mm^2$ og tåler inntil 5 A per pinne.

Figur 4.2: SubConn Micro Low Profile Series, hentet fra [43].

  – **SubConn Power Battery**
  SubConn Power Battery er sirkulære kontakter med et areal på $\pi \cdot 15.5^2 = 754.38mm^2$. De har mulighet for 2, 3 og 4 pinner, og er spesifisert for inntil 25 A per pinne. Denne har en enda høyere oppgitt vanntetting på inntil 800 bar, som tilsvarer 8000 meter under vann.

- **BlueRobotics**



Figur 4.3: M10 Kabelpenetrator, hentet fra [44].

  - **M10 Cable Penetrator for 8mm Cable**
    M10 Cable Penetrator er en liten og kompakt løsning. Kabelen tres gjennom penetratoren og limes fast med epoksy. Penetratoren skrues deretter fast i bakplaten, og holdes tett ved hjelp av en o-ring. Penetratoren har en oppgitt vanntetting på inntil 100 meter. Grunnet at penetratorene må limes fast i kabelen og skrues fast i bakplaten, er de ikke brukervennlige for til- og frakobling fra elektronikkhuset.

Alle kontaktene er testet for bruk i vann på inntil 100 meters dybde eller mer. BlueRobotics sin penetrator må dog tettes med epoksy, som gir mulighet for lekkasje grunnet feil i tettingsprosessen. MacArtney sine kontakter er spesifisert til et trykk på 300 og 800 bar i vann, som tilsvarer 3000 og 8000 meter dybde. Dette er mye mer enn resten av ROV-en spesifiseres til, og gir dermed en veldig god sikkerhetsmargin.

BlueRobotics sine penetratorer er helt klart gunstige med tanke på vekt. MacArtney sine kontakter er noe tyngre, men gir og mer robusthet. Videre vil vi måtte ofre muligheten for lett av- og påkobling for å ta nytte av den lave vekten til penetratorene.

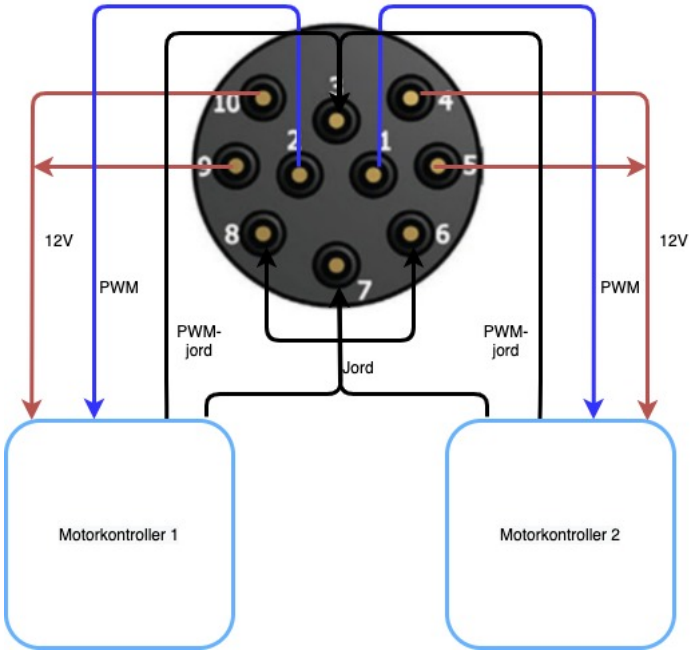
MacArtney's kontakter har et maksimalt strømtrekk per pinne på enten 5A eller 10A, som spesifisert over. For manipulatormotorene er dette innenfor kravene. For hovedthrusterne kreves det at hver thruster skal kunne trekke opp mot 16A. Dermed må vi parallellkoble spenningen for å oppnå ønsket tverrsnitt, ved $2 \cdot 10A$ pinner per thruster. BlueRobotics sine penetratorer har plass til kabel med 8 millimeter diameter, som er nøyaktig det samme som originalkabel til motorkontroller. Penetratorene oppfyller dermed krav om tverrsnitt.

4.1.2 Valg av kontakter

Gjennom avstemning ble det bestemt av hele UiS Subsea at alle skulle unngå å bruke penetratorer, da dette ville gi en mindre utfordring ved frakobling av elektronikkhuset etter montering. Dermed ble de eliminert fra alternativene våres. Videre valgte vi å se bort fra SubConn Power Battery kontaktene, da de både er veldig mye

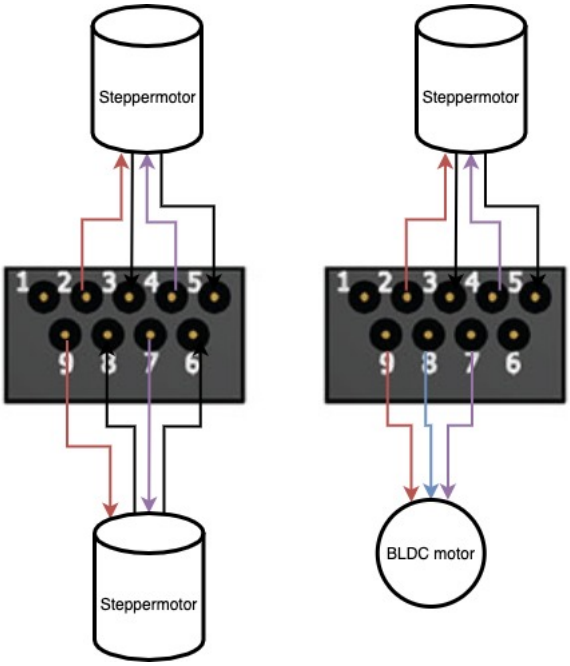
større enn de andre alternativene, samt unødvendig overdimensjonert tversnitt og vanntetthet.

For å minske antall kontakter valgte vi å koble to thrustere på samme kontakt, og dermed halvere antall kontakter. Dette gjøres ved å spleise jorden til thrusterne til tre pinner, istedenfor å bruke to pinner hver. Det samme gjøres med PWM-jorden, slik at vi bruker 1 pinne istedenfor 2. Da vi ønsker å måle strømtrekket til hver motor fra forsyningen, kan ikke positiv spenningskabel spleises, og krever 2 pinner hver. Dermed kan vi koble to thrustere på en kontakt med totalt 10 pinner, der hver pinne er spesifisert for 10 A. SubConn Circular er de eneste kontaktene som oppfyller disse kravene, og vi har derfor valgt å benytte 4 av disse til 8 motorkontrollere.



Figur 4.4: Skjermategning av kobling fra kontakt til motorkontrollere. Bilde av kontakt hentet fra [45].

Videre trenger vi $3 \cdot 4$ pinner til stegmotorene og 3 pinner til manipulatorens BLDC motor, som totalt utgjør 15 pinner.



Figur 4.5: Skjermategning av kobling fra kontakt til stepper- og BLDC-motorer. Bilde av kontakt hentet fra [46].

For å minske antall kontakter har vi som vist over igjen valgt å koble to motorer på samme kontakt, for å halvere antall kontakter. Disse signalene krever ikke mer enn 5A hver, da maksimalt strømtrekk er 4.2A (ref. delkapittel 3.4.2) og det kan dermed kobles på 2 stk 9 pinnede Micro Low Profile Series, uten parallellkoblinger.

Disse valgene gir totalt 6 kontakter. Sensorgruppen [2] har tatt på seg ansvar for å designe og utvikle elektronikkhuset og dens bakplate, og det er sammen med dem konkludert med at denne løsningen er optimal og gjennomførbar, med tanke på vekt, av/på-kobling, tverrsnitt og størrelse.

4.2 Tverrsnitt

Riktig valg av kabeltverrsnitt er viktig for at systemet skal fungere optimalt. For lite tverrsnitt kan føre til høye temperaturer, som i verste fall kan føre til at kabler kan smelte og brann kan oppstå, eller at det oppstår spenningsfall som kan føre til svikt i motorkontrollerne. Thrusterne for fremdrift opererer på 12V og har maksimalt tillatt strømtrekk på 16A, som utgjør en maksimal effekt på 192W. Manipulator motorene opererer på 12V og har maksimalt strømtrekk på 4.2A.

Teori, formler og konstanter videre i dette kapittelet er hentet fra [47], om ikke annet er spesifisert. Det anbefales å tillate et maksimalt spenningsfall til $\Delta U = 3\,\%$. Elektronikkhuset er spesifisert med operasjonell temperatur på maksimalt 50°, ifølge sensorgruppen [2]. Dette gir grunnlag for at det kan brukes kabel med PVC som isolasjonsmateriale, da dette tåler en temperatur opp mot 70°før isolasjonen smelter.

Formel 4.1 omhandler minimum tverrsnitt[1] for de gitte verdiene i formelen, utledet fra [[47], side 178].

$$A = \frac{(enfaseogtofase)/trefase \cdot I_b \cdot \rho \cdot l \cdot cos(\phi) \cdot 1.2}{\Delta U \cdot U} \tag{4.1}$$

Hvor:
- A = Tverrsnitt [mm^2].
- (enfase og tofase)/trefase = 2 for enfase eller tofase, og $\sqrt{3}$ for trefase.
- I_b = Belastningstrøm [A].
- ρ = Resistivitet [$\Omega \cdot mm^2/m$].
- l = Kabellengden [m].
- cos(ϕ) = Effektfaktor til lasten, i dette tilfelle er den 1.
- 1.2 = Konstant ved ledertemperatur på 70°for PVC kabel.
- ΔU = Maksimalt tillatt spenningsfall [%].
- U = Nominell spenning [V].

Det er ønskelig å ha et tverrsnitt som er robust, lett håndterlig, og ikke minst som kan lede strømmen kabelen er tiltenkt til uten temperaturøkning og/eller spenningsfall utenfor spesifisert område. Det kan med det være lurt å ta hensyn til diverse faktorer. Dersom en øker tverrsnittet kan den maksimale ledningsevnen til kabelen regnes ut ved å snu formel 4.1 slik:

$$I_{Kabel} = \frac{A \cdot V \cdot \Delta U}{\rho \cdot cos(\phi) \cdot 1.2 \cdot (enfaseogtofase)/trefase} \tag{4.2}$$

Dersom flere kabler ligger ved siden av hverandre vil det være fare for at temperaturen øker. Det er derfor lurt å ta hensyn til korreksjonsfaktor for nærføring, som med 12 kabler = 0.45, og korreksjonsfaktoren ved 50° som er den maksimale antatte temperaturen i elektronikkhuset, som er lik 0.71, hentet fra [47] [48].

$$I_{Kabel,korrigert} = I_{Kabel} \cdot 0.71 \cdot 0.45 \tag{4.3}$$

$I_{Kabel,korrigert}$ må være større enn belastningsstrømmen I_b for at tverrsnittet skal være tilstrekkelig.

4.2.1 Internt i ROV-en

Fra mikrokontroller til kontaktene

Kablene vil gå fra mikrokontrolleren til kontaktene på bakplaten i elektronikkhuset (disse kontaktene går videre til motorkontrollerne for thrusterne). Denne lengden blir oppgitt fra sensorgruppen til å være maksimalt 0.30 meter [2]. Ved å sette inn verdiene nevnt over får en tverrsnittet:

$$A = \frac{2 \cdot 16A \cdot 0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30m \cdot 1 \cdot 1.2}{12V \cdot 0.03} = 0.56\ mm^2 \tag{4.4}$$

[1] Gjelder for tverrsnitt under 25 mm^2

Denne verdien er minimalt tverrsnitt for en enkelt kabel med inntil 16A strømtrekk. Som nevnt over er det lurt å ta hensyn til korreksjonsfaktorer for nærføring og temperatur. Med en økning på tverrsnittet til 2 mm^2 får en da maksimal ledeevne:

$$I_{Kabel} = \frac{2.00\ mm^2 \cdot 12\ V \cdot 0.03}{0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30\ m \cdot 1 \cdot 1.2 \cdot 2} = 57.14\ A \tag{4.5}$$

Ved å korrigere ledeevnen med korreksjonsfaktorene nevnt over får en den faktiske ledeevnen til å være:

$$I_{Kabel,korrigert} = 57.14 \cdot 0.71 \cdot 0.45 = 18.26\ A \tag{4.6}$$

18.26 A ledeevne er innenfor kravet for 16A strømtrekk, og en velger derfor 2 mm^2 tverrsnitt for disse.

Fra mikrokontroller til manipulatorens motorkontrollere

Kablene vil gå fra mikrokontrolleren til motokontrollerne for manipulatorens motorer, som befinner seg inne i elektronikkhuset. Denne lengden er oppgitt fra sensorgruppen til å være maksimalt 0.15 meter [2]. Fra mikrokontrolleren til motorkontrollerne er maksimalt tillatt strømtrekk 12 A.

$$A = \frac{2 \cdot 12A\ \cdot 0.0175\ \Omega\frac{mm^2}{m} \cdot 0.15\ m \cdot 1.2 \cdot 1}{12\ V \cdot 0.03} = 0.21\ mm^2 \tag{4.7}$$

Som tidligere forklart er ønskelig å øke tversnittet slik at det blir mer håndterlig, med tanke på nærføring og temperatur. Det økes derfor til 1.00 mm^2, som gir:

$$I_{Kabel} = \frac{1.00\ mm^2 \cdot 12\ V \cdot 0.03}{0.0175\ \Omega\frac{mm^2}{m} \cdot 0.15\ m \cdot 1 \cdot 1.2 \cdot 2} = 57.14\ A \tag{4.8}$$

$$I_{Kabel,korrigert} = 57.14\ A \cdot 0.71 \cdot 0.45 = 18.26A \tag{4.9}$$

18.26 A ledeevne er innenfor kravet for 12A strømtrekk, og en velger derfor 1 mm^2 tverrsnitt for disse.

Fra stegmotorenes motorkontrollere til kontaktene

Kablene vil gå fra den børsteløse motorens motorkontroller som befinner seg inne i elektronikkhuset, til kontaktene som er plassert på bakplaten. Denne lengden er oppgitt fra sensorgruppen til å være maksimalt 0.12 meter [2].

$$A = \frac{2 \cdot 4.2\ A \cdot 0.0175\ \Omega\frac{mm^2}{m} \cdot 0.12\ m \cdot 1.2 \cdot 1}{12\ V \cdot 0.03} = 0.06\ mm^2 \tag{4.10}$$

Som tidligere forklart er ønskelig å øke tversnittet slik at det blir mer håndterlig, med tanke på nærføring og temperatur. Tverrsnittet økes derfor til 1.00 mm^2, som gir:

$$I_{Kabel} = \frac{1.00\ mm^2 \cdot 12\ V \cdot 0.03}{0.0175\ \Omega\frac{mm^2}{m} \cdot 0.12\ m \cdot 1 \cdot 1.2 \cdot 2} = 71.43\ A \tag{4.11}$$

$$I_{Kabel,korrigert} = 71.43\ A \cdot 0.71 \cdot 0.45 = 22.82\ A \tag{4.12}$$

22.82 A ledeevne er innenfor kravet for 4.2 A strømtrekk, og en velger derfor 1 mm^2 tverrsnitt for disse.

Fra manipulatorens BLDC-motorkontroller til kontaktene

Kablene vil gå fra manipulatorens motorkontroller for BLDC-motoren som befinner seg inne i elektronikkhuset, til kontakene som er plassert på bakplaten. Denne lengden er oppgitt fra sensorgruppen til å være maksimalt 0.12 meter [2].

$$I_{kabel} = \frac{P}{U \cdot \sqrt{3}} = \frac{144W}{12V \cdot \sqrt{3}} = 6.93\ A \tag{4.13}$$

$$A = \frac{\sqrt{3} \cdot 6.93\ A \cdot 0.0175\ \Omega\frac{mm^2}{m} \cdot 0.12\ m \cdot 1.2 \cdot 1}{12\ V \cdot 0.03} = 0.08\ mm^2 \tag{4.14}$$

0.08 mm^2 er altså minimum tverrsnitt. Ved å øke tevrrsnitt til $1mm^2$ får en lederevnen:

$$I_{Kabel} = \frac{1\ mm^2 \cdot 12\ V \cdot 0.03}{0.0175\ \Omega\frac{mm^2}{m} \cdot 0.12\ m \cdot 1 \cdot 1.2 \cdot \sqrt{3}} = 82.47\ A \tag{4.15}$$

Med de nevnte korreksjonsfaktorene over blir da den korrigerte lederevnen:

$$I_{Kabel,korrigert} = 82.47\ A \cdot 0.71 \cdot 0.45 = 26.34\ A \tag{4.16}$$

26.34 A ledeevne er innenfor kravet for 6.93 A strømtrekk, og en velger derfor 1 mm^2 tverrsnitt for disse.

4.2.2 Eksternt i ROV-en

Utenfor elektronikkhuset skal det gå kabler fra bakplaten $\longrightarrow$ motorkontrollerne $\longrightarrow$ thrusterne og kabler fra bakplaten $\longrightarrow$ manipulatorens motorer. Det er antatt at omgivelsestemperaturen vil være lavere enn på innsiden av elektronikkhuset. Fra

delkapittel 1.4 i innledningen, under avsnittet om tekniske løsninger, ble det beskrevet at klor-bassenget har en temperatur på 15-30°. Det er lurt å ta høyde for verst mulig tilfelle, ved 30° er korreksjonsfaktor lik 1.00. Som forklart i delkapittel 4.1.2 har hver kabel fra kontaktene ni eller ti ledere, hvor det er parallellkoblet totalt to kurser. Det brukes derfor en korreksjonsfaktor for to nærliggende kurser lik 0.80.

Fra [42] og [43] er det oppgitt at medfølgende kabler til valgte kontakter har henholdsvis tverrsnitt på $0.75mm^2$ for thrusterne, og $0.50mm^2$ for manipulatormotorene. Thrusterne og motorkontrollerne har allerede kabler for å koble sammen, disse kablene har et tverrsnitt på 3.14 mm^2. Det vil videre i dette kapittelet bli beregnet om dette er tilstrekkelig for våres behov.

Fra kontaktene til thrusternes motorkontrollere

Kabelen vil gå fra kontaktene på bakplaten til thrusternes motokontrollere, som er montert på rammeverket til ROV-en. Denne lengden er oppgitt fra sensorgruppen til å være maksimalt 0.30 meter [2]. Her er det fortsatt enfase spenning.

$$A = \frac{2 \cdot 16\ A \cdot 0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30\ m \cdot 1.2 \cdot 1}{12\ V \cdot 0.03} = 0.56\ mm^2 \tag{4.17}$$

Kablene som medfølger kontaktene har et tverrsnitt på $0.75mm^2$, som gir lederevnen:

$$I_{Kabel} = \frac{0.75\ mm^2 \cdot 12\ V \cdot 0.03}{0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30\ m \cdot 1 \cdot 1.2 \cdot 2} = 21.43\ A \tag{4.18}$$

Med de nevnte korreksjonsfaktorene over blir da den korrigerte lederevnen:

$$I_{Kabel,korrigert} = 21.43\ A \cdot 1 \cdot 0.80 = 17.14\ A \tag{4.19}$$

17.14 A lederevne er innenfor kravet for 10 A strømtrekk, og kablene er derfor tilstrekkelige.

Fra motorkontroll til thrusterne

Kabelen vil gå fra thrusterne sine motokontrollere som er montert på rammeverket til ROV-en. Denne lengden er oppgitt fra sensorgruppen til å være maksimalt 0.30 meter [2]. Fra motorkontrollerne vil det gå trefase spenning til thrusterne, og må derfor først regne ut strømmen ved trefase:

$$I_{kabel} = \frac{P}{U \cdot \sqrt{3}} = \frac{192W}{12V \cdot \sqrt{3}} = 9.24\ A \tag{4.20}$$

$$A = \frac{\sqrt{3} \cdot 9.24\ A \cdot 0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30\ m \cdot 1.2 \cdot 1}{12\ V \cdot 0.03} = 0.28\ mm^2 \tag{4.21}$$

0.28 mm^2 er altså minimum tverrsnitt. Motorenes tilhørende kabler har som sagt et tverrsnitt på $3.14mm^2$, som gir lederevnen:

$$I_{Kabel} = \frac{3.14\ mm^2 \cdot 12\ V \cdot 0.03}{0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30\ m \cdot 1 \cdot 1.2 \cdot \sqrt{3}} = 103.59\ A \tag{4.22}$$

Det er her ikke nødvendig med korreksjonsfaktor for nærliggende kurser, da de ligger for seg selv til hver motor. Korreksjonsfaktoren ved 30° er lik 1.00, lederevnen er derfor 103.59 A. Dette betyr at kablene med dette tverrsnittet er mer enn tilstrekkelig.

Fra kontakt til manipulatorens stegmotorer

Kabelen vil gå fra kontakten på bakplaten til manipulatorens motorer. Denne lengden er oppgitt fra sensorgruppen til å være maksimalt 0.30 meter [2].

$$A = \frac{2 \cdot 4.2\ A \cdot 0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30\ m \cdot 1.2 \cdot 1}{12\ V \cdot 0.03} = 0.15\ mm^2 \tag{4.23}$$

$0.15mm^2$ er altså minimum tverrsnitt. Kablene som følger med kontakten har et tverrsnitt på $0.50mm^2$, som gir lederevnen:

$$I_{Kabel} = \frac{0.50\ mm^2 \cdot 12\ V \cdot 0.03}{0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30\ m \cdot 1 \cdot 1.2 \cdot 2} = 14.29\ A \tag{4.24}$$

Med de nevnte korreksjonsfaktorene over blir da den korrigerte lederevnen:

$$I_{Kabel,korrigert} = 14.29\ A \cdot 1 \cdot 0.80 = 11.43\ A \tag{4.25}$$

11.43 A lederevne er innenfor kravet for maksimalt strømtrekk på 4.2 A. Dette betyr at kablene er tilstrekkelige.

Fra kontakt til manipulatorens BLDC-motor

Kabelen vil som forklart over gå fra kontakten på bakplaten til manipulatorens motorer. Denne lengden er oppgitt fra sensorgruppen til å være maksimalt 0.30 meter [2].

$$I_{kabel} = \frac{P}{U \cdot \sqrt{3}} = \frac{144W}{12V \cdot \sqrt{3}} = 6.93\ A \tag{4.26}$$

$$A = \frac{\sqrt{3} \cdot 6.93\ A \cdot 0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30\ m \cdot 1.2 \cdot 1}{12\ V \cdot 0.03} = 0.21\ mm^2 \tag{4.27}$$

0.21 mm^2 er altså minimum tverrsnitt. Kablene som følger med kontakten har et tverrsnitt på $0.50mm^2$, som gir lederevnen:

$$I_{Kabel} = \frac{0.50\ mm^2 \cdot 12\ V \cdot 0.03}{0.0175\ \Omega\frac{mm^2}{m} \cdot 0.30\ m \cdot 1 \cdot 1.2 \cdot \sqrt{3}} = 16.49\ A \tag{4.28}$$

Med de nevnte korreksjonsfaktorene over blir da den korrigerte lederevnen:

$$I_{Kabel,korrigert} = 16.49\ A \cdot 1 \cdot 0.80 = 13.19\ A \tag{4.29}$$

13.19 A lederevne er innenfor kravet for maksimalt strømtrekk på 6.93 A. Dette betyr at kablene er tilstrekkelige.

4.3 Konklusjon

Det ble funnet at det bør være minimum 0.56 mm^2 tverrsnitt på kablene fra mikrokontrolleren til kontaktene som skal til thrusternes motorkontrollere, minimum 0.21 mm^2 tverrsnitt på kablene fra mikrokontrolleren til manipulatorens motorkontrollere, minimum 0.06 mm^2 tverrsnitt på kablene fra motorkontrolleren for stegmotorene og minimum 0.08 mm^2 tverrsnitt på kablene fra motorkontrolleren for BLDC-motor. Med korreksjonsfaktorer for nærføring og temperatur er det dog nødvendig med et større tverrsnitt. Det konkluderes dermed med at internt i ROV-ens elektronikkhus er et tversnitt på 1.00 mm^2 tilstrekkelig for kabler til manipulatorens motorkontrollere, og et tverrsnitt på 2.00 mm^2 tilstrekkelig for kabler til thrusternes motorkontrollere.

I delkapittel 4.1.2 ble det konkludert med valg av kontakter som skal brukes fra bakplaten til de nevnte komponentene. Valget falt på kontakter av typen *SubConn Circular* for thrusternes motorkontrollere, og *Micro Low Profile Series* for manipulatormotorene. Som forklart oppfyller disse kravene om tverrsnitt, [42] [43]. Det spesifiseres at i datablad for disse kontaktene er det oppgitte maksimale strømtrekk på henholdsvis 5 A og 10 A per pinne, som vi har beregnet til 11.43 A, 13.19 A og 17.14 A. Dette store avviket kan grunnes flere ting, som ulike utgangspunkt i temperatur og nærføring, lengde på kabel og lignende. Det konkluderes med at de er tilstrekkelige, da både våres beregninger og det oppgitte maksimale strømtrekk er innenfor kravene.

Kapittel 5

Manipulator

Innhold

5.1 Design av manipulatorarmen 57

5.1.1 Nedre arm 58

5.1.2 Øvre arm 58

5.1.3 Klype 59

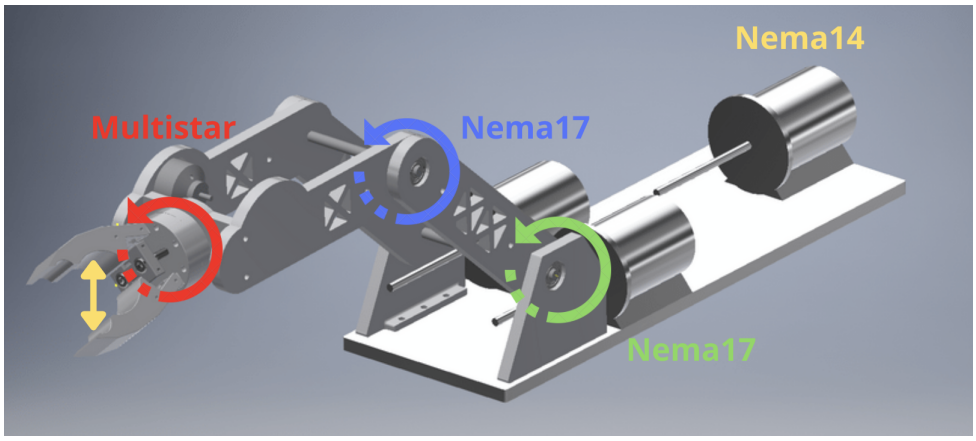
5.1.4 Rotasjon av klypen 59

5.2 Konklusjon 60

I dette kapitellet blir det kort forklart bakgrunn for design og hvilke funksjoner de ulike leddene til manipulatorarmen har, samt hvilken rolle motorene har og hvordan de påvirker de ulike leddene. Det er viktig å ha god forståelse for dette for å videre kunne utvikle et effektivt og brukervennlig styringssystem.

5.1 Design av manipulatorarmen

Manipulatoren er kritisk for at ROV-en skal klare å gjennomføre de fleste oppgavene i MATE-konkurransen. Den er designet av Design and control of ROV manipulators-gruppen [17] i henhold til oppgavene som skal løses i konkurransen. I årets MATE-konkurranse er det tyngste objektet som skal løftes under vann 0.5 kg, som dermed krever en løftekraft på > 5 N. Det er stor variasjon i objektenes utforming, og armen må derfor være i stand til å gripe både korte og lange objekter, samt alle slags former. Med hensyn til dette er manipulatoren designet til å bestå av fire ledd, henholdsvis nedre arm, øvre arm, klype og rotasjon av klypen, som vist i figuren under:



Figur 5.1: CAD-modell av årets manipulatorarm, med markering av ledd og navn på motorene til leddende. Figur hentet fra [17], markeringer og navn innsatt i etterkant.

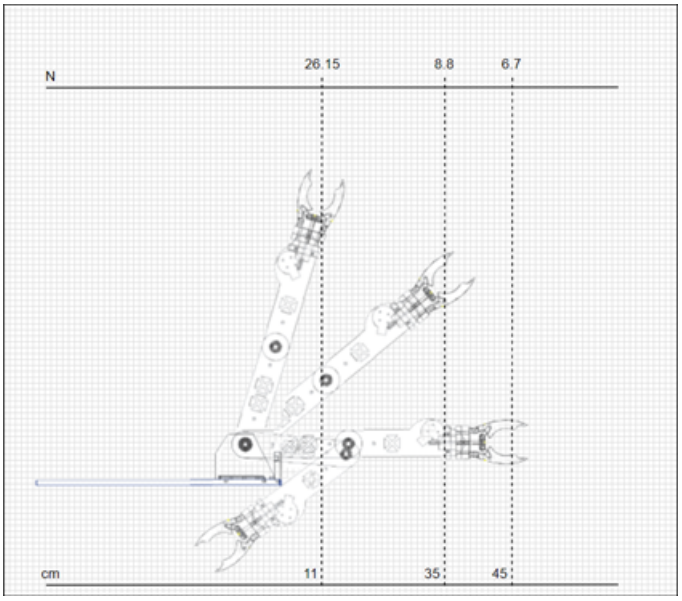
I figur 5.1 er motorene markert med navn og fargekode. Grønn sirkel viser rotasjonspunkt for nedre arm styrt av Nema 17 motor, blå sirkel viser rotasjonspunkt for øvre arm styr av Nema 17 motor, rød sirkel viser rotasjonpunkt for klypen styrt av Multistar Elite motor og gul pil viser klypen styrt av Nema 14 motor.

5.1.1 Nedre arm

Den nedre armen er det nederste leddet som vist med grønn sirkel i figur 5.1. Dette leddet skal være i stand til å kunne beveges opp og ned ved hjelp av stegmotoren Nema 17HS24-2104S [20]. Stegmotoren driver leddet ved hjelp av ormgir[1] med 1:50 girutveksling[2], for å rotere selve leddet. Hovedfunksjonen til nedre arm er å justere avstanden fra ROV til klypen. I figur 5.2 vises det at dersom leddene stilles inn for maksimal lengde, vil en oppnå en effektiv klypdistanse på 45 cm fra armens festepunkt.

5.1.2 Øvre arm

Øvre arm er det midterste leddet, vist med blå sirkel i figur 5.1. Dette leddet blir styrt av den andre stegmotoren av typen Nema 17HS24-2104S [20]. Dette leddet benytter seg også av ormgir av samme type som nedre arm. Hovedfunksjonen til den øvre armen er å posisjonere klypen for å gripe objekter, både over og under selve manipulatorens festepunkt. Som illustrert på figur 5.2 vil en kunne gripe objekter langt under selve manipulatoren dersom en stiller leddene riktig.



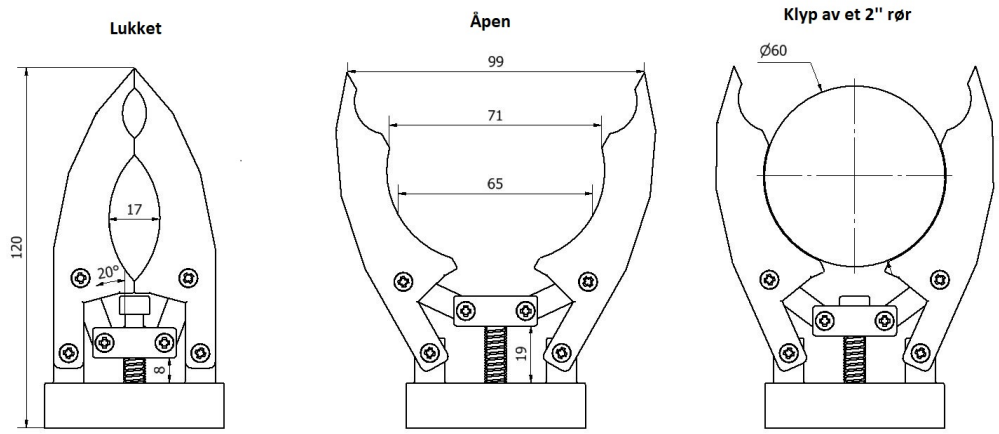
Figur 5.2: Figuren viser en tegning av distansen manipulatoren har mulighet å strekke seg, samt hvordan den kan nå objekter plassert under seg selv. Figur hentet fra [17].

[1] En type sammensetning av ormhjul og ormskrue for å lett kunne overføre bevegelse fra ene siden til andre, uten å tillate bevegelse i andre retningen. På denne måten oppnår en et stort mekanisk holdemoment.

[2] Nedjustering i hastighet for økning i moment. Ved 1:50 girutveksling vil 50 omdreininger på girhjulet fra stegmotoren føre til 1 omdreining på girhjulet til leddet.

5.1.3 Klype

Klypfunksjonen er det viktigste og mest kritiske leddet til manipulatoren. Et godt design for klyp-funksjonen vil være tidsparende ved utførelse av oppgavene i MATE-konkurransen. I figur 5.3 vises målene til klypen i lukket og åpen posisjon, samt et eksempel på når et 2″ rør blir klypt, som er av samme dimensjon som de fleste rørene i MATE-konkurransen.

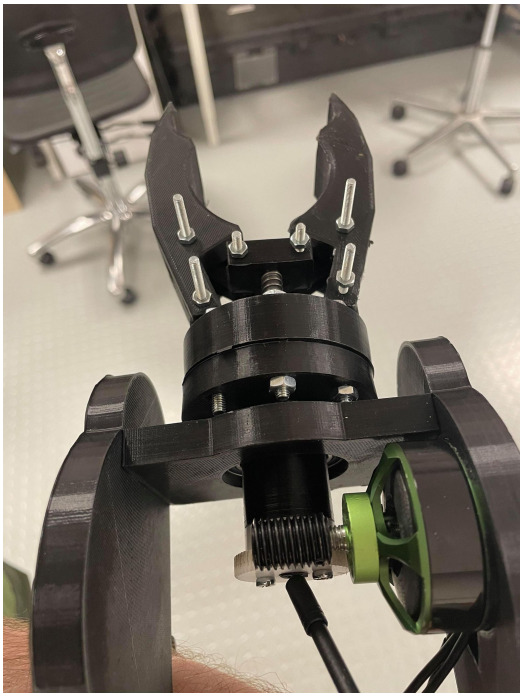


Figur 5.3: Figuren viser målene til klypen i lukket og åpen posisjon. Til høyre i figuren vises et eksempel for bruk av klypen på et 2″ rør.

Klyp-funksjonen blir styrt av stegmotoren Nema 14HS20-1504S. Stegmotoren er koblet til en vaier, som er direkte festet i klypen. Når stegmotoren roterer i klokkeretning trekker den vaieren til seg, som forårsaker at klypen lukker seg, og motsatt retning for at den skal åpne seg. Klypen er designet med fjøringer som sørger for at den holder seg naturlig i åpen posisjon. Dersom klypen griper noe vil holdemomentet til stegmotoren sørge for at den forblir lukket med den vinkelen den griper objektet i. Dette er veldig praktisk med tanke på at når en har klypt et objekt så trenger man ikke å fortsette å gi pådrag for å holde objektet.

5.1.4 Rotasjon av klypen

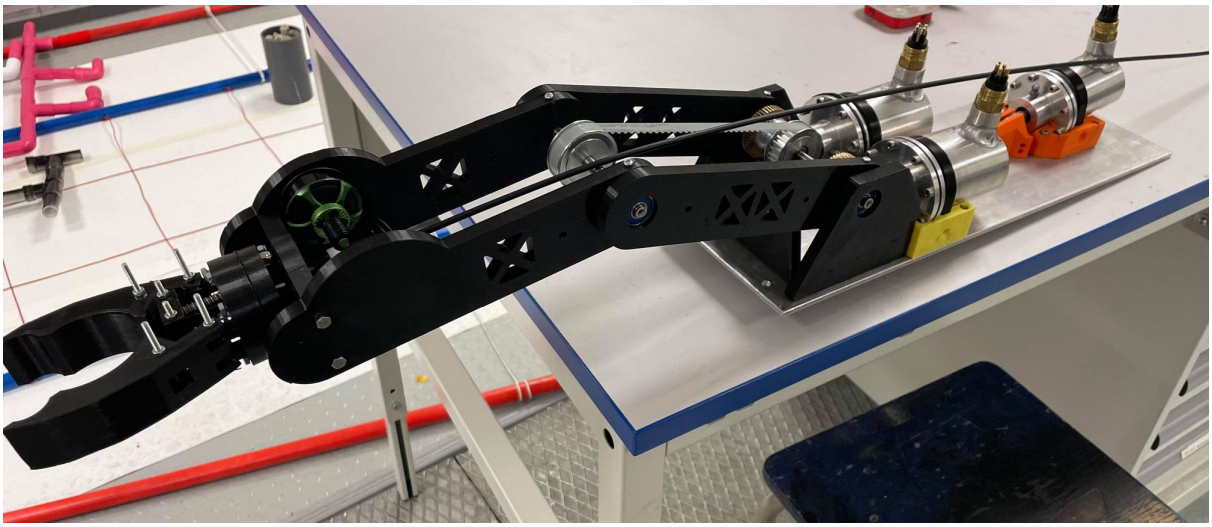
Rotasjon av klypen er en viktig funksjon å ha på en manipulatorarm montert på en ROV. Dette er fordi det gir mulighet til å gripe objekter av alle slags former fra alle slags vinkler, uten å måtte rotere selve ROV-en. Denne rotasjonen utføres ved hjelp av BLDC-motoren Multistar Elite 3508-268KV. Denne motoren har ingen holdemoment, som betyr at dersom en slipper pådraget til motoren vil den ikke klare å holde igjen for motstand. Det er derfor implementert et ormgir med 1:60 girutveksling, som vist på figur 5.4 hvor den grønne motoren er koblet til, som gjør at klypen klarer å holde vinkelen uten at motoren må gi et pådrag, eller ha holdemoment.



Figur 5.4: Bildet viser den grønne BLDC-motoren på høyresiden som er koblet til et ormgir. Bildet viser også klypen sett ovenfra.

5.2 Konklusjon

I figur 5.5 ser en bilde av den ferdigmonterte manipulatorarmen. Designet gir godt utgangspunkt for å kunne utføre de forskjellige oppgavene i MATE-konkurransen, samt at den er allsidig til bruk i andre omgivelser. Alle dens funksjoner har blitt testet og verifisert, og kan leses mer om i kapittel 10. Videre kan det leses mer om motorenes funksjoner og beskrivelse av styresystemet, samt bakgrunn for mangel på posisjonsmåling og/eller endebrytere, i delkapittel 7.5.2.



Figur 5.5: Bilde av den ferdigutviklede og monterte manipulatorarmen.

Kapittel 6

Oppsett av mikrokontroller

Innhold

6.1 Innledning 61
6.2 Konfigurering 62
6.2.1 Timere 62
6.2.2 SysTick 62
6.2.3 Pulsbreddemodulasjon 62
6.2.4 Pulsfrekvensmodulasjon 62
6.2.5 SPI 63
6.3 Generelt oppsett av kode 63
6.3.1 Fil- og programstruktur 63
6.4 Verifisering 68
6.5 Konklusjon 72

6.1 Innledning

Dette kapittelet tar for seg oppsett av mikrokontroller, beskrivelse av systemene brukt til dette, forklaring av mikrokontrollerens funksjoner og begrensninger, samt overordnet oppsett for kommunikasjon med andre systemer i ROV-en. Valg av mikrokontroller ble utført av Kraftoverføring og -fordeling gruppen [35], der vi stilte kravene:

- Mulighet til å generere og sende 12 ulike PWM-signal.
- 3 av PWM-signalene må være tilordnet egne timere, for å kunne endre frekvens individuelt.
- Rask kommunikasjon med andre systemer i ROV, spesielt sensorkort.

Basert på disse kravene, samt krav fra andre grupper som skal benytte seg av samme kort, falt valget på NUCLEO-H7A3ZI-Q med mikrokontrolleren STM32H7A3ZI [49]. Dette oppfyller alle krav om antall timere nødvendig for å generere PWM signaler, samt at den tilbyr SPI-moduler for kommunikasjon med sensorkortet.

6.2 Konfigurering

6.2.1 Timere

En timer (teller) er en liten maskinvare som er tilgjengelig i de fleste typer mikrokontrollere. Funksjonen til en timer er å telle til en satt verdi i en fast hastighet, som videre kan brukes til å utføre forskjellige handlinger. En 16-bits timer kan telle til maksimalt $2^{16} - 1 = 65535$, før den da begynner å telle fra 0 igjen. Denne verdien er dog konfigurerbar, og en kan sette telleverdi til hva enn en ønsker innenfor timerens maksimale verdiområde. Timeren er også hele tiden klar over hvor langt den har telt, og en kan dermed utføre handlinger etter hvor langt den er kommet.

6.2.2 SysTick

SysTick er en timer som allerede er konfigurert i mange mikrokontrollere, inkludert STM32H7A3ZI. Dens formål er i hovedsak å lage nøyaktige avbrudd til forskjellige oppgaver. Som oftest er den satt til å telle opp til 1000 i løpet av 1 sekund, altså med en tellefrekvens på 1000 Hz. Ved å lage variabler som øker med 1 for hver SysTick kan en dermed aktivere disse som flagg etter et valgt antall millisekunder. Dette vises i den første kodesnutten under, der det er satt opp tellere som aktiverer tre flagg etter en bestemt tid, for så å nullstille tellerene. Videre blir disse flaggene brukt i neste kodesnutt for å utføre diverse handlinger, samt at flaggene blir nullstilt, slik at handlingene ikke utføres igjen før neste gang tellebetingelsen er oppfylt.

6.2.3 Pulsbreddemodulasjon

En av de mest vanlige bruksområdene til en timer er pulsbreddemodulasjon (PWM). PWM går ut på å modulere et pulssignal ved å justere hvor lenge signalet er aktivt per periode, mens periodetiden forblir uendret. Forholdet mellom disse kalles en arbeidssyklus (duty-cycle), og blir ofte oppgitt i prosent. Ved å integrere denne pulsen får man en spenning som tilsvarer pulslengdens aktive prosentandel, og kan måles som effektiv-verdien av signalet. Denne typen signaler blir brukt for å styre de fleste typer børsteløse likestrømsmotorer, inkludert de som blir brukt på årets ROV. Mer om oppsett og implementering av dette kan leses om i kapittel 7.

6.2.4 Pulsfrekvensmodulasjon

Pulsfrekvensmodulasjon (PFM) er på mange måter det samme som pulsbreddemodulasjon, ved at en bruker timerens telleverdi til å justere et puls-signal. Forskjellen er i hovedsak at ved pulsfrekvensmodulasjon modulerer en frekvensen til signalet, og har ofte en fast arbeidssyklus på 50%. Måten en endrer frekvensen på er ved å justere telleverdien til timeren, som direkte justerer hvor ofte telleverdien og signalet nullstilles, som dermed påvirker signalets periodetid. Stegmotorer blir i hovedsak kontrollert ved at en puls tilsvarer et steg, og for å justere hastigheten er det da vanlig å ta i bruk pulsfrekvensmodulasjon. En kan da fortløpende endre stegmotorens hastighet, samt at en kan oppnå høye frekvenser uten å måtte bruke mye prosessorkraft.

6.2.5 SPI

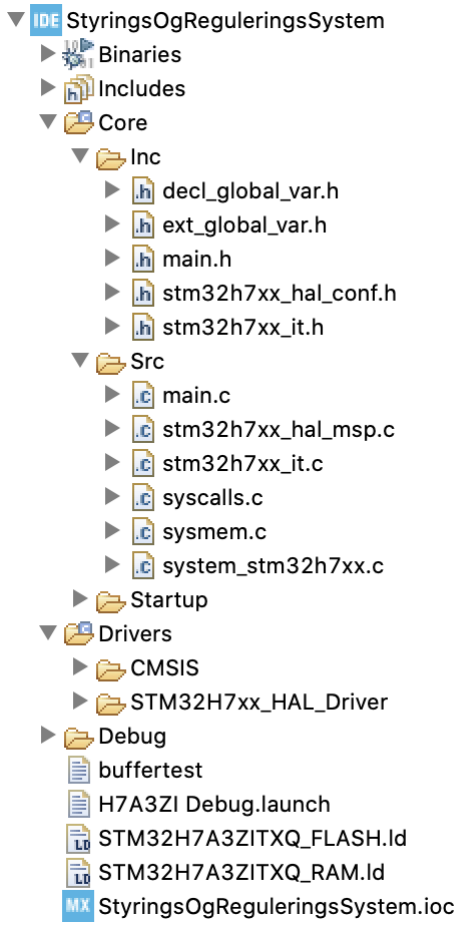
Serielt perifert grensesnitt (SPI) er en synkron seriekommunikasjonsgrensesnittspesifikasjon som brukes for kortdistansekommunikasjon. For mer detaljert forklaring om SPI, se sensorgruppens rapport [2].

6.3 Generelt oppsett av kode

Overordnet oppsett av kode for styring, regulering og henting av sensordata, blir forklart under. Videre blir den implementerte koden for styring og regulering utledet og forklart i egne kapitler, henholdsvis 7 og 9.

6.3.1 Fil- og programstruktur

Når en oppretter et prosjekt i utviklingsverktøyet STM32CubeIDE får en automatisk initialisert alle nødvendige filer og mapper. I tillegg til disse blir det lagt inn noen egendefinerte filer, som blir forklart senere i kapittelet. Filstrukturen for styrings- og reguleringssystemet ser dermed ut som følger:

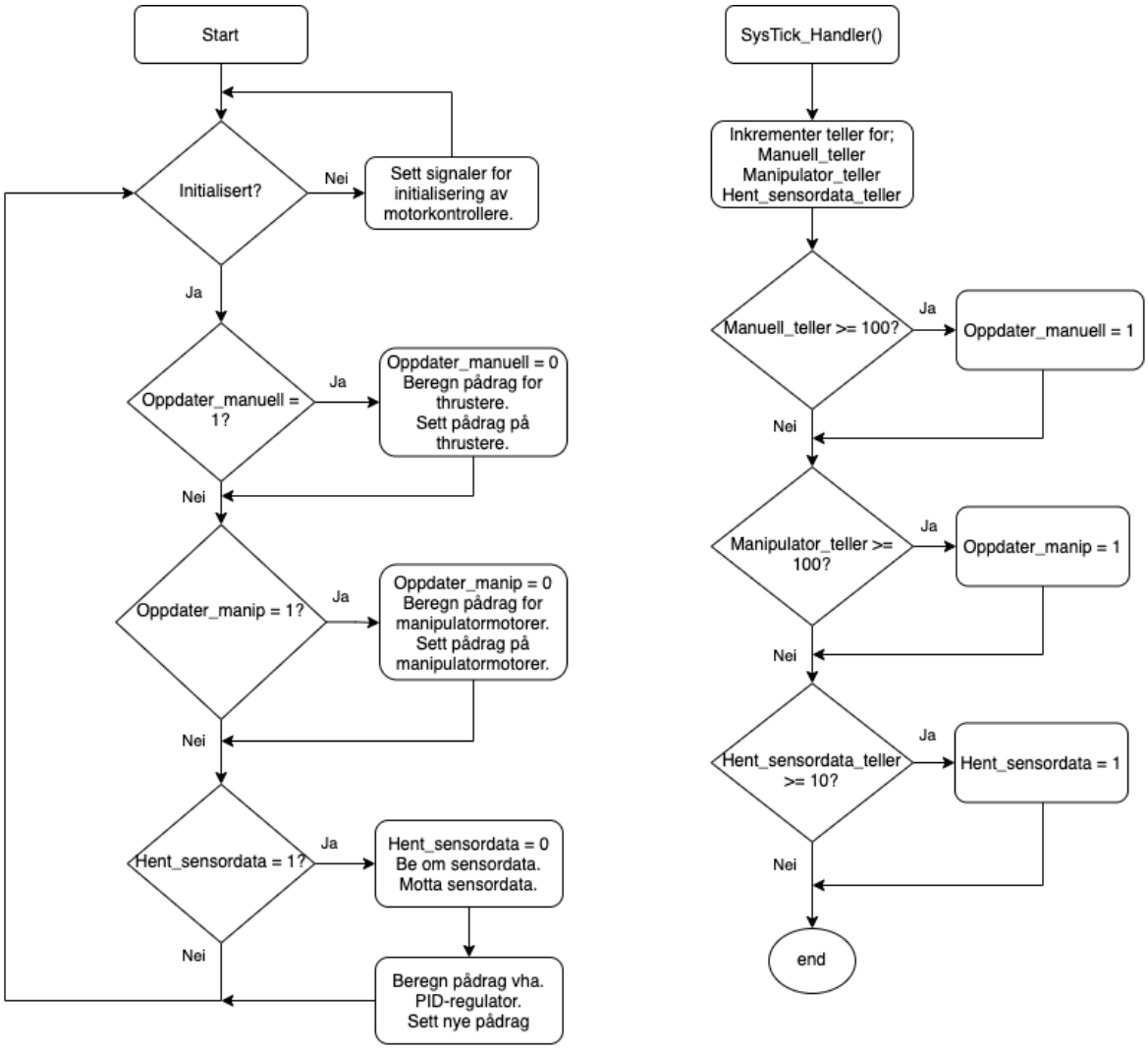


Figur 6.1: Filstruktur for styrings- og reguleringssystemet. Filene **decl_global_var.h** og **ext_global_var.h** er egendefinerte, resten er automatisk iniatalisert.

StyringsOgReguleringssystem.ioc-filen (nederst) er konfigureringsfil for pinner

og moduler. Denne blir brukt og forklart videre i delkapittel 7.6.

Videre er det hovedsaklig filene **main.c** og **stm32h7xx_it.c** som blir brukt for initialiseringer, beregninger, avbruddshåndteringer og setting av signaler. Videre i rapporten vil alle kodesnutter bli markert med navn på hvilke fil koden er implementert i. Under vises overordnet flytskjema for programstrukturen som videre blir forklart i resten av kapittelet:



Figur 6.2: Overordnet flytdiagram for programstrukturen for styrings- og reguleringssystemene.

I første omgang blir det tatt hensyn til hvor ofte en ønsker å utføre de forskjellige beregningene, og sette nye verdier. Dette løses ved tellere i **SysTick_Handler**, som forklart er en innebygd funksjon for å oppnå tidspresise avbrudd. Se kodesnutt:

filnavn: stm32h7xx_it.c

```
1 void SysTick_Handler(void)
2 {
3   /* USER CODE BEGIN SysTick_IRQn 0 */
4
5   /* USER CODE END SysTick_IRQn 0 */
```

```
6   // Forhaandsdefinert funksjon som er nodvendig for at HAL
     modulen skal fungere.
7   HAL_IncTick();
8   /* USER CODE BEGIN SysTick_IRQn 1 */
9
10  // Initialisering
11  if (Initialisert_teller >= 5000) { // Gaar hoy etter 5
     sekunder. Denne blir brukt videre i neste kodesnutt.
12      Initialisert = 1;
13  } else {
14      Initialisert_teller ++;
15  }
16
17  // Manuell styring
18  Oppdater_manuell_teller ++;
19  if (Oppdater_manuell_teller >= 100) { // Gaar hoy 10 ganger i
     sekundet, og setter "Oppdater_manuell" hoy hver gang. Denne
     blir brukt videre i neste kodesnutt.
20      Oppdater_manuell_teller = 0;
21      Oppdater_manuell = 1;
22  }
23
24  // Manipulatorstyring
25  Oppdater_manipulator_teller ++;
26  if (Oppdater_manipulator_teller >= 100) { // Gaar hoy 10
     ganger i sekundet, og setter "Oppdater_manipulator" hoy hver
     gang. Denne blir brukt videre i neste kodesnutt.
27      Oppdater_manipulator_teller = 0;
28      Oppdater_manipulator = 1;
29  }
30
31  // Hent sensor. Oppdaterer PID paa samme flagg.
32  Hent_sensordata_teller ++;
33  if (Hent_sensordata_teller >= 10) { // Gaar hoy 100 ganger i
     sekundet, og setter "Hent_sensordata" hoy hver gang. Denne
     blir brukt videre i neste kodesnutt.
34      Hent_sensordata_teller = 0;
35      Hent_sensordata = 1;
36  }
37
38  /* USER CODE END SysTick_IRQn 1 */
39 }
```

Videre blir dette brukt til å utføre den tilhørende kode med jevne mellomrom. Se kode:

filnavn: main.c

```
1  while (1)
2  {
3      // Sjekk om "initialisert" er 1, dvs. om det er gaatt mer
      enn 5 sekunder siden oppstart.
4      if (Initialisert) {
5          // Sjekk om flagg for henting av sensordata er 1
6          if (Hent_sensordata) {
7              Hent_sensordata = 0;
8
```

```
9             // Send en forhaandsbestemt verdi tx = ('T') til sensorkortet for aa informere om at vi onsker ny data
10            HAL_SPI_Transmit(&hspi3, tx, 1, HAL_MAX_DELAY); // NB! HAL_MAX_DELAY skal ikke bli brukt slik, da det kan fore til store forsinkelser i hele programmet dersom SPI-kommunikasjon skulle bli forstyrret. Denne blir satt til en gunstig verdi ved senere testing.
11            // Motta ny data fra sensorkortet og last inn i rx_data. Sensorsystemet er satt opp til aa kontinuerlig fylle en global liste med nye sensordata. Med en gang de mottar signalet fra oss om onsket for nye maalinger, blir det aktivert et avbrudd som sender dette. Det kommer derfor tilnaermet umiddelbart. Dataene er ogsaa da siste maalte verdi.
12            HAL_SPI_Receive(&hspi3, (uint8_t *)&rx_data, sizeof(std_reply_t), HAL_MAX_DELAY);
13
14            // Kode for PID-regulering basert paa nye sensordata. Se kapittel 9 i rapport for kode og forklaring av dette.
15        }
16
17        // Sjekk om flagg for oppdatering av manuell styring er 1
18        if (Oppdater_manuell) {
19            Oppdater_manuell = 0;
20
21            // Kode for manuell kjoring. Se kapittel 7 i rapport for kode og forklaring av dette.
22        }
23
24        // Sjekk om flagg for oppdatering av manipulatorstyring er 1
25        if (Oppdater_manipulator) {
26            Oppdater_manipulator = 0;
27
28            // Kode for styring av manipulator. Se kapittel 7 i rapport for kode og forklaring av dette.
29        }
30    } else {
31        // Kode for initialisering av nodvendige signaler. Se kapittel 7 i rapport for kode og forklaring av dette.
32    }
33 }
```

rx_data, som responsen med sensordata blir plassert i (linje 12 over), er en struktur for forhåndsbestemt oppsett av arrayet sensorkortet sender. Dette er et array med lengde på 19 byte hvor hver byte er angitt en fast variabel, se figur 6.3. Strukturen og dens initialisering er dermed satt opp som følger:

filnavn: main.c

```
1 #pragma pack(push,1)
2 typedef struct{
3     int16_t rull;
4     int16_t stamp;
5     int16_t gir;
```

```
6
7       int16_t temp1;
8       uint32_t dybde;
9       int16_t temp2;
10      uint16_t avstand;
11      uint8_t konfidens;
12      uint8_t lekkasje;
13      uint8_t feil;
14  } std_reply_t;
15  #pragma pack(pop)
16
17  ...
18
19  int main(void) {
20      ...
21      std_reply_t rx_data = {0};
22      ...
23  }
```

Beskrivelse	Min verdi	Maks verdi	Polaritet	Oppløsning (LSB)	Antall byte til overføring	Indeks
IMU Rull-vinkel	-180	180	U	0,01 °	2	0-1
Stamp-vinkel	-180	180	U	0,01 °	2	2-3
Gir-vinkel	-180	180	U	0,01 °	2	4-5
Temperatur	...	...	U	0,01 °C	2	6-7
Trykksensor Dybde	0	100 000	U	1 mm	4	8-11
Temperatur	...	...	U	0,01 °C	2	12-13
Altimeter Avstand	0	30 000	P	1 mm	2	14-15
Konfidensintervall	0	100	P	1 %	1	16
Lekkasje	0 (ikke lekkasje)	1 (lekkasje)	P		1	17
Feilmelding			P		1	18
Totalt					19	

Figur 6.3: Tabell av innhold i array sendt fra sensorkort, gitt av sensorgruppen [2]. Tabellen viser minimum og maksimum verdi til de ulike dataene, samt hvilken indeks de har i arrayet. Med **indeks** menes plasseringen til den gitte verdien i arrayet, for eksempel vil **Rull-vinkel** hentes fra plass nummer 0 og 1.

Hvert av bytene i kodesnuttet blir forklart i tabellen i figur 6.3 over, med lik rekkefølge. For eksempel ser en at **Rull-vinkel** har en verdi fra -180 til 180, og får derfor tildelt typen **int** som kan være både positive og negative verdier. Videre har den en størrelse på 2 byte til tilsvarer $2 \cdot 8 = 16$ bits, og har derfor fått tildelt størrelsen int**16**. Til slutt er den bestemt til å bli sendt med indeks 0-1, og er derfor plassert først i strukturen i kodesnutten over.

6.4 Verifisering

Før dette blir fastsatt og man starter på implementasjon av videre logikk, er det nødvendig å verifisere at det fungerer som det skal. For å verifisere **SysTick_Handler** metoden, som skal sørge for å kunne utføre handlinger og beregninger med jevne og selvvalgte mellomrom, settes det opp en test.

Dette gjøres ved å kun ha en enkelt teller og et enkelt flag, i dette tilfellet **Test_teller** og **Test_systick**, samt å utføre en enkel, målbar handling i hovedløkken, i dette tilfellet å sette en GPIO-pinne høy/lav for hver iterasjon. Det ser dermed ut som vist her:

filnavn: stm32h7xx_it.c

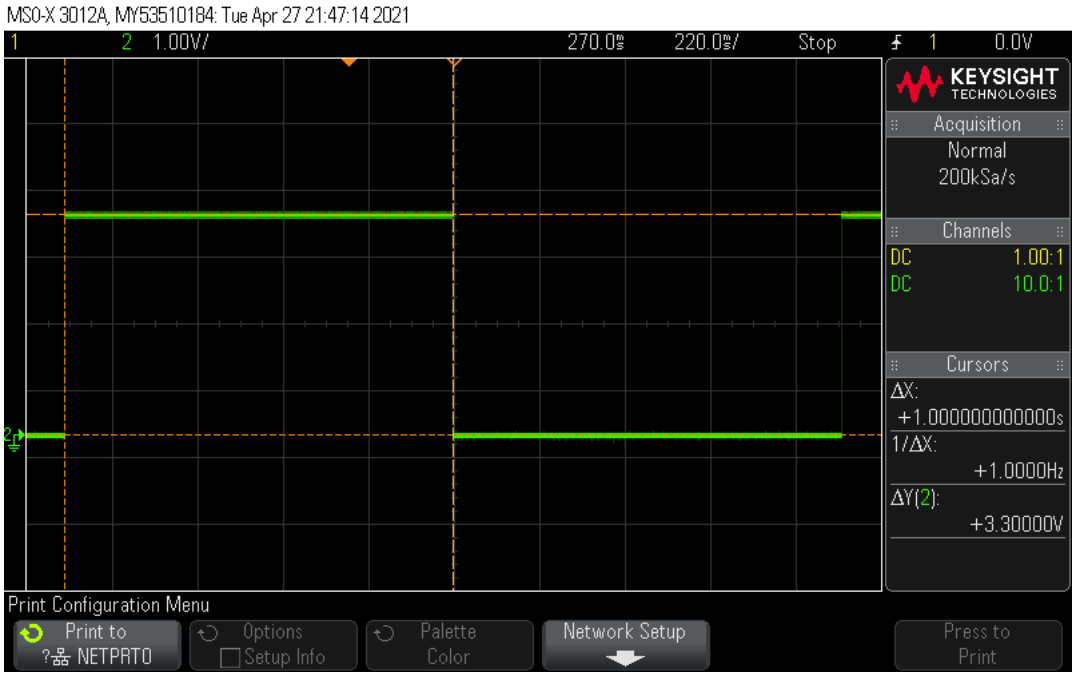
```
1  void SysTick_Handler(void)
2  {
3    /* USER CODE BEGIN SysTick_IRQn 0 */
4
5    /* USER CODE END SysTick_IRQn 0 */
6    // Forhaandsdefinert funksjon som er nodvendig for at HAL
       modulen skal fungere.
7    HAL_IncTick();
8    /* USER CODE BEGIN SysTick_IRQn 1 */
9
10   // Testoppsett. Test_teller intererer for hver SysTick, og en
       if-betingelse blir oppfylt naar den har naadd 1000.
11   Test_teller ++;
12   if (Test_teller >= 1000) { // Gir en oppdatering per sekund.
13     Test_teller = 0;
14     // Setter Test_systick hoy.
15     Test_systick = 1;
16   }
17   /* USER CODE END SysTick_IRQn 1 */
18 }
```

filnavn: main.c

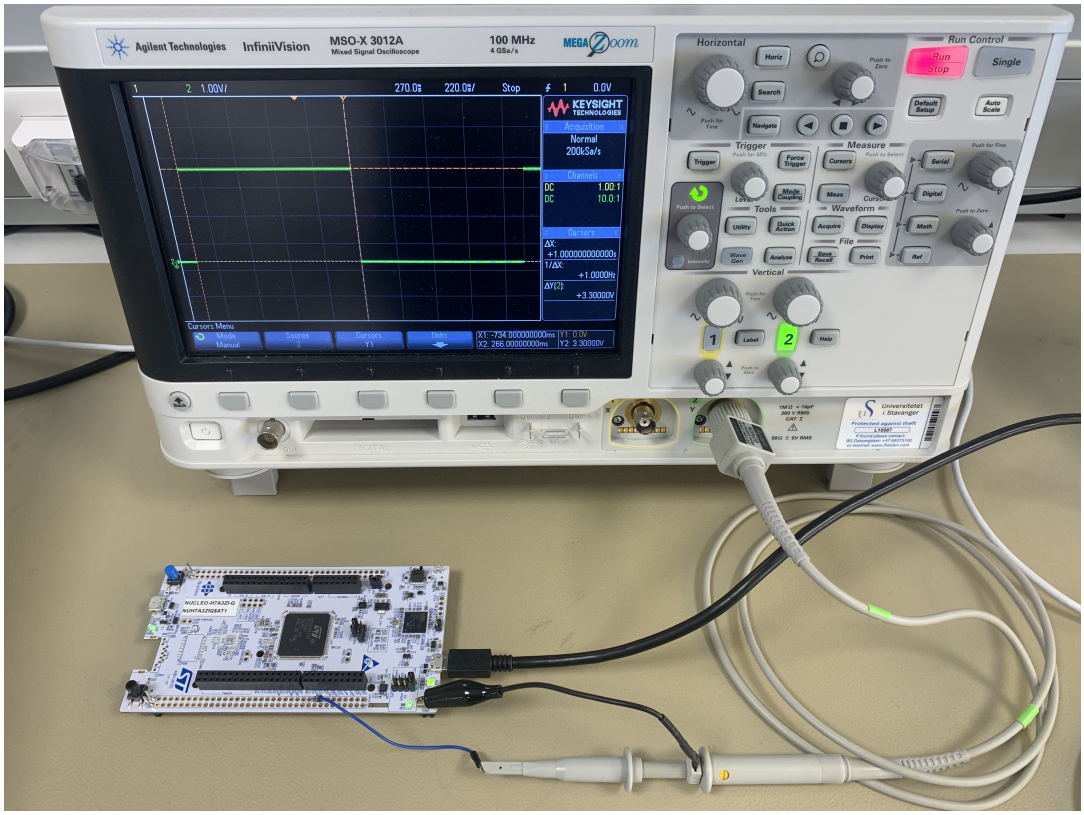
```
1  while (1)
2  {
3      /* USER CODE END WHILE */
4
5      /* USER CODE BEGIN 3 */
6
7      // Oppfyller en if-betingelse naar Test_systick fra
       kodesnutt over er oppfylt.
8      if (Test_systick) {
9          Test_systick = 0;
10         // Sett en GPIO-pinne hoy/lav for hver iterasjon, dvs.
       hvert sekund.
11         HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_6);
12     } else {
13         // Initialiser som lav.
14         HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, GPIO_PIN_RESET);
15     }
16 }
```

Funksjonen **SysTick_Handler** og løkken **while (1)** er i to forskjellige filer, noe som må tas hensyn til slik at oppdatering av en variabel blir reflektert i den andre. Dette løses ved å lage to nye filer dedikert for å definere globale variabler, for så å inkludere de der det trengs. Filene får navn **decl_global_var.h** og **ext_global_var.h**, og inneholder alle nødvendige globale variabler. Variablene i **ext_..** filen inneholder en **extern** forran selve defineringen, for å unngå dobbeldefinering ved importering til hovedfilen **main.c**, og alle andre **.c**-filer en tar i bruk.

GPIO-pinne **nr. 6** på perifermodul **B** som en i denne testen setter høy/lav, kan måles på pinne **PB6**. Ved å måle denne pinnen ved hjelp av et oscilloskop får en følgende resultat:



Figur 6.4: Skopbilde av måling av GPIOB, pinne nr. 6 (PB6). En ser at signalet er høyt i 1.00 sekunder (ΔX til høyre i skopbildet), deretter lavt, og igjen høyt. Dette betyr at testen var vellykket.



Figur 6.5: Oppsett for test av kodeoppsett. Kortet NUCLEO-H7A3ZI-Q sees som hvitt kretskort nede til venstre, og måles av et oscilloskop av typen MSO-X 3012A. Proben måler over pinne PB6 med blå kabel og er koblet til kortets jord med svart klype. Kortet et koblet til PC med svart kabel av typen mikro-USB.

Oppsett og modulering, samt beregning av ønskede arbeidssykluser og frekvenser for pulsbredde- og pulsfrekvenssignaler, er detaljert beskrevet i kapittel 7.6. For ryddighetens skyld blir verifiseringen av at koden for styring av disse signalene fungerer, vist her. Oppsett for testen er lik som over, med nytt innhold i **if**-betingelsen. For test av pulsbreddemodulering blir følgende kode brukt:

filnavn: main.c

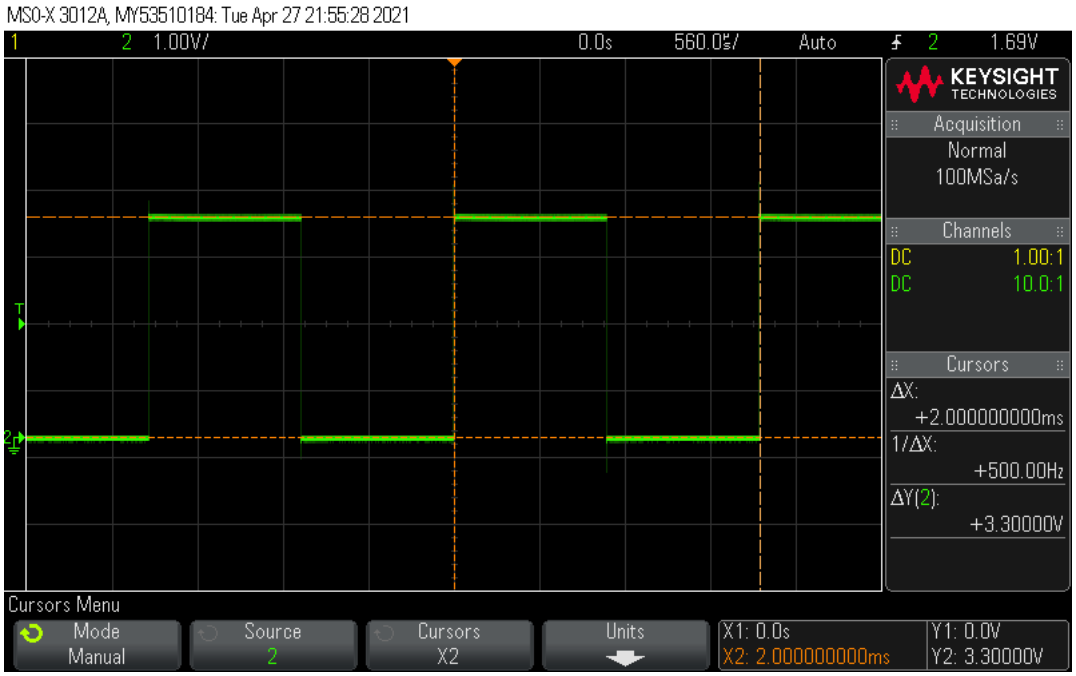
```
1 // Test_sytick gaar hoy etter 1 sekund, som forklart i starten
     av kapittel 6.4 i rapport. Dvs. at om testen er vellykket
     skal signalet beskrevet under skrues paa 1 sekund etter
     oppstart.
2 if (Test_systick) {
3     Test_systick = 0;
4
5     TIM1->CCR1 = standby; // Standby er satt til 15000 som vil
     gi 1.5ms paa-tid. ARR er satt til 50000 som vil gi en periode
      paa 5ms. Forklaring av dette er beskrevet i kapittel 7 i
     rapport.
6 } else {
7     TIM1->CCR1 = 0; // Initialiser med 0% arbeidssyklus.
8 }
```

Figur 6.6: Skopbilde av måling av PWM-signal på pinne PE9. I øverste figur ser en at signalet har en på-tid på 1.500ms (ΔX til høyre i skopbildet). Med en periodetid på 5.000ms som vist i nederste figur, gir dette $\frac{1.500}{5.000} \cdot 100\% = 30\%$ arbeidssyklus. Denne ble endret til nye ønskede verdier ved å endre TIM->CCR1 som vist i kode over.

Videre ble det satt opp en test for pulsfrekvensmodulering, som følger:
filnavn: main.c

```
1 // Test_sytick gaar hoy etter 1 sekund, som forklart i starten
    av kapittel 6.4 i rapport. Dvs. at om testen er vellykket
    skal signalet beskrevet under skrues paa 1 sekund etter
    oppstart.
2 if (Test_systick) {
3     Test_systick = 0;
```

```
4
5       TIM1->ARR = 20000; // Gir en frekvens paa 500 Hz.
6       TIM1->CCR1 = 10000; // 20000/2. Gir en arbeidssyklus paa
        50%.
7   } else {
8       // Initialiser med lavest mulig frekvens og 0% arbeidssyklus
        .
9       TIM1->ARR = 65535;
10      TIM1->CCR1 = 0;
```



Figur 6.7: Skopbilde av måling av PFM-signal på pinne PE9. Timeren er initialisert på samme måte som over, men ser at signalet har fått en ny frekvens på 500 Hz. ($\frac{1}{\Delta X}$ til høyre i skopbildet).

6.5 Konklusjon

Verifisering av at sensordata ble sendt og mottatt ble gjennomført av Sensorgruppen [2], og var vellykket. Dette kan leses mer om i deres rapport. Dermed er alle deler av hovedoppsett for mikrokontroller verifisert, og en kan konkludere med at dette er en god struktur for videreutvikling av styrings- og reguleringsprogrammet. Det må nevnes at gruppen er klar over at dette oppsettet kunne på mange måter vært bedre, for eksempel ved å sette opp funksjoner for hver handling istedenfor å ha if-betingelser i hovedløkken. Gruppen har derimot lite tidligere erfaring med programmering i programmeringsspråket C, og det ble derfor prioritert å i første omgang lage et program som fungerer godt til våres behov innen rimelig tid. Videreutvikling og forbedring av strukturen er høyt prioritert for gruppen etter rapportens

innlevering.

Kapittel 7

Styringssystem

Innhold

7.1 Innledning 74
7.2 ROV-ens posisjon og dens bevegelser i vann 74
7.3 Utregning av motorpådrag 75
7.4 Oppsett av styrestikke 78
7.5 Implementering av styredata 79
7.5.1 Styring av thrustere 85
7.5.2 Styring av manipulator 86
7.6 Implementering på mikrokontroller 87
7.6.1 Styring av thrustere 91
7.6.2 Styring av manipulator 94

7.1 Innledning

Dette kapittelet tar for seg styresystemets oppbygging, hvordan det brukes og hvilke funksjoner det har. Det ble valgt å bruke to styrestikker: en for å styre ROV, og en annen for å styre manipulator samt mikro-ROV. Videre ble det bestemt å bruke Xbox360-kontrollere som styrestikker, da det allerede finnes gode bibliotek i programmeringsspråkene Python og JavaScript for innlesing av kontrollsignaler fra denne, samt at det tilbyr et enkelt brukergrensesnitt for piloten.

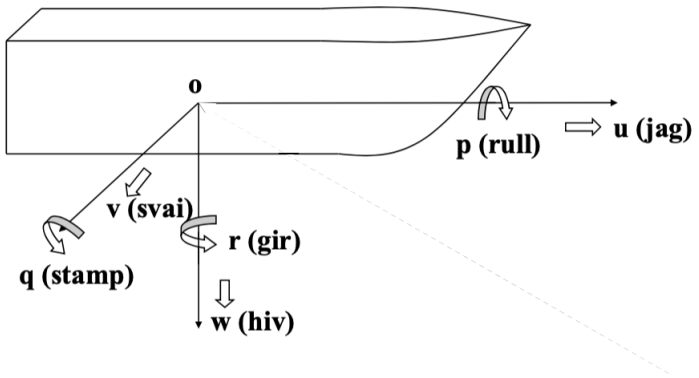
7.2 ROV-ens posisjon og dens bevegelser i vann

For å lage et styringssystem er det er viktig å definere et koordinatsystem for fartøyet. ROV-en har seks frihetsgrader, henholdsvis jag, svai, hiv, rull, stamp og gir, se figur 7.1. Med utgangspunkt i Thor I. Fossen [50] er de tre første koordinatene posisjon langs x-, y- og z-aksene, og de tre siste orientering om disse. Deres tidsderiverte representerer henholdsvis ROV-ens bevegelse langs aksene og ROV-ens rotasjonsbevegelser.

Dette koordinatsystemet trenger et referansesystem som det kan måles mot. Ifølge Thor I. Fossen [50] er det mest vanlig å bruke et North-East-Down (NED) koordinatsystem som referansesystem. Dette er et system som baseres på jordens geodetiske referansesystem, som gir en nøyaktig plassering av legemet og dets orientering i forhold til jordoverflatens ekte nord, øst og vinkelrett nedover. Nøyaktig plassering

oppnås ved å bruke to vinkler, ι og μ, som betegner henholdsvis lengdegrad og breddegrad i forhold til e-rammen[1].

Et Body-Fixed referansesystem (BFF) er derimot et bevegelig koordinatsystem som er fiksert på fartøyet. Dette systemet baseres altså ikke på omverden, og gir kun informasjon om fartøyets posisjon og orientering relativt til sitt eget massesenter i et forhåndsbestemt nullpunkt. Dette nullpunktet settes vanligvis når fartøyet er horisontalt i rull og stamp retning, og en gitt gir retning, ref. figur 7.1.



Figur 7.1: Bevegelses- og posisjonsvariabler for et fartøy i BFF. Hentet fra [51].

Hvilke av disse systemene som brukes baseres på hvilke frihetsgrader en skal regulere. Det skal for årets ROV kun reguleres i hiv, rull og stamp, som blir begrunnet mer detaljert i delkapittel 8.1. En har derfor ingen behov for å vite ROV-ens plassering og orientering i forhold til jordoverflatens ekte nord, øst og vinkelrett ned (NED-system). Dette begrunnes med at hiv baseres på målinger referert direkte til fartøyets dybde, uavhengig av jordoverflatens ekte nord, øst og vinkelrett ned, og rull/stamp baseres på målinger referert direkte til fartøyets rotasjon om x- og y-aksen. Sistnevnte tilsvarer vinkelrett ned i NED-systemet, men kan enkelt refereres til BFF-systemet ved bruk av gyroskop- og akselerometersensorer. Det konkluderes dermed med at vi kun trenger å bruke BFF, da alle disse refereres direkte til fartøyet.

7.3 Utregning av motorpådrag

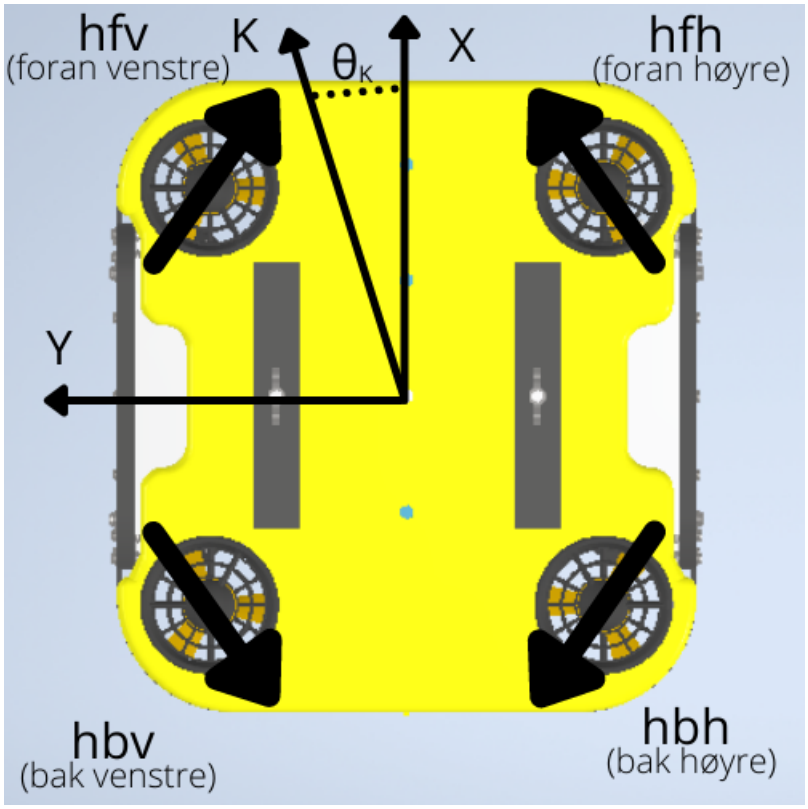
Dette delkapittelet tar for seg utregning av motorpådrag for horisontal translatorisk bevegelse. Videre blir utregning av motorpådrag for hiv- og gir-retning beskrevet i delkapittel 7.5.1.

Ved hjelp av Morten Tengesdals skriv om utregning av motorpådrag for en ROV [52] kan motorpådraget til hver av de fire horisontale motorene kalkuleres, basert på deres posisjon og vinkling på ROV-en. Vilkårene for denne måten å beregne motorpådrag på er:

[1] E-rammen er et jordsentrert og jordfast referansesystem som har sitt nullpunkt i midten av jorden.

- Forholdet mellom utslagene i X_s- og Y_s-retning på styrestikken gir kjøreretning
- Det samlede pådraget i kjøreretningen skal være gitt av utslaget til styrestikken
- Det skal ikke være pådragskomponent normalt på kjøreretningen

Disse beregningene gir kun grunnlag for translatorisk, horisontal bevegelse. Årets ROV har et motoroppsett som vist i figur 7.2.



Figur 7.2: Motoroppsett i horisontal retning for UiS Subsea 2021 ROV. Modellen er sett fra oversiden, horisontale motorer er indikert med tykke svarte piler, og må ikke forveksles med de vertikale motorene som vises på bildet. En gitt kjøreretning K er vist med en vinkel θ_K i forhold til X-aksen.

Et positivt pådrag P på en motor vil gi en kraftretning som vist med tykke, svarte piler i figur 7.2. Vinklingen til denne kraftretningen referert til X-retningen i ROV-ens faste koordinatsystem, blir for de fire motorene:

- hfv: θ_1 = -35°
- hfh: θ_2 = +35°
- hbh: θ_3 = +145°
- hbv: θ_4 = -145°

Videre er følgende tilleggsvilkår satt, som vil gjelde for alle utregninger av pådrag for horisontal translatorisk bevegelse:

- hfv og hbh har samme absoluttpådrag med motsatt fortegn, altså $P_{hfv} = -P_{hbh}$
- hfh og hbv har samme absoluttpådrag med motsatt fortegn, altså $P_{hfh} = -P_{hbv}$

Pådraget i kjøreretningen P_K og det samlede pådraget normalt på kjøreretningen P_N er gitt ved følgende ligninger:

$$\begin{aligned} 1 &: P_K = P_{hfv}\cos(\theta_1 - \theta_K) + P_{hfh}\cos(\theta_2 - \theta_K) + P_{hbh}\cos(\theta_3 - \theta_K) + P_{hbv}\cos(\theta_4 - \theta_K) \\ 2 &: P_N = P_{hfv}\sin(\theta_1 - \theta_K) + P_{hfh}\sin(\theta_2 - \theta_K) + P_{hbh}\sin(\theta_3 - \theta_K) + P_{hbv}\sin(\theta_4 - \theta_K) \end{aligned} \tag{7.1}$$

Ved å følge fremgangsmåten i [52] med årets ROV sitt motoroppsett kan ligning 7.1 skrives om slik, ved bruk av cosinussetningen:

$$\begin{aligned} P_K &= P_{hfv}(c\theta_1 c\theta_K + s\theta_1 s\theta_K) + P_{hfh}(c\theta_2 c\theta_K + s\theta_2 s\theta_K) + \\ &\quad P_{hbh}(c\theta_3 c\theta_K + s\theta_3 s\theta_K) + P_{hbv}(c\theta_4 c\theta_K + s\theta_4 s\theta_K) \\ &= P_{hfv}(0.8191 c\theta_K - 0.5735 s\theta_K) + P_{hfh}(0.8191 c\theta_K + 5735 s\theta_K) + \\ &\quad P_{hbh}(-0.8191 c\theta_K + 0.5735 s\theta_K) + P_{hbv}(-0.8191 c\theta_K - 0.5735 s\theta_K) \\ &= (P_{hfv} + P_{hfh} - P_{hbh} - P_{hbv}) \cdot 0.8191 c\theta_K + (-P_{hfv} + P_{hfh} + P_{hbh} - P_{hbv}) \cdot 0.5735 s\theta_K \\ &= (P_{hfv} - P_{hbh}) \cdot (0.8191 c\theta_K - 0.5735 s\theta_K) + (P_{hfh} - P_{hbv}) \cdot (0.8191 c\theta_K + 0.5735 s\theta_K) \end{aligned} \tag{7.2}$$

hvor:
- $c\theta = \cos(\theta)$
- $s\theta = \sin(\theta)$

Det vises dermed, ved vilkårene satt tidligere i kapittelet, samt resultatet fra ligning 7.2, at motorpådragene for horisontal, translatorisk bevegelse blir som følger:

$$\begin{aligned} P_{hfv} &= P_K \cdot (0.8191 \cdot \cos(\theta_K) - 0.5735 \cdot \sin(\theta_K)) \\ P_{hfh} &= P_K \cdot (-0.8191 \cdot \cos(\theta_K) - 0.5735 \cdot \sin(\theta_K)) \\ P_{hbh} &= -P_{hfv} \\ P_{hbv} &= -P_{hfh} \end{aligned} \tag{7.3}$$

7.4 Oppsett av styrestikke

Xbox360-kontrolleren tilbyr 2 styrespaker med fri bevegelse i to akser, samt 2 to spaker med gradvis inntrykk og 12 av/på knapper.

For fremdrift av ROV skal styrestikken hovedsaklig styre horisontal translatorisk bevegelse med venstre styrespake, og vertikal bevegelse samt gir-rotasjon med høyre styrespake. For å kunne ta utnytte av motorenes maksimale kapasitet samtidig som at piloten skal slippe å måtte være ekstremt nøyaktig med styrestikken i lave hastigheter, vil det være mulig å justere maksimalt utslag fra 10 % til 100 % ved hjelp av **OPP** og **NED** knappene på retningsputen. I tillegg vil diverse knapper aktivere/deaktivere forskjellige moduser og funksjoner. Dette er lett konfigurerbart, og vil bli eksperimentert med og fastsatt når ROV-en er klar til testing i vann.

Figur 7.3: Figuren viser styrestikken med beskrivelse av styreprotokollen for fremdrift av ROV. Eksempelvis kan knapp A brukes til å aktivere/deaktivere 'hold dybde' modus, som vist her.

Kontrolleren skal styre **jag** basert på X_{LS} (X-retning, venstre stikke) og svai basert på Y_{LS} (Y-retning, venstre stikke). En kombinasjon av disse skal gi korrelerende utslag på ROV-ens pådragsvinkel. Dette blir direkte implementert i utregning av motorpådrag som beskrevet i delkapittel 7.3. Videre skal **hiv** styres av X_{RS} (X-retning, høyre stikke) og **gir** styres av Y_{RS} (Y-retning, høyre stikke). Y_{RS} vil dermed

påvirke ROV-ens bevegelser i horisontal retning, mens X_{RS} jobber uavhengig av disse.

Maksimalt utslag settes til en verdi mellom 10% og 100%, og skal kunne endres fortløpende av piloten. Den påvirker pådraget til hver enkelt motor ved

$$P_s = P_m \cdot \alpha \tag{7.4}$$

Hvor:
- P_s = Skalert pådrag
- P_m = Motorpådrag
- α = Skaleringsverdi [%]

7.5 Implementering av styredata

Styresystemet som blir implementert består i hovedsak av 4 deler:

1. Et program for å ta inn og tolke signaler fra styrestikken.

2. Et program for å sortere disse signalene til korrekte variabler og sende videre til mikrokontroller.

3. En seksjon for transformering av kontrollerens inndata til et mer brukervennlig dataområde.

4. En seksjon med logisk behandling av disse signalene for å bestemme, samt sette pådrag til hver enkelt motor.

For å tolke signaler fra styrestikken ble det i første omgang tatt utgangspunkt i et bibliotek hentet fra pygame-xbox360controller [53]. Videre lagde vi et Python script som leste disse signalene kontinuerlig, behandlet de, og genererte en JSON streng som inneholdt alle thrusternes ønskede pådrag. Dette skulle så bli sendt til en lokal server som igjen sendte verdiene videre til mikrokontrolleren.

I ettertid ble denne strukturen gjort om, da det viste seg at det var mer gunstig å lese inn styringssignal via JavaScript og sende dette direkte til serveren, for så å behandle og beregne nødvendige verdier på selve mikrokontrolleren. Dette gir flere fordeler, for eksempel ved at en pilot kan fjernstyre ROV-en ved å koble seg til serveren, uten å måtte installere ekstra programvare, da JavaScript er innebygd i de fleste typer nettlesere. Det gir også mulighet for å lage et integrert styringsgrensesnitt på serveren, vha. enten berøringsskjerm eller tastatur, i tilfelle piloten ikke har en Xbox360-kontroller tilgjengelig.

Innlesing av styrestikke i JavaScript

Innlesing av styrestikkens data gjøres ved hjelp av Gamepad API [54]. Gamepad API er et integrert API i de fleste nettlesere, som gir mulighet for å lese status til en tilkoblet styrestikke ved hjelp av JavaScript. For å visualisere dataen i en nettleser ble det skrevet et script som vist under, med inpsirasjon fra [55]:

```
1
2  window.addEventListener('gamepadconnected', event => {
3      console.log('Gamepad connected:')
4      console.log(event.gamepad)
5  })
6
7  window.addEventListener('gamepaddisconnected', event => {
8      console.log('Gamepad disconnected:')
9      console.log(event.gamepad)
10 })
11
12 const gamepadDisplay = document.getElementById('gamepad-display'
       )
13
14 function update() {
15     const  gamepads = navigator.getGamepads()
16     if (gamepads[0]) {
17         const gamepadState = {
18             id: gamepads[0].id,
19             axes: [
20                 { Left_x: gamepads[0].axes[0].toFixed(2)},
21                 { Left_y: gamepads[0].axes[1].toFixed(2)},
22                 { Right_x: gamepads[0].axes[2].toFixed(2)},
23                 { Right_y: gamepads[0].axes[3].toFixed(2)},
24             ],
25             buttons: [
26                 { A: gamepads[0].buttons[0].pressed },
27                 { B: gamepads[0].buttons[1].pressed },
28                 { X: gamepads[0].buttons[2].pressed },
29                 { Y: gamepads[0].buttons[3].pressed },
30                 { LB: gamepads[0].buttons[4].pressed },
31                 { RB: gamepads[0].buttons[5].pressed },
32                 { LT: gamepads[0].buttons[6].pressed },
33                 { RT: gamepads[0].buttons[7].pressed },
34                 { Back: gamepads[0].buttons[8].pressed },
35                 { Start: gamepads[0].buttons[9].pressed },
36                 { LST: gamepads[0].buttons[10].pressed },
37                 { RST: gamepads[0].buttons[11].pressed },
38                 { PAD_UP: gamepads[0].buttons[12].pressed },
39                 { PAD_DOWN: gamepads[0].buttons[13].pressed },
40                 { PAD_LEFT: gamepads[0].buttons[14].pressed },
41                 { PAD_RIGHT: gamepads[0].buttons[15].pressed },
42             ]
43         }
44         gamepadDisplay.textContent = JSON.stringify(gamepadState
       , null, 2)
45     }
46     window.requestAnimationFrame(update)
47 }
48
49 window.requestAnimationFrame(update)
```

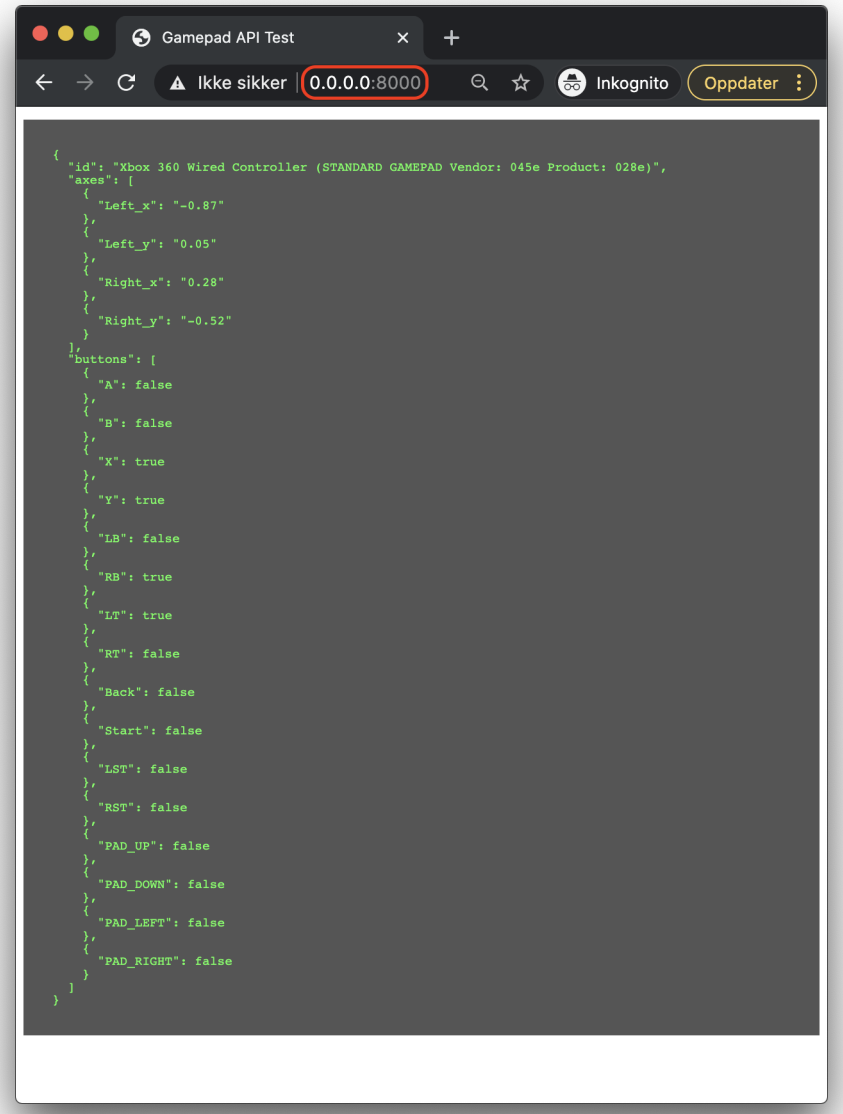
Dette programmet henter status for alle knapper på en styrestikke, her satt opp spesifikt for en Xbox360-kontroller. Deretter oppdaterer den et vindu kontinuerlig,

som igjen kan bli visualisert i en nettleser ved hjelp av HTML5:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF=8">
    <title>Gamepad API Test</title>
    <style>
        pre {
            background: #555;
            color: #5F5;
            padding: 30px;
        }
    </style>
</head>
<body>
    <pre id="gamepad-display"></pre>

    <script src="/js/app.js"></script>
</body>
</html>
```

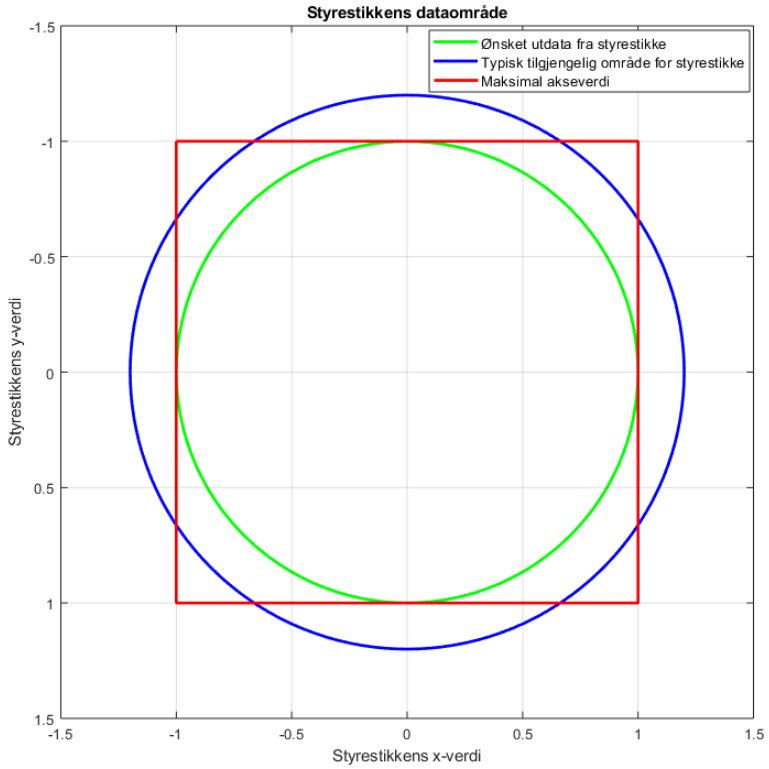
Disse filene kjøres sammen ved bruk av en **Python3 HTTP server**. Ved å dermed åpne den lokale nettsiden med adresse **http://0.0.0.0:8000** får man følgende resultat:



Figur 7.4: Figuren viser utskrift på nettsiden som resultat av kodesnuttene vist over. Nettadressen er vist i rød sirkel. Ved tidspunktet bildet ble tatt kan en se at knappene **X, Y, RB og LT** var inntrykket og får verdien 'true', samt at venstre og høyre styrestikke er beveget.

Transformering av kontrollerens inndata

Før man bruker styrestikkens akseverdier til å beregne motorpådrag er det viktig å sørge for at de er på et brukervennlig, intuitivt - og ikke minst fysisk tilgjengelig område. Styrestikkene på Xbox360-kontrolleren er bundet av en sirkulær plastikkring, og det er dermed ønskelig at koordinatene fra stikkene er definert i et sirkulært område. Kontrolleren gir derimot verdier på et kvadratisk x- og y-plan med verdier fra -1 til 1, se figur 7.5.



Figur 7.5: Visualisering av en typisk kontrollers tilgjengelige styreområde (blå sirkel) bundet av sirkulær plastikkring, kontrollerens avgrensede område (rød firkant) grunnet maksimale akseverdier [-1, 1], og nytt ønsket område (grønn sirkel).

Som vist i figur 7.5 er styrestikkens naturlige dataområde (blå) en sirkel, og maksimal akseverdi (rød) et kvadrat. Kontrolleren vil dermed redusere alle verdier utenfor det røde området til 1. Som et resultat av dette vil et maksimalt utslag på styrestikken gi forskjellig størrelse på pådragsvektoren, alt etter hvilken retning den peker. En måte å løse dette på er å definere et sirkulært område der ingen verdier vil overstige maksimale akseverdier, vist som grønn sirkel i figuren, og skalere pådragsvektoren til dette området.

Dette kan gjøres med noen enkle matematiske utregninger basert på fremgangsmåte hentet fra [56], samt logisk behandling av resultatet for å styre ROV-en i ønsket retning. Først finner man størrelsen og vinkelen til retningsvektoren ved:

$$P = \sqrt{X_{LS}^2 + Y_{LS}^2} \tag{7.5}$$

Hvor:
- P = Retningsvektorens størrelse
- X_{LS} = venstre styrespakes x-verdi
- Y_{LS} = venstre styrespakes y-verdi

og

$$\theta = atan2(Y_{LS}, X_{LS}) \tag{7.6}$$

Hvor:
- θ = Retningsvektorens vinkel
- X_{LS} = venstre styrespakes x-verdi
- Y_{LS} = venstre styrespakes y-verdi
- atan2 er en funksjon som definerer vinkelen i polarkoordinater, og har et område fra (-π, π].

Retningsvektorens størrelse og vinkel for høyre styrestikke blir kalkulert på samme måte. Da retningsvektoren kan overstige ønsket maksimal radius på r = 1 i de fire hjørnene, settes den til:

$$P = \begin{cases} P, & \text{if } P < 1 \\ 1, & \text{ellers} \end{cases} \tag{7.7}$$

Hvor P = retningsvektorens størrelse.

Videre finner man de nye koordinatene ved

$$\begin{aligned} X_{LS} &= \cos(\theta) \cdot P \\ Y_{LS} &= \sin(\theta) \cdot P \end{aligned} \tag{7.8}$$

Dermed vil alle verdier være korrekt justert med en maksimal verdi på $r = 1$, med full utnyttelse av kontrollspakenes tilgjengelige område. Dette er implementert i samme program som leser inn verdier fra styrestikken, som vist under:

```
1  // ---Konverter styringsinput fra et kvadratisk omraade til et
       sirkulaert omraade
2  function konverter(venstre_x, venstre_y, hoyre_x, hoyre_y) {
3
4      // Finner lengden til paadragsvektorene, ref. ligning 7.5 i
       rapport.
5      paadrag_venstre = Math.sqrt(Math.pow(venstre_x, 2) + Math.
       pow(venstre_y, 2))
6      paadrag_hoyre = Math.sqrt(Math.pow(hoyre_x, 2) + Math.pow(
       hoyre_y, 2))
7
8      // Finner vinkelen til paadragsvektorene, ref. ligning 7.6 i
       rapport.
9      vinkel_venstre = Math.atan2(venstre_y, venstre_x)
10     vinkel_hoyre = Math.atan2(hoyre_y, hoyre_x)
11
12     // Setter maksimal lengde til 1, ref. ligning 7.7 i rapport.
13     if (paadrag_venstre > 1) {
14         paadrag_venstre = 1
15     }
16     if (paadrag_hoyre > 1) {
17         paadrag_hoyre = 1
18     }
19
20     // Beregner nye koordinater fra nye paadragsvektorer, ref.
       ligning 7.8 i rapport.
21     venstre_x = Math.cos(vinkel_venstre)*paadrag_venstre
```

```
22      venstre_y = Math.sin(vinkel_venstre)*paadrag_venstre
23      hoyre_x = Math.cos(vinkel_hoyre)*paadrag_hoyre
24      hoyre_y = Math.sin(vinkel_hoyre)*paadrag_hoyre
25
26      return [venstre_x, venstre_y, hoyre_x, hoyre_y,
        paadrag_venstre, paadrag_hoyre, vinkel_venstre, vinkel_hoyre]
27  }
```

7.5.1 Styring av thrustere

Venstre styrespake

Venstre styrespake's styresignaler skal styre horisontal, translatorisk bevegelse, og blir beregnet ved følgende ligninger, gitt av ligning 7.3 og ligning 7.4:

$$\begin{aligned} P_{hfv} &= \alpha \cdot P_m \cdot (0.8191 \cdot cos(\theta) - 0.5735 \cdot sin(\theta)) \\ P_{hfh} &= \alpha \cdot P_m \cdot (-0.8191 \cdot cos(\theta) - 0.5735 \cdot sin(\theta)) \\ P_{hbh} &= -P_{hfv} \\ P_{hbv} &= -P_{hfh} \end{aligned} \tag{7.9}$$

Hvor:
- P_{hfv} = pådrag på thruster: (h)orisontal (f)remme (v)enstre.
- P_{hfh} = pådrag på thruster: (h)orisontal (f)remme (h)øyre.
- P_{hbh} = pådrag på thruster: (h)orisontal (b)ak (h)øyre.
- P_{hbv} = pådrag på thruster: (h)orisontal (b)ak (v)enstre.
- α = skaleringsverdi.
- P_m = motorpådrag.
- θ = pådragsvinkel, som beregnet av $atan2(Y_{LS}, X_{LS})$ i ligning 7.6.

Høyre styrespake

Høyre styrespake's styresignaler skal styre både hiv opp og ned, samt gir venstre og høyre, ref. figur 7.1. Y_{RS} står for justeringer i hiv-retning, og X_{RS} står for justeringer i gir-retning. Pådrag i hiv-retning blir beregnet ved følgende ligning:

$$P_{vfv}, P_{vfh}, P_{vbh}, P_{vbv} = \alpha \cdot Y_{RS} \tag{7.10}$$

Hvor:
- P_{vfv} = pådrag thruster: (v)ertikal (f)remme (v)enstre.
- P_{vfh} = pådrag thruster: (v)ertikal (f)remme (h)øyre.
- P_{vbh} = pådrag thruster: (v)ertikal (b)ak (h)øyre.
- P_{vbv} = pådrag thruster: (v)ertikal (b)ak (v)enstre.
- α = skaleringsverdi.
- Y_{RS} er høyre styrespake's Y-verdi etter konvertering til nytt, sirkulært dataområde.

Videre beregnes pådrag i gir-retning ved følgende ligning:

dersom X_{RS} er større enn null:

$$P_{hfv}, P_{hbh} = \alpha \cdot X_{RS} \tag{7.11}$$

dersom X_{RS} er mindre enn null:

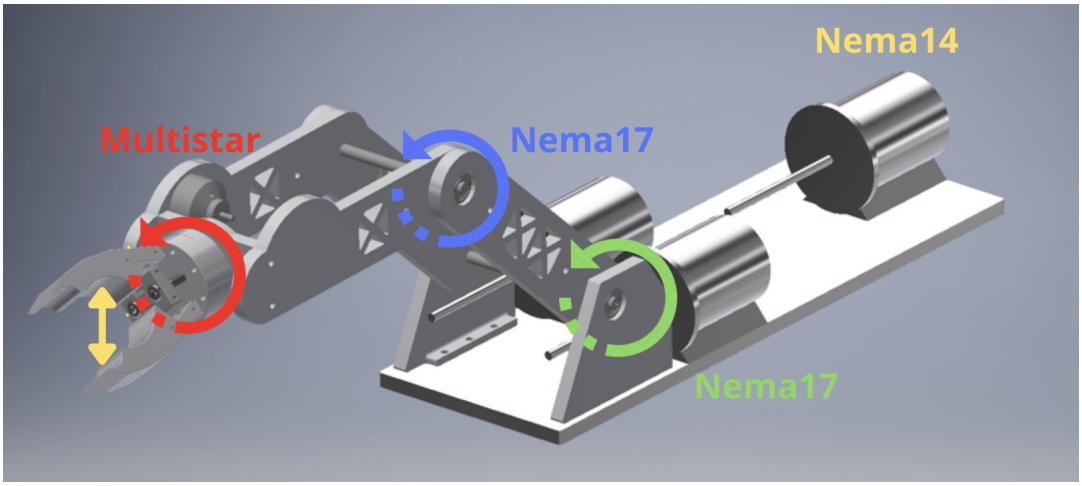
$$P_{hfh}, P_{hbv} = \alpha \cdot X_{RS} \tag{7.12}$$

Hvor:
- P_{hfv} = pådrag thruster: (h)orisontal (f)remme (v)enstre.
- P_{hfh} = pådrag thruster: (h)orisontal (f)remme (h)øyre.
- P_{hbh} = pådrag thruster: (h)orisontal (b)ak (h)øyre.
- P_{hbv} = pådrag thruster: (h)orisontal (b)ak (v)enstre.
- α = skaleringsverdi.
- X_{RS} er høyre styrespake's X-verdi etter konvertering til nytt, sirkulært dataområde.

Det gires altså kun ved bruk av to thrustere om gangen.

7.5.2 Styring av manipulator

Manipulatorarmen har totalt 3 stegmotorer og 1 BLDC-motor. Disse styrer henholdsvis **nedre arm**, **øvre arm**, **klo** og **rotasjon**, vist i figur under.



Figur 7.6: Figuren viser manipulatorarmen. De fire bevegelsespunktene for **nedre arm**, **øvre arm**, **klype** og **rotasjon** er indikert med henholdsvis **grønn**, **blå**, **gul** og **rød** markering.

For optimal kontroll på manipulatorarmens posisjon er det ønskelig å bruke enkodere og/eller endebrytere. Da denne oppgaven har en relativt kort tidsfrist for et så omfattende prosjekt som dette er, har ikke slik funksjonalitet blitt implementert. Manipulatorarmen er bygget på en måte som gjør at den ikke blir ødelagt i et tilfelle hvor motorene gir pådrag utenfor armens rekkevidde, og ROV-ens fremdrift, samt rull- stamp- og dybderegulering ble derfor prioritert først. Det er derfor mulig for en pilot å styre manipulatoren i alle dens frihetsgrader, dog med visse begrensninger.

Manipulatorarmen styres av en egen kontroller av samme type som ROV-ens thrustere, nemlig en Xbox360-kontroller. Dens knapper vil styre manipulatorens funksjoner som vist i tabellen under:

Knapp	Funksjon	Pådragsverdi	Retningsverdi
A	Nedre arm inn	α_m	-1
B	Nedre arm ut	α_m	1
X	Øvre arm inn	α_m	-1
Y	Øvre arm ut	α_m	1
LB	Rotasjon mot klokken	α_m	-1
RB	Rotasjon med klokken	α_m	1
LT	Klyp inn	α_m	-1
RT	Klyp ut	α_m	1

Tabell 7.1: Knappefunksjoner på Xbox360-kontroller for manipulatorarm. α_m er skaleringsverdi som fungerer på samme måte som α forklart tidligere, og justerer manipulatormotorenes hastighet. Retningsverdi indikerer kjøreretningen til motoren.

Et knappetrykk vil gi et pådrag til korrelerende motor med en verdi beregnet som følger:

$$P_K = \alpha_m \cdot retningsverdi \tag{7.13}$$

hvor:
- P_K = korrelerende motor.
- α_m = skaleringsverdi, som direkte bestemmer hastigheten til motoren.
- positiv retningsverdi gir kjøreretning med klokken, negativ retningsverdi gir kjøreretning mot klokken.

En større skaleringsverdi vil dermed gi et større pådrag, som dermed gir høyere hastighet. Piloten kan dermed styre alle manipulatorens ledd i ønsket hastighet, ved å justere skaleringsverdien som forklart i delkapittel 7.4.

7.6 Implementering på mikrokontroller

Som forklart i kapittel 6 blir hver motor styrt av PWM- og PFM-signal. PWM-signalene, som styrer de 8 thrusterne for fremdrift, må ha en frekvens på 200 Hz. Motorkontrolleren har som forklart i delkapittel 3.4.1 et signalområde fra 1100 μs til 1900 μs, med nullpunkt i 1500 μs. Ved 200 Hz har signalet en total periode på:

$$\frac{1}{200} = 0.005\ s = 5000\ \mu s \tag{7.14}$$

som dermed gir at thrusterne styres i følgende driftssyklusområde:

$$\begin{aligned} P_{maksrevers} &= \frac{1100}{5000} \cdot 100\ \% = 22\ \% \\ P_{stillestende} &= \frac{1500}{5000} \cdot 100\ \% = 30\ \% \\ P_{maksforover} &= \frac{1900}{5000} \cdot 100\ \% = 38\ \% \end{aligned} \tag{7.15}$$

Motorkontrollerne som styrer de 3 stegmotorene på manipulatorarmen blir derimot styrt av individuelle pulser, uavhengig av deres bredde. En puls indikerer et steg, og

hastigheten til motoren blir dermed bestemt av frekvensen til signalet. Fra motorkontrollernes datablad [38] ser man at frekvensområdet er 0 - 60 kHz. Det er også gitt at stegmotorene driver manipulatorarmens nedre og øvre ledd gjennom en 1 : 50 gir, av Manipulatorgruppen [17]. Videre blir manipulatorens BLDC-motor styrt på samme måte og med samme driftsyklysområde som thrusterne.

En timer på en STM32 mikrokontroller er enkelt forklart en logisk krets som teller oppover for hver klokkesyklus. Hvor mye timeren skal telle til kan bestemmes etter ønske, og en kan videre bruke denne telleperioden til å utføre forskjellige handlinger. I våres tilfelle ønsker vi å bruke timerene til å generere PWM-signaler og modulere på-tiden til signalet, samt PFM-signaler der man modulerer frekvensen til signalet.

Mikrokontrolleren har totalt 14 timere [57] hvorav vi tar i bruk 8, hvor igjen 6 av disse er for motorstyring. De resterende to sørger for belysning og viftestyring. Timerene kan ha opptil 4 tilgjengelige PWM/PFM utganger, og man kan dermed styre flere motorer fra en og samme timer. En av begrensningene med dette er at alle kanalene til en timer vil ha samme frekvens (detaljert forklaring senere i kapittelet). Motorkontrollerene til de 9 BLDC-motorene krever et signal med fast frekvens, hvor kun på-tiden blir modulert. Signalet til de 3 resterende stegmotorenes motorkontrollerne blir kontrollert ved endring i frekvens, og trenger dermed egne timere. På grunnlag av dette er fordelingen av timere og timerkanaler blitt som følger:

Timer	**Kanal**	**Motor**
Timer 1	Kanal 1	Horisontal fremme venstre
	Kanal 2	Horisontal fremme høyre
	Kanal 3	Horisontal bak høyre
	Kanal 4	Horisontal bak venstre
Timer 2	Kanal 1	Vertikal fremme venstre
	Kanal 2	Vertikal fremme høyre
	Kanal 3	Vertikal bak høyre
	Kanal 4	Vertikal bak venstre
Timer 3	Kanal 1	Rotasjon, manipulator
Timer 4	Kanal 1	Nedre ledd, manipulator
Timer 5	Kanal 1	Øvde ledd, manipulator
Timer 8	Kanal 2	Klype, manipulator

Tabell 7.2: Tabelloversikt over timer- og timerkanalfordeling.

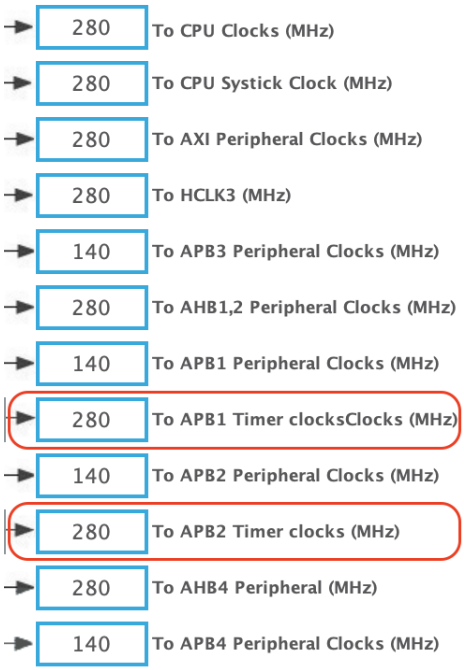
For å implementere dette på mikrokontrolleren, blir programmene CubeMX og CubeIDE brukt, som forklart i delkapittel 6.3. Før man justerer klokkefrekvens for timerene må man vite hvilke perifermoduler som er relevante. Dette finner man ved å lese datablad for mikrokontrolleren, og er som vist under:

Peripheral		I_{DD}(Typ)				Unit
		VOS0	VOS1	VOS2	VOS3	
AHB2	HSEM	0.10	0.10	0.10	0.10	µA/MHz
	RNG registers	1.50	1.40	1.20	1.10	
	RNG kernel	10.00	9.70	9.50	9.20	
	SDMMC2 registers	6.80	6.30	5.70	5.20	
	SDMMC2 kernel	2.30	2.10	1.90	1.70	
	BDMA1	1.70	1.60	1.50	1.30	
	AHBSRAM1	0.70	0.70	0.60	0.60	
	AHBSRAM2	0.70	0.60	0.60	0.50	
	Bridge	9.10	8.40	7.70	7.00	
AHB4	GPIOA	2.00	1.80	1.70	1.50	
	GPIOB	1.80	1.70	1.50	1.40	
	GPIOC	2.00	1.80	1.70	1.50	
	GPIOD	2.00	1.80	1.70	1.50	
	GPIOE	1.90	1.80	1.60	1.50	
	GPIOF	1.90	1.80	1.60	1.50	
	GPIOG	2.00	1.80	1.70	1.50	
	GPIOH	1.90	1.80	1.60	1.50	
	GPIOI	1.90	1.80	1.60	1.50	
	GPIOJ	1.90	1.80	1.60	1.50	
	GPIOK	2.00	1.80	1.70	1.50	
	BDMA2	4.20	3.90	3.50	3.20	
	SRDSRAM	0.60	0.50	0.50	0.50	
	BKPRAM	0.80	0.70	0.70	0.60	
	IWDG	0.07	0.07	0.07	0.07	
	Bridge	0.10	0.10	0.10	0.10	
APB3	LTDC	12.00	11.00	9.80	8.90	
	WWDG1	1.10	1.00	0.90	0.90	
	Bridge	0.10	0.10	0.10	0.10	
APB1	TIM2	7.50	6.90	6.30	6.20	
	TIM3	6.30	5.90	5.40	4.90	
	TIM4	5.80	5.40	4.90	4.50	
	TIM5	7.20	6.70	6.10	5.60	
	TIM6	1.60	1.50	1.30	1.20	
	TIM7	1.60	1.40	1.30	1.20	
	TIM12	3.60	3.30	3.00	2.80	
	TIM13	2.80	2.60	2.40	2.10	
	TIM14	2.50	2.30	2.10	1.90	
	LPTIM1 registers	0.80	0.80	0.70	0.60	
	LPTIM1 kernel	2.20	2.00	1.80	1.70	

Peripheral		I_{DD}(Typ)				Unit
		VOS0	VOS1	VOS2	VOS3	
APB1	SPI2 registers	2.20	2.00	1.80	1.70	µA/MHz
	SPI2 kernel	0.90	0.80	0.80	0.70	
	SPI3 registers	2.70	2.40	2.30	2.00	
	SPI3 kernel	0.90	0.80	0.70	0.70	
	SPDIFRX1 registers	0.60	0.50	0.50	0.40	
	SPDIFRX1 kernel	2.90	2.70	2.50	2.20	
	USART2 registers	2.00	1.80	1.70	1.50	
	USART2 kernel	4.60	4.30	3.90	3.60	
	USART3 registers	2.00	1.80	1.70	1.50	
	USART3 kernel	4.50	4.20	3.80	3.40	
	UART4 registers	1.70	1.60	1.50	1.30	
	UART4 kernel	3.70	3.40	3.10	2.80	
	UART5 registers	1.80	1.70	1.50	1.40	
	UART5 kernel	3.80	3.50	3.20	2.90	
	I2C1 registers	0.90	0.80	0.80	0.70	
	I2C1 kernel	2.10	2.00	1.80	1.70	
	I2C2 registers	0.90	0.80	0.70	0.70	
	I2C2 kernel	2.10	1.90	1.80	1.60	
	I2C3 registers	0.90	0.80	0.70	0.70	
	I2C3 kernel	2.20	2.00	1.80	1.70	
	HDMICEC registers	0.50	0.50	0.40	0.40	
	HDMICEC kernel	0.10	0.10	0.10	0.10	
	DAC1	1.40	1.30	1.20	1.10	
	UART7 registers	1.80	1.70	1.50	1.40	
	UART7 kernel	3.80	3.50	3.20	2.90	
	UART8 registers	2.10	2.00	1.80	1.70	
	UART8 kernel	3.80	3.50	3.20	2.90	
	Bridge	0.30	0.30	0.20	0.10	
	CRS	0.50	0.40	0.40	0.40	
	SWP registers	2.30	2.10	2.00	1.80	
	SWP kernel	0.10	0.10	0.10	0.10	
	OPAMP	4.20	3.80	3.50	3.20	
	MDIO	3.10	2.90	2.60	2.40	
	FDCAN registers	17.00	16.00	15.00	14.00	
	FDCAN kernel	5.60	4.80	3.50	1.10	
	Bridge	0.10	0.10	0.10	0.10	
APB2	TIM1	9.80	9.10	8.30	7.60	
	TIM8	9.50	8.80	8.00	7.30	
	USART1 registers	0.10	0.10	0.10	0.10	

Figur 7.7: Skjermbilde av perifermoduler for STM32H7A3, hentet fra [57]. Alle timere hører som vist til perifermodulene APB1 og APB2.

Under **Clock Configuration** i CubeMX kan man se at **APB1-** og **APB2 Timer clocks** er satt til 280 MHz, som vist i figuren under:



Figur 7.8: Skjermbilde av CubeMX, **Clock Configuration**. De relevante modulene er markert i rødt.

Videre blir Counter Period (AutoReload Register) **(ARR)**, som bestemmer hvor

mye tellerene skal telle til for en enkelt periode, bestemt av formelen:

$$ARR = \frac{F_T}{F_S} \tag{7.16}$$

hvor:
- ARR = Verdi som tilsier hvor mye timeren skal telle til.
- F_T = Timerens klokkefrekvens.
- F_S = Ønsket frekvens på utgangssignalet.

Ved å bruke denne formelen for de 9 motorkontrollerne som krever en signalfrekvens på 200 Hz, får man $\frac{280\ MHz}{200\ Hz} = 1.4 \cdot 10^6$. De fleste timerene på mikrokontrolleren er dog begrenset til 16 bits verdier, som dette ikke oppfyller. Ved å ta i bruk en Prescaler **(PSC)** kan vi nedjustere timerens frekvens, for så å kunne sette en lovlig verdi. Ny ønsket frekvens og PSC blir beregnet ved:

$$F_{T\_ny} = \frac{F_T}{PSC + 1} \Rightarrow PSC = \frac{F_T}{F_{T\_ny}} - 1 \tag{7.17}$$

hvor:
- F_{T_ny} = Ny ønsket klokkefrekvens for timer.
- PSC = Verdi som brukes til å redusere klokkefrekvens på timer.
- Formelen er hentet fra mikrokontrollerens referansemanual, 42.4.13 [58].

Maksimal telleverdi for en 16 bits timer er $2^{16} - 1 = 65535$. For enkelhetens skyld implementeres telleverdien for PWM-signalene til 50 000, som da blir direkte korrelert til signalets påtid i mikrosekunder med en faktor på 10, nemlig 11000 til 19000. Ny klokkefrekvens blir dermed:

$$F_{T\_ny} = 50000 \cdot 200\ Hz = 10\ MHz \tag{7.18}$$

som igjen blir brukt til å finne PSC-verdi ved:

$$PSC = \frac{280\ MHz}{10\ MHz} - 1 = 27 \tag{7.19}$$

Samme klokkefrekvens blir satt for de tre timerene som står for styring av stegmotorer, og ved å justere på ARR kan en dermed modulere frekvensen til signalene mellom $F_{min} = 153\ Hz$ og $F_{maks} = 65359\ Hz$ gitt av:

$$\begin{aligned} F_{min} &= \frac{10\ MHz}{65535} = 152.59\ Hz \\ F_{max} &= \frac{10\ MHz}{153} = 65359\ Hz \end{aligned} \tag{7.20}$$

Ved å sette en lavere klokkefrekvens for disse timerene kunne en oppnådd mye lavere hastighet for stegmotorene, men ville da ofret oppløsning i høyere frekvenser. Ved å teste eksperimentelt ble det konkludert med å bruke 12800 steg per omdreining

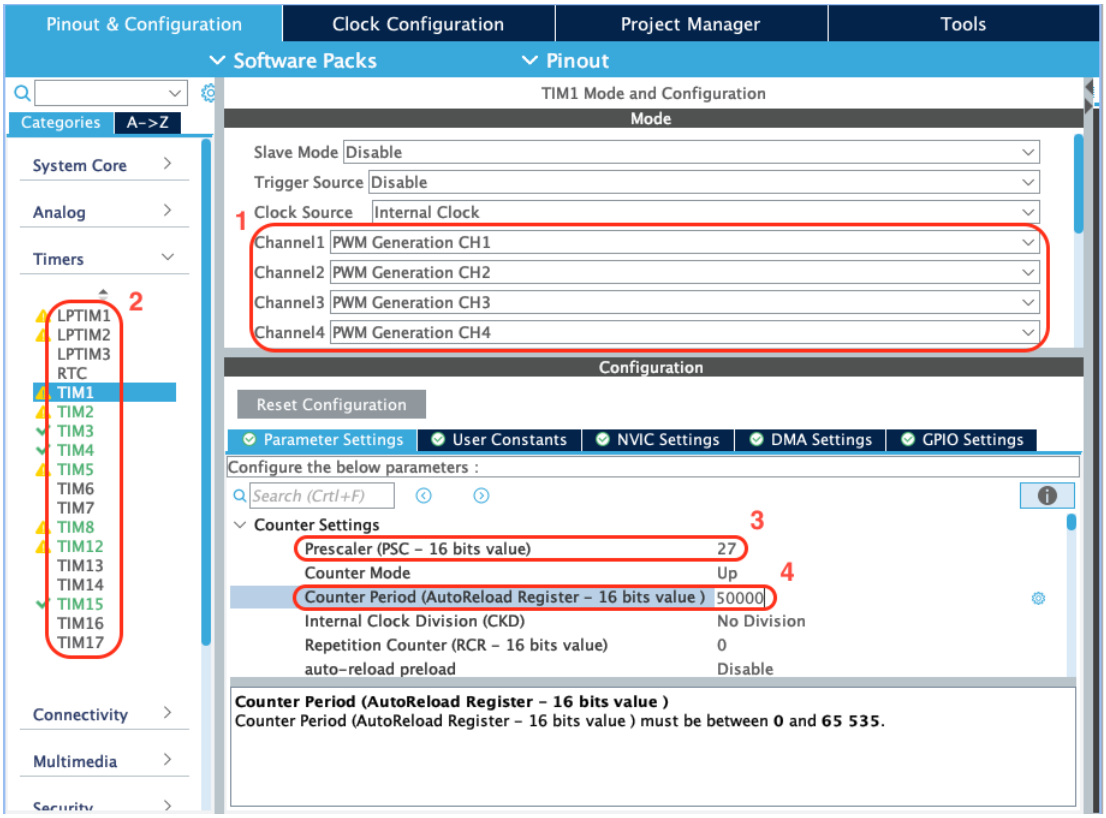
på motorkontrolleren for kortest mulig steg samt høyest mulig moment, og finner dermed laveste mulig rotasjonshastighet ved:

$$RPM = \frac{12800 \ \frac{steg}{revolusjon}}{F_{min}} = \frac{12800 \ \frac{steg}{revolusjon}}{153 \ \frac{steg}{sekund}} = 83.66 \ \frac{sekund}{revolusjon} = 0.71 \ RPM \quad (7.21)$$

Tatt i betraktning at stegmotorene driver manipulatorarmen gjennom en $1:50$ gir som nevnt tidligere, og har en minstehastighet på 0.71 RPM, vil selve leddene til manipulatorarmen ha en minstehastighet på:

$$\frac{0.71}{50} = 0.0142 \ RPM = 0.09 \ \frac{grader}{sekund} \quad (7.22)$$

Oppsett for timere ser da ut som vist her:



Figur 7.9: Skjermbilde av oppsett for Timer 1 i CubeMx. Markert område: 1 viser valg av kanaler, 2 viser aktiverte timere på mikrokontroller, 3 viser prescalerverdi og 4 viser tellerverdi. Figuren viser flere gule varseltrekanter ved siden av timerne i sirkel nr. 2, dette skyldes at noen bestemte moduser for timerne ikke er tilgjengelige. Dette påvirker ikke oppsettet på noen måte.

7.6.1 Styring av thrustere

Beregninger og setting av verdier i timernes utganger er gjort i C. Styringsdata fra kontrolleren på topside blir hentet kontinuerlig, og kan leses mer om i Kommunikasjongruppens rapport [59]. Koden under er implementert i `main.c` under

if-betingelsen for oppdatering av manuell styring, som forklart i delkapittel 6.3. Se forklarende kommentarer i koden under:

filnavn: main.c

```
1          // Kalkuler paadrag for horisontale motorer:
2          // Venstre og hoyre vinkel blir beregnet ved
   funksjonen atan2 som returnerer vinkelen i polarkoordinater,
   i et omraade fra (-pi,pi], ref. ligning 7.6 i rapport.
3       vinkel_venstre = atan2(venstre_y, venstre_x);
4       vinkel_hoyre = atan2(hoyre_y, hoyre_x);
5
6       // Lengden til paadragsvektoren blir funnet ved roten til
   summen av x og y kvadrert, ref. ligning 7.5 i rapport.
7       paadrag_venstre = sqrt(pow(venstre_x, 2) + pow(venstre_y,
   2));
8       paadrag_hoyre = sqrt(pow(hoyre_x, 2) + pow(hoyre_y, 2));
9
10         // Beregner kun motorpaadrag til horisontale motorer
   dersom paadragsvektoren har en lengde.
11      if (paadrag_venstre != 0) {
12         // Beregn motorpaadrag til horisontale motorer (ref
   ligning 7.9 i rapporten)
13        hfv = skalering * paadrag_venstre * (0.8191 * cos(
   vinkel_venstre) - 0.5735 * sin(vinkel_venstre));
14        hfh = skalering * paadrag_venstre * (0.8191 * -cos(
   vinkel_venstre) - 0.5735 * sin(vinkel_venstre));
15        hbh = -hfv;
16        hbv = -hfh;
17      } else {
18        hfv = 0;
19        hfh = 0;
20        hbh = 0;
21        hbv = 0;
22      }
23
24         // Adder skalert motorpaadrag til korrelerende motorer
    for rotasjon
25      if (hoyre_x > 0) {
26         // Rotasjon med klokken, sett ovenfra, ref. ligning
   7.11 i rapport.
27        hfv = hfv + (skalering * hoyre_x);
28        hbh = hbh + (skalering * hoyre_x);
29      } else if (hoyre_x < 0) {
30         // Rotasjon mot klokken, sett ovenfra, ref. ligning
   7.12 i rapport.
31        hfh = hfh + (skalering * hoyre_x);
32        hbv = hbv + (skalering * hoyre_x);
33      }
34
35         // Sett ny duty cycle ved hjelp av max_change_fn, les
   mer om denne funksjonen i neste kodeblokk. Eksempelvis hfv,
   som er en verdi mellom -100 til 100, blir multiplisert med '
   scaling' og addert med startForover/startBakover (
   forhaandsdefinerte konstanter), henholdsvis 38 og
   15200/14800, som skalerer hfv til timerens onskede omraade
   (11000-19000). StartForover/Bakover sorger altsaa for at et
   eksempelvis 1% paadrag gir paatid i 1523.8 mikrosekunder, da
   motoren starter paa 1520, og 1480 for bakover. Standby er
```

```
    15000 som indikerer null paadrag.
36
37          if (hfv > 0) {
38          dc_hfv = max_change_fn(TIM1->CCR1, hfv*scaling +
    startForover, max_change, standby);
39          } else if (hfv < 0) {
40              dc_hfv = max_change_fn(TIM1->CCR1, hfv*scaling +
    startBakover, max_change, standby);
41          } else if (hfv == 0) {
42              dc_hfv = max_change_fn(TIM1->CCR1, hfv*scaling +
    standby, max_change, standby);
43          }
44
45          if (hfh > 0) {
46          dc_hfh = max_change_fn(TIM1->CCR1, hfh*scaling +
    startForover, max_change, standby);
47          } else if (hfh < 0) {
48              dc_hfh = max_change_fn(TIM1->CCR1, hfh*scaling +
    startBakover, max_change, standby);
49          } else if (hfh == 0) {
50              dc_hfh = max_change_fn(TIM1->CCR1, hfh*scaling +
    standby, max_change, standby);
51          }
52
53          if (hbh > 0) {
54          dc_hbh = max_change_fn(TIM1->CCR1, hbh*scaling +
    startForover, max_change, standby);
55          } else if (hbh < 0) {
56              dc_hbh = max_change_fn(TIM1->CCR1, hbh*scaling +
    startBakover, max_change, standby);
57          } else if (hbh == 0) {
58              dc_hbh = max_change_fn(TIM1->CCR1, hbh*scaling +
    standby, max_change, standby);
59          }
60
61          if (hbv > 0) {
62          dc_hbv = max_change_fn(TIM1->CCR1, hbv*scaling +
    startForover, max_change, standby);
63          } else if (hbv < 0) {
64              dc_hbv = max_change_fn(TIM1->CCR1, hbv*scaling +
    startBakover, max_change, standby);
65          } else if (hbv == 0) {
66              dc_hbv = max_change_fn(TIM1->CCR1, hbv*scaling +
    standby, max_change, standby);
67          }
68
69          // CCRx (Capture/compare register x) definerer pulsens
     start-tid. Dvs. en kanal paa timeren er lav helt til CCRx
    verdien er naadd, og deretter hoy helt til timeren har telt
    ferdig. Deretter gaar den lav igjen, og begynner aa telle fra
     0 igjen.
70
71        TIM1->CCR1 = dc_hfv;
72        TIM1->CCR2 = dc_hfh;
73        TIM1->CCR3 = dc_hbh;
74        TIM1->CCR4 = dc_hbv;
75
76        TIM2->CCR1 = dc_vfv;
```

```
77        TIM2->CCR2 = dc_vfh;
78        TIM2->CCR3 = dc_vbh;
79        TIM2->CCR4 = dc_vbv;
```

Videre er funksjonen **max_change_fn** som vist under:
filnavn: main.c

```
1  // max_change_fn sorger for at paadraget ikke endrer seg for mye
       om gangen, som et sikkerhetstiltak for aa unngaa store hopp
   i stromtrekk, samt enklere styring for pilot.
2  // funksjonen tar inn forrige paadragsverdi, ny onsket
   paadragsverdi, maksimal endring tillat og verdi for nullpunkt
   . Funksjonen returnerer ny beregnet pwm.
3  float max_change_fn(float prev_pwm, float new_pwm, float
   max_change, float standby) {
4    if (new_pwm > standby) { // fremover
5      if (prev_pwm < standby) {
6          // Retning for motoren har endret. Setter da ny verdi
   til standby for aa stanse motoren, for saa aa la den kjore
   videre i motsatt retning neste gang.
7        new_pwm = standby;
8      } else {
9          // Dersom endringen er for stor endres verdien kun saa
   mye som er tillat.
10       if (new_pwm > (prev_pwm + max_change)){
11         new_pwm = prev_pwm + max_change;
12       }
13     }
14   } else if (new_pwm < standby) { // bakover
15     if (prev_pwm > standby) {
16         // Retning for motoren har endret. Setter da ny verdi
   til standby for aa stanse motoren, for saa aa la den kjore
   videre i motsatt retning nestegang.
17       new_pwm = standby;
18     } else {
19         // Dersom endringen er for stor endres verdien kun saa
   mye som er tillat.
20       if (new_pwm < (prev_pwm - max_change)){
21         new_pwm = prev_pwm - max_change;
22       }
23     }
24   }
25   // Returnerer ny verdi
26   return new_pwm;
27 }
```

7.6.2 Styring av manipulator

Manipulatoren styres som forklart ved hjelp av dedikerte knapper som er angitt bestemte motorer og retninger, se tabell 7.2. Videre blir pådraget beregnet av ligning 7.13. Dette blir dermed implementert med koden under, plassert i main.c under **if**-betingelsen for oppdatering av manipulatorstyring som forklart i delkapittel 6.3.

Se forklarende kommentarer:
filnavn: main.c

```
1  // Sjekk om en av knappene for manipulatormotor 1 (stegmotor) er
       trykket (-1 for bakover, 1 for fremover).
2    if (manip1 != 0) {
3      // 65535 for laveste hastighet (153Hz), 153 for hoyeste
       hastighet (65359Hz).
4      // Juster ARR for frekvensmodulering, sett CCRx til
       halvparten av ARR for 50% paatid.
5      TIM4->ARR = (65535-65535*skalering_m);
6      TIM4->CCR1 = (65535-65535*skalering_m)/2;
7      if (manip1 > 0) {
8        // Sett motorretning forover
9        HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, GPIO_PIN_SET);
10     } else if (manip1 < 0) {
11       // Sett motorretning bakover
12       HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, GPIO_PIN_RESET);
13     }
14   } else {
15      TIM4->CCR1 = 0;
16   }
17
18   // Sjekk om en av knappene for manipulatormotor 2 (stegmotor)
       er trykket (-1 for bakover, 1 for fremover).
19   if (manip2 != 0) {
20     // 65535 for laveste hastighet (153Hz), 153 for hoyeste
       hastighet (65359Hz).
21     // Juster ARR for frekvensmodulering, sett CCRx til
       halvparten av ARR for 50% paatid.
22     TIM5->ARR = (65535-65535*skalering_m);
23     TIM5->CCR1 = (65535-65535*skalering_m)/2;
24     if (manip2 > 0) {
25       // Sett motorretning forover
26       HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, GPIO_PIN_SET);
27     } else if (manip2 < 0) {
28       // Sett motorretning bakover
29       HAL_GPIO_WritePin(GPIOB, GPIO_PIN_5, GPIO_PIN_RESET);
30     }
31   } else {
32      TIM5->CCR1 = 0;
33   }
34
35   // Sjekk om en av knappene for manipulatormotor 3 (stegmotor)
       er trykket (-1 for bakover, 1 for fremover).
36   if (manip3 != 0) {
37     // 65535 for laveste hastighet (153Hz), 153 for hoyeste
       hastighet (65359Hz).
38     // Juster ARR for frekvensmodulering, sett CCRx til
       halvparten av ARR for 50% paatid.
39     TIM8->ARR = (65535-65535*skalering_m);
40     TIM8->CCR3 = (65535-65535*skalering_m)/2;
41     if (manip3 > 0) {
42       // Sett motorretning forover
43       HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, GPIO_PIN_SET);
44     } else if (manip3 < 0) {
45       // Sett motorretning bakover
46       HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, GPIO_PIN_RESET);
```

```
47      }
48    } else {
49       TIM8->CCR1 = 0;
50    }
51
52    // Sjekk om en av knappene for manipulatormotor 4 (BLDC-motor)
        er trykket (-1 for bakover, 1 for fremover).
53    if (manip4 != 0) {
54      if (manip4 > 0) {
55        TIM3->CCR1 = skalering_m * scaling + startForover;
56      } else if (manip4 < 0) {
57        TIM3->CCR1 = -skalering_m * scaling + startBakover;
58      }
59    } else {
60      TIM3->CCR1 = standby;
61    }
```

Kapittel 8

Modellering

Innhold

8.1 Hvilke frihetsgrader som skal reguleres 97
8.1.1 Behovet for regulator til manipulator 98
8.2 Viktige momenter for modelleringen 98
8.3 Dybdeprosessen 100
8.3.1 Matematisk modell av dybdeprosessen 102
8.3.2 Simulering av dybdeprosessen 105
8.4 Rull- og stampprosessene 108
8.4.1 Matematisk modell av rull- og stampprosessen 113
8.4.2 Simulering av rullprosessen 120
8.4.3 Konklusjon for rull- og stampprosessen 121

I dette kapittellet vil det bli gjort rede for behovet for hvilke frihetsgrader det skal lages matematisk modell av, som grunnlag for reguleringssystemet. De matematiske modellene vil bli simulert i Simulink for å visualisere arbeidet som er blitt gjort teoretisk. Til slutt vil det også bli gjort en konklusjon om de ulike prosessene.

8.1 Hvilke frihetsgrader som skal reguleres

For å vite hvilke frihetsgrader det skal lages matematisk modell og regulatorer for, er det viktig å finne ut av hva som kan hjelpe piloten mest mulig for å manøvrere ROV-en. En regulator i dette tilfellet skal hjelpe piloten til å måtte gjøre så lite som mulig manuelt. Det ville vært upraktisk å hatt en regulator for jag, svai og gir, ettersom piloten mest sannsynlig vil styre disse selv. Det ville også krevd enda flere sensorer for å måle referanser i hver frihetsgrad, som ville vært unødvendig å implementert dersom det ikke er behov. Noe som derimot kan hjelpe piloten er regulering for dybde, rull og stamp. Disse frihetsgradene kan også lettere måles av enkle sensorer som barometer, gyroskop og akselerometer.

Dybderegulator er praktisk å ha for piloten, da den for eksempel ved bruk av manipulatoren vil sørge for å holde ROV-en på en gitt dybde, selv med større forstyrrelser. Det skal også være mulighet for å sende inn en ønsket dybde fra toppsiden slik at ROV-en kjører til denne, uten noe behov for innspill fra pilotens kontroller.

ROV-en blir bygget slik at den har et naturlig oppretningsbidrag, som under normale omstendigheter vil sørge for at ROV-en holder en rull- og stampvinkel på tilnærmet null. Det kan dog komme forstyrrelser, for eksempel ved at manipulatoren strekkes

ut og/eller plukker opp noe, som vil føre til at ROV-en ikke lenger er i vater. Det er heller ikke et raskt system, og vil i tillegg ha en naturlig oversving. Det er derfor en stor fordel å regulere disse vinklene, for å gjøre systemet raskere, minimere oversving, samt kunne håndtere ubalanse i vektfordeling.

8.1.1 Behovet for regulator til manipulator

Som nevnt ovenfor implementeres regulatorer for å hjelpe piloten å manøvrere best mulig. Manipulatoren har fire ledd, der alle vil ha et holdemoment, som betyr dersom en stopper å gi et pådrag vil motoren holde seg i posisjonen den sist var i, så lenge lasten ikke blir for stor. Det er testet at holdemomentet er mer enn stort nok til å holde objektene i MATE-konkurransen, mer om dette i kapittel 10. Dersom de ulike motorene/leddene ikke hadde hatt et holdemoment, ville det vært praktisk med hver sin regulator for å kunne holde armen i en gitt posisjon. For å kunne implementere ulike regulatorer til manipulatoren ville det krevd ulike sensorer og/eller enkodere i hvert ledd for å måle vinkler og hastighet. En viktig faktor er også at de ulike leddene kan kjøres til endeposisjon uten at skade oppstår. Det konkluderes dermed med å ikke regulere disse leddene, da det ikke vil forenkle pilotens styring, samt at det ikke kreves for å holde stabil posisjon ellers. Det er heller ingen krav om å regulere manipulatoren i forhold til MATE-konkurransen.

8.2 Viktige momenter for modelleringen

Motorkarakteristikk

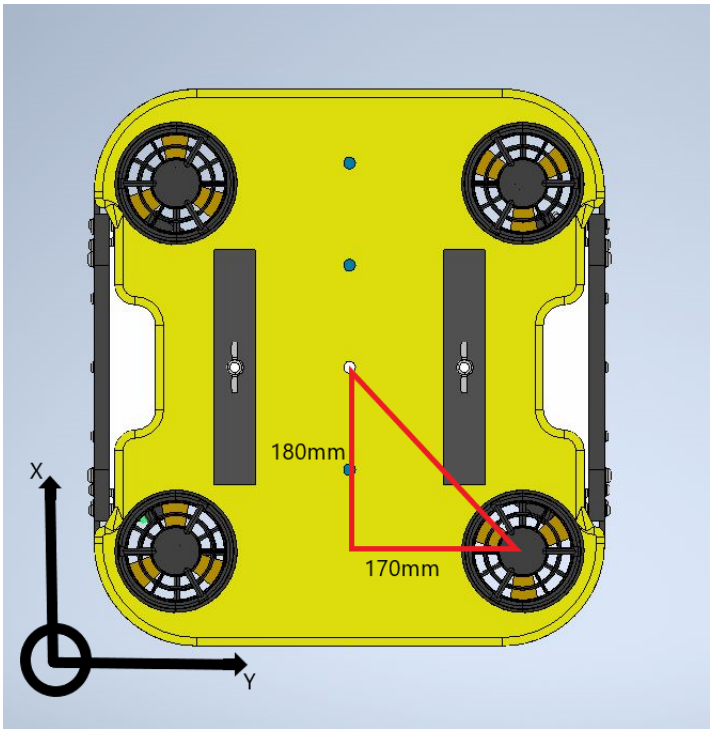
Det ble gjennomført en test av thrusternes stigningstall i Newton per prosent, både i fremover- og bakoverretning. Dette ble gjort som forklart i 3.4.1, og stigningstallet ble som forklart funnet til å være tilnærmet lineært, med en verdi på $\gamma = 0.78\ \frac{N}{\%}$. Flere thrustere ble testet og stigningstallet var tilnærmet lik for alle, gitt noe usikkerhet grunnet middelmådig testoppsett.

I ligning 8.1.1 vises totalkraften gitt av thrusterne i hivretning hvor u bestemmer kjøreretningen, og i ligning 8.1.2-3 vises dreiemomentet fra motorpådraget om x-aksen (rull) og y-aksen (stamp). Pådraget fra styrestikken, u, vil ha et verdiområde på [-1, 1]. Det er fire thrustere som gir pådrag på z-aksen og fire thrustere som gir pådrag for rotasjonsprosessene. Den totale kraften for pådraget til thrusterene blir dermed fireganget, som vist under:

$$\begin{aligned} &1. \quad F_u = 4\gamma u 100\% \ = \ 400 \cdot 0.78\ N \cdot u \\ &2. \quad \tau_{u,r} = 4\gamma r_{m,r} u 100\% \ = \ 400 \cdot 0.78\ N \cdot 0.17\ m \cdot u \\ &3. \quad \tau_{u,s} = 4\gamma r_{m,s} u 100\% \ = \ 400 \cdot 0.78\ N \cdot 0.18\ m \cdot u \end{aligned} \tag{8.1}$$

Hvor:
- 4 tallet er antall thrustere som gir pådrag.
- γ er stigningstallet til thrusterene i $\frac{N}{\%}$.
- u er pådraget som gis fra styrestikken i verdiområde [-1,1], uten enhet.
- $r_{m,r}$ er avstanden fra massesenteret til ROVen til punktet mellom to thrustere for

rullprosessen, vist på figur 8.1.
- $r_{m,s}$ er avstanden fra massesenteret til ROVen til punktet mellom to thrustere for stampprosessen, vist på figur 8.1.



Figur 8.1: Figuren viser avstanden fra massesenteret til ROVen til punktet mellom to thrustere. Denne avstanden er viktig for å finne dreiemomentet fra pådraget. For rullprosessen er denne avstanden 170 mm = 0.17 m og for stampprossen er denne avstanden 180 mm = 0.18 m. Dreiemomentet gitt fra to thrustere med samme pådrag vil være på punktet midt mellom de. Mål og bilde er hentet fra [36].

Dødtid

Det kommer til å være dødtid i ROV-en, som betyr at det vil gå en tid fra et signal blir sendt eller sensordata endres, til ønsket handling blir utført. Det består hovedsaklig av tiden det tar fra en sensorverdi endres til denne er sendt, mottatt, behandlet og utført av reguleringssystemet, samt tiden fra et pådrag gis på styrestikken til ønsket pådrag er sendt til ønsket thruster.

Disse tidsforsinkelsene har blitt beregnet til et omtrentlig estimat ved hjelp av sendefrekvensen fra sensorkortet og toppside. Sensordata blir sendt fra sensorkortet til reguleringssystemet ved bruk av SPI-kommunikasjon. Det vil ikke være en stor tidsforsinkelse på selve signalet, men i sammarbeid med sensorgruppen ble det bestemt at dataene skal sendes med en frekvens på 100 Hz, som dermed gir en tidsforsinkelse på $\tau_{sensor} = 10\ ms$. Normalt stabil sendefrekvens fra styringsprogrammet på toppsiden til styringssystemet på ROV-en, er også i følge kommunikasjonsgruppen [59] 100 Hz, som igjen gir en tidsforsinkelse på $\tau_{kommunikasjon} = 10\ ms$.

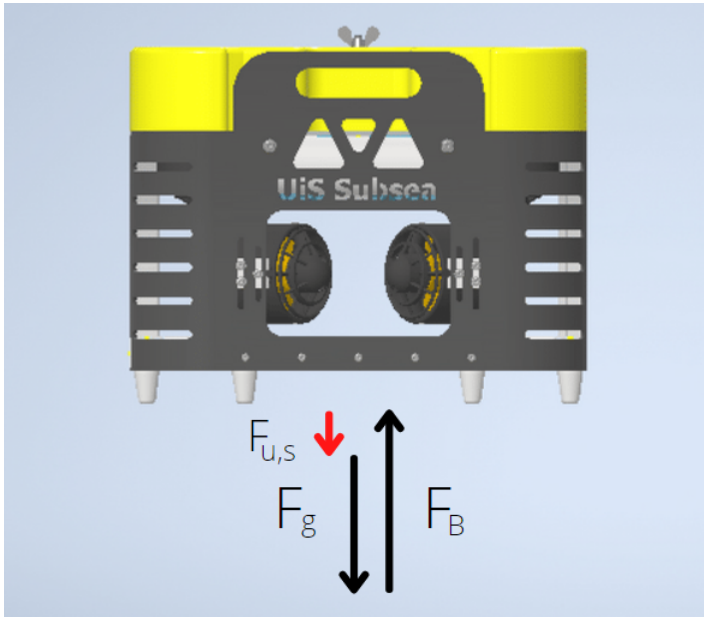
Dødtiden består av $\tau_{kommunikasjon} + \tau_{sensor}$, som utgjør at systemet har en total dødtid $\tau = 20\ ms$. Denne dødtiden vil bli implementert videre i den matematiske

modellen og simuleringer for reguleringssystemet.

8.3 Dybdeprosessen

Oppdrift når ROVen er i ro

Rammegruppen [36], som har ansvar for rammeverket til ROV-en, har oppgitt at det kommer til å være en naturlig konstant oppdrift. Det vil si at når ROV-en står i ro under vann vil den flyte litt, men veldig lite, oppover. Dette er fordi ROV-ens oppdriftskraft F_B er litt større en gravitasjonskraften F_g, som vist på figur 8.2. Det er oppgitt at ROV-en har en lengde d = 0.545m, bredde w = 0.500m og høyde h = 0.390m. ROV-en er dog delvis hul, og en kan dermed ikke regne volum basert på disse målene. Rammegruppen har et mål om å beregne volumet eksakt, men per skrivende stund er dette ikke klart.



Figur 8.2: Summen av alle krefter vises på figuren. Oppdriftskraften F_B er litt større enn F_g som gjør at summen av ROVens naturlige krefter gir oppdrift. $F_{u,s}$ er kraften som blir tilført for å nulle ut denne kraften.

For at ROV-en skal holde en gitt dybde må det derfor bli gitt et stasjonært pådrag til thrusterene, $F_{u,s}$, slik at summen av kreftene som påvirker ROV-en i vertikal retning er lik null. Ved bruk av en PI- eller PID-regulator, som blir forklart i kapittel 9, vil I-leddet kompensere for dette. Videre for den dynamiske delen vil det bli gitt et dynamisk pådrag, $F_{u,d}$.

Oppdriftskraften F_B er avhengig av tettheten til vann, ρ, volumet til ROVen, V_{ROV}, og tyngdeakselerasjonen, g, og beregnes ved formelen for oppdriftskraft [60]:

$$\begin{aligned} F_B &= \rho \cdot V_{ROV} \cdot g \\ F_B &= 1000 \ \tfrac{kg}{m^3} \cdot V_{ROV} \ m^3 \cdot 9.81 \ \tfrac{m}{s^2} \end{aligned} \tag{8.2}$$

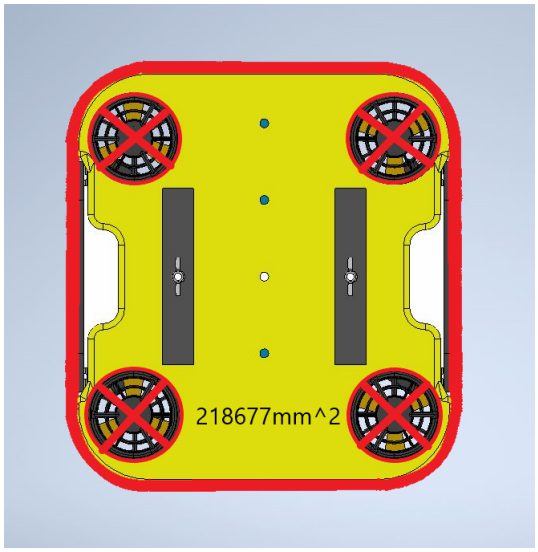
Oppdriftskraften kan altså ikke beregnes nøyaktig da vi ikke har volumet, men av rammegruppen er det gitt at denne vil ha veldig liten påvirkning for ROVens dynamikk, og blir derfor sett vekk fra videre i modelleringen for dybdeprosessen.

Vannmotstand langs z-aksen

I alle frihetsgrader vil ROV-en oppleve en en kraft som virker *mot* ROV-ens kjøreretning, nemlig vannmotstanden F_D. Kraften er avhengig av den kvadrerte hivfarten i kjøreretningen. Det vil si at når når ROV-en ikke har en hivfart, altså at $v_z = 0$ vil også F_D være lik 0. Når ROV-en akselererer ved et gitt pådrag vil $F_{u,d}$ - $F_D \neq 0$, men når ROV-en har oppnådd konstant hivfart vil $F_{u,d} = F_D$. Ved maks pådrag vil det etter omtrent 2 sekund være en konstant hivfart som fører til at $F_{u,d} = F_D$, dette er vist i figur 8.4. Formelen for vannmotstand er vist i ligningen under, hentet fra [61].

$$F_D = \frac{1}{2}\rho C_d A |v_z| v_z \tag{8.3}$$

Hvor:
- ρ er tettheten til omgivelsen objektet er i, vann har tetthet på 1000 kg/m^3.
- C_d er dragkoeffisienten.
- A er arealet til ROV-en som treffer vann i hiv-retning. A = 0.219 m^2 vist på figur 8.3.
- v_z er hivfarten til ROVen.



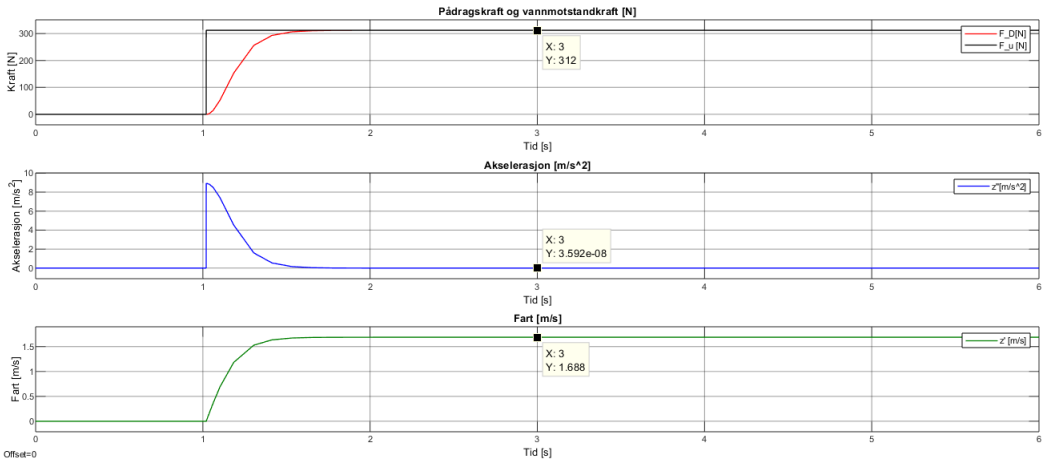
Figur 8.3: Figuren viser overflatearealet sett ovenfra. Det totale overflatearealet er $A = 218677mm^2 \approx 0.219\ m^2$, gitt av rammegruppen [36].

C_d er en koeffisient som er veldig kompleks å fastslå matematisk. Ifølge NASA [62] består dragkoeffisienten av veldig mange faktorer som er utilgjengelig for oss. Det går an å tilnærme C_d eksperimentelt, men det er også en lang og krevende prosess, og det er derfor gunstig å gjøre en tilnærming basert på ferdige tabeller for forskjellige former. ROV-ens utforming ligger et sted mellom en rektangulær boks og en kvadratisk boks (utformingen er vist i figur 8.9), og ifølge tabell over dragkoeffisienter

[63] har den dermed en verdi mellom 0.8 til 2.1, der jo mer kvadratisk utformingen er jo lavere er C_d. Ettersom ROVens utforming er omtrent en kvadratisk boks gjøres det en tilnærming sammen med rammegruppen [36], at $C_d \approx 1$.

I formelen for vannmotstand er egentlig hastigheten kvadrert uten hensyn til kjøreretningen, slik som dette: v_z^2. Problemet med dette er at da vil alltid kraften være positiv uansett kjøreretning. Det er ønskelig å skille mellom kjøreretningene ved å beholde det korrekte fortegnet til v_z. Da vil også $F_{u,d}$ - F_D alltid ha samme sammenheng som beskrevet over. Derfor er v_z^2 gjort om til $|v_z|v_z$.

I figur 8.4 vises sammenhengen mellom pådragskraften og vannmotstandskraften som forklart tidligere. Når ROV-en akselererer vil $F_{u,d}$ - $F_D \neq 0$, men ettersom F_D øker vil akselerasjonen avta, og det oppnås stasjonær hivfart som dermed fører til at $F_{u,d} = F_D$. Den maksimale hivfarten simuleres til å være 1.688 $\frac{m}{s}$ når maksimalt pådrag gis som vist på figuren under.

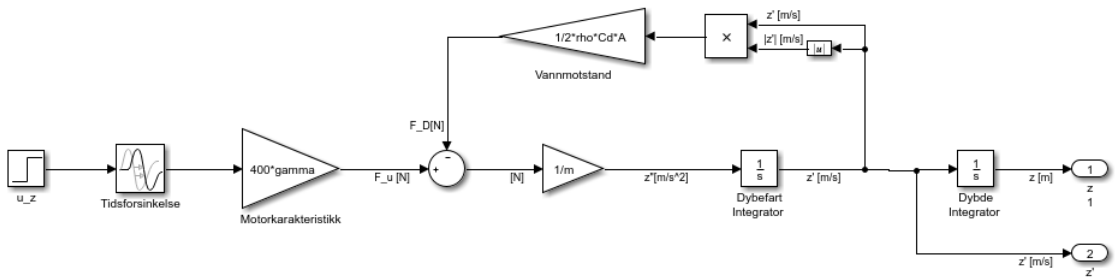


Figur 8.4: Ved å gi et sprang med maksimalt pådrag fra u = 0 til u = 1 i Simulink modellen som er vist over kan en simulere ulike krefter, akselerasjon og fart.

På figuren ser en at når akselerasjonen avtar og det er oppnådd konstant hivfart er $F_{u,d} = F_D = 312\ N$. Dette er altså den maksimale kraften til ROV-en ved et maksimalt pådrag på fire thustere i dybderetning.

8.3.1 Matematisk modell av dybdeprosessen

I figur 8.5 vises blokkskjema for dybdeprosessen. Blokkskjemaet består av de ulike faktorene og kreftene som er tatt hensyn til i delkapitlene over, og blir utgangspunktet for modelleringen og reguleringssystemet for dybdeprosessen.



Figur 8.5: Blokkskjema i Simulink for simulering av dybdeprosessen.

For den dynamiske delen av dybdeprosessen kan kraftbalansen brukes på kreftene som påvirker ROV-en på z-aksen, $F_{u,d}$ (ligning 8.1) og F_D (ligning 8.3). Utregninger videre i kapittelet baseres på [64][65]. Ved bruk av kraftbalansen kan summen av kreftene skrives som:

$$\sum F = m\ddot{z} = F_{u,d} - F_D = 400\gamma u - \frac{\rho C_d A}{2}|v_z|v_z \tag{8.4}$$

ligningen kan skrives om til to 1.ordens differensialligninger:

$$\begin{aligned} \dot{z} &= v_z \\ \dot{v}_z &= \frac{400\gamma u}{m} - \frac{\rho C_d A}{2m}|v_z|v_z \end{aligned} \tag{8.5}$$

Modellen i ligning 8.4 er ulineær på grunn av $|v_z|v_z$, og det er derfor gunstig å linearisere modellen rundt et relevant arbeidspunkt. Ved hjelp av en linearisert modell kan en da finne de dynamiske egenskapene rundt det relevante arbeidspunktet. $\dot{v}_z$ kan skrives som en funksjon som er avhengig av farten v_z og pådraget u, $\dot{v}_z(t) = f(v_z(t), u(t))$. Ved å gi en liten endring Δ rundt et arbeidspunkt A, kan vi linearisere modellen ved å partiellderivere funksjonen $\frac{\partial f}{\partial(v_z,u)}\Big|_A$:

$$\begin{aligned} \Delta\dot{z}(t) &= \Delta v_z(t) \\ \Delta\dot{v}_z(t) &= \frac{\partial f}{\partial u}\Big|_A \Delta u(t) + \frac{\partial f}{\partial vz}\Big|_A \Delta v_z(t) \end{aligned} \tag{8.6}$$

$$\begin{aligned} \frac{\partial f}{\partial u}\Big|_A &= \frac{\partial}{\partial u}\Big(\frac{400\gamma}{m}\ u(t) - \frac{\rho C_d A}{2m}|v_z(t)|v_z(t)\Big)\Big|_{u=u_A} = \frac{400\gamma}{m} \\ \frac{\partial f}{\partial v_z}\Big|_A &= \frac{\partial}{\partial v_z}\Big(\frac{400\gamma}{m}\ u(t) - \frac{\rho C_d A}{2m}|v_z(t)|v_z(t)\Big)\Big|_{v_z=v_{z,A}} = -\frac{\rho C_d A}{2m}|v_z(t)| \end{aligned} \tag{8.7}$$

Ved å sette inn de partiellderiverte fra ligning 8.7 i ligning 8.6 får en:

$$\begin{aligned} \Delta\dot{z}(t) &= \Delta v_z(t) \\ \Delta\dot{v}_z(t) &= \frac{400\gamma}{m}\ \Delta u(t) - \frac{\rho C_d A|v_{z,A}|}{2m}\Delta v_z(t) \end{aligned} \tag{8.8}$$

Ved å deretter Laplacetransformere de to lineariserte 1.ordens ligningene kan en finne overføringsfunksjonen til modellen:

$$\begin{aligned} &\mathcal{L}\Big\{\Delta\dot{z}(t) = \Delta v_z(t)\Big\} \\ &\mathcal{L}\Big\{\Delta\dot{v_z}(t) = \frac{400\gamma}{m}\ \Delta u(t) - \frac{\rho C_d A|v_{z,A}|}{2m}\Delta v_z(t)\Big\} \end{aligned} \tag{8.9}$$

$$\Downarrow$$

$$\begin{aligned} &s\Delta z(s) = \Delta v_z(s) \\ &s\Delta v_z(s) = \frac{400\gamma}{m}\ \Delta u(s) - \frac{\rho C_d A|v_{z,A}|}{2m}\Delta v_z(s) \end{aligned} \tag{8.10}$$

For å finne overføringsfunksjonen løses nederste ligning i 8.10 for $\frac{\Delta v_z(s)}{\Delta u(s)}$, og deretter ved hjelp av sammenhengen $s\Delta z(s) = \Delta v_z(s)$, finnes overføringsfunksjonen for $\frac{\Delta z(s)}{\Delta u(s)}$:

$$\begin{aligned} &s\Delta z(s) = \Delta v_z(s) \\ &\Big(s + \frac{\rho C_d A|v_{z,A}|}{2m}\Big)\Delta v_z(s) = \frac{400\gamma}{m}\ \Delta u(s) \longrightarrow \frac{\Delta v_z(s)}{\Delta u(s)} = \frac{\frac{400\gamma}{\rho C_d A|v_{z,A}|}}{\Big(\frac{m}{\rho C_d A|v_{z,A}|}s + 1\Big)} \end{aligned} \tag{8.11}$$

Til slutt fjernes Δ-notasjonen og dødtiden legges med i overføringsfunksjonen som $e^{-\tau s}$. Med dette ender vi opp med to overføringsfunksjoner, der den første ligningen er forholdet mellom fart og pådrag i z-retning, og den andre er forholdet mellom dybde og pådrag i z-retning, vist i ligningene under:

$$\begin{aligned} H_{p,v_z}(s) &= \frac{v_z(s)}{u(s)} = \frac{\frac{400\gamma}{\rho C_d A|v_{z,A}|}}{\Big(\frac{m}{\rho C_d A|v_{z,A}|}s + 1\Big)}e^{-\tau s} \\ H_{p,z}(s) &= \frac{z(s)}{u(s)} = \frac{\frac{400\gamma}{\rho C_d A|v_{z,A}|}}{s\Big(\frac{m}{\rho C_d A|v_{z,A}|}s + 1\Big)}e^{-\tau s} \end{aligned} \tag{8.12}$$

Ved bruk av standardformen for et 1.ordens system med tidsforsinkelse i ligning 8.13, kan en sammenligne standardformen med overføringsfunksjonen og finne prosesskonstantene K_z og T_z.

$$H_p(s) = \frac{K}{Ts + 1}e^{-\tau s} \tag{8.13}$$

Prosessforsterkningen er avhengig av motorkarakteristikk, og et høyere stigningstall γ ville gitt større forsterkning. Dersom overflatearealet hadde vært mindre slik at

det er mindre areal som får vannmotstand ved en viss hivfart $v_{z,A}$ ville også forsterkningen blitt større. Disse variablene ville gjort at en pådragsendring ville gitt en høyere hivfart.

$$K_z = \frac{400\gamma}{\rho C_d A|v_{z,A}|} = \frac{400 \cdot 0.78\ N}{1000\ kg/m^3 \cdot 1 \cdot 0.219\ m^2|v_{z,A}|\ m/s} = \frac{312}{219|v_{z,A}|}\ \frac{m}{s} \tag{8.14}$$

Ofte ønskes det en lav tidskonstant slik at modellen når den stasjonære verdien raskest mulig ved et gitt pådrag. Fra ligning 8.15 kan en se at et objekt med mindre masse vil ha mulighet for raskere tidskonstant. Dersom objektet har en lav arbeidsfart vil det ta lengre tid å komme opp til den stasjonære farten, ettersom $v_{z,A}$ er et lavere tall.

$$T_z = \frac{m}{\rho C_d A|v_{z,A}|} = \frac{35\ kg}{1000\ kg/m^3 \cdot 1 \cdot 0.219m^2 \cdot |v_{z,A}|\ m/s} = \frac{35}{219|v_{z,A}|}\ s \tag{8.15}$$

Et arbeidspunkt blir definert som et punkt hvor alle deriverte er lik 0, også kalt et likevektspunkt. Det vil si at $\ddot{z} = 0$. Arbeidspunktet for farten er avhengig av pådraget som også er i et arbeidspunkt, u_A, og derfor er det hensiktsmessig og bruke u = u_A. Med dette tatt i betraktning kan arbeidspunktet til $v_{z,A}$ finnes ved:

$$\begin{gathered} 0 = \frac{400\gamma u_A}{m} - \frac{\rho C_d A}{2m}|v_{z,A}|v_{z,A} \\ v_{z,A} = \sqrt{\frac{2 \cdot 400\gamma u_A}{\rho C_d A}} \ \longrightarrow\ v_{z,A} = \sqrt{2.849 \cdot u_A} \end{gathered} \tag{8.16}$$

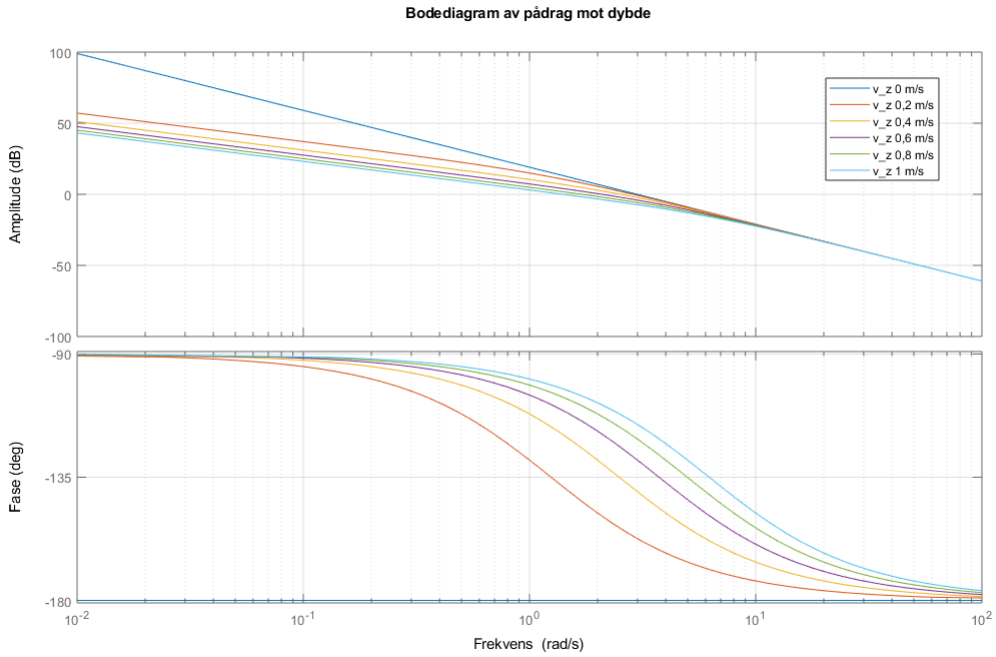
Dette vil si at farten er avhengig av pådraget som gis, og den maksimale stasjonære hastigheten vil oppnås når $u_A = 1$. Det utgjør $v_{z,A} = \sqrt{2.849 \cdot 1} = 1.688\frac{m}{s}$. Dette er det samme som ble vist i tidligere ved simulering i Simulink på figur 8.4. Dersom $u_A = 1$ ville prosessforsterkningen blitt $K_z = 0.843$ og tidskonstanten $T_z = 0.095$

8.3.2 Simulering av dybdeprosessen

Det er praktisk å simulere den matematiske modellen i Simulink for å få et bedre bilde og visualisere oppførselen til de ulike prosessene. Simulink er kun en kompleks kalkulator, og derfor skal det ikke være noe forskjell fra det teoretiske og simuleringene i Simulink. Det er mulig å bruke hjelpemiddelet i Simulink kalt *Linear Analysis Tool* som et annet alternativ til å se om modellen er ulineær og lineærisere den. Dette gjøres i Simulink modellen vist i figur 8.5 ved å høyreklikke på inngangsignalet *u_z* og trykke *Linear Analysis* $\longrightarrow$ *open-loop input* og å høyreklikke på utgangen for dybde, *z* og trykke *Linear Analysis* $\longrightarrow$ *open-loop output*. Derretter må en sette modellen i et arbeidspunkt for dybdefarten $v_{z,A}$, dette gjøres ved å endre initialverdien i *Dybdefart Integrator*. I figur 8.6 kan en se en Bode-plott analyse i *Linear Analysis Tool* vinduet ved ulike arbeidspunkt for $v_{z,A}$.

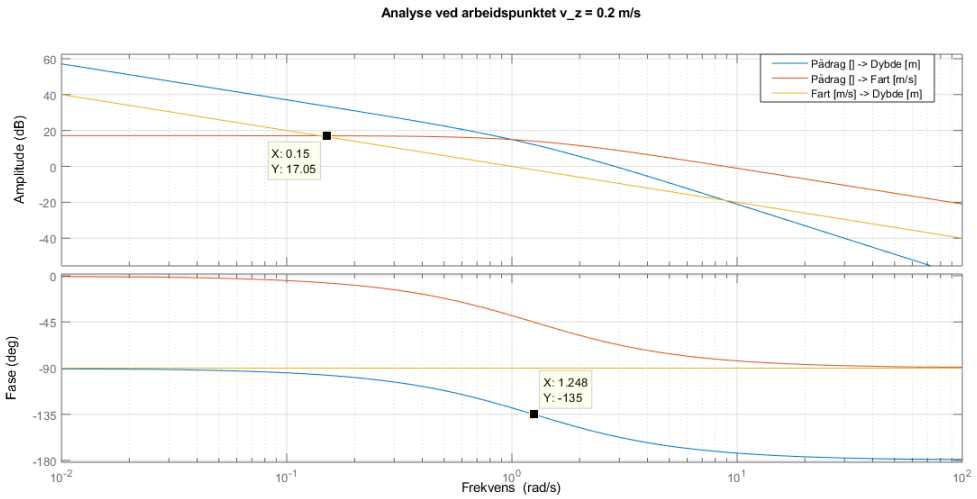
Figuren kan tolkes utifra de ulike arbeidspunktene og det kan kjapt konkluderes med at modellen er ulineær. Dette er fordi ved ulike hastigheter er den stasjonære

forsterkning og knekkfrevensen forskjellig i de ulike arbeidspunktene. Desto høyere fart desto mer minker den stasjonære forsterkning og fasen øker. En kan observere at når $v_{z,A} = 0\ \frac{m}{s}$ er også fasen -180° som vil si at prosessen er dobbeltintegrerende. Dette har en sammenheng med at vannmotstanden er avhengig av den kvadrerte farten og vil være lik null når farten også er null.



Figur 8.6: Figuren viser de ulike frekvensresponsene ved ulike hastigheter. Modellen ble kjørt i ulike hastigheter for å illustrere at modellen var ulineær. Desto høyere fart desto lavere blir den stasjonære forsterkningen (startpunktet går nedover for høyere hastigheter) og fasen blir høyere (-3 dB punktet forskyver seg mot en høyere frekvens).

Etter å ha bekreftet at modellen er ulineær kan den lineæriseres rundt et valgt arbeidspunkt for $v_{z,A}$, i dette tilfellet velges $v_{z,A} = 0.2\ \frac{m}{s}$ som kan nås med et pådrag $u_A \approx 0.014$. Det utføres deretter en ny Bode-plott-analyse i figur 8.7 ved det konkrete arbeidspunktet, og en kan se oppbygningen til de ulike komponentene i modellen. Signalet vist i blått er frekvensresponsen ved $v_{z,A} = 0.2\ \frac{m}{s}$, likt som i figur 8.6. Signalet vist i rødt viser den stasjonære forsterkningen som består av vannmotstanden og massen. Det gule signalet viser en enkel integrator som integrerer fra dybdefart $[\frac{m}{s}]$ til dybde [m].



Figur 8.7: Figuren viser oppbygningen av de ulike komponentene. Øverste i høyre hjørne vises det hvordan *Linear Analysis Tool* blir brukt, der det til venstre for pilen blir valgt som *open-loop input*, og det til høyre for pilen blir valgt som *open-loop output*.

Ut ifra denne analysen kan en nå finne prosesskonstantene for arbeidspunktet $v_z = 0.2\ \frac{m}{s}$. Prosessforsterkningen kan finnes ved den stasjonære forsterkningen gitt ved det øverste datatipset for 1.ordens systemet (rød), $K_{1,dB} = 17.05\ dB$. Den gule er integratorforsterkningen, $K_{2,dB} = 0\ dB$. Disse forsterkningene kan gjøres om fra dB til en forsterkningskonstant med formelen $K_{dB} = 20log(K) \longrightarrow K_z = 10^{\frac{K_{dB}}{20}}$. Knekkfrekvensen finnes ved punktet som er -3 dB under den stasjonære forsterkningen, altså ved 14.05 dB som tilsvarer en fase lik -135°. Fra det nederste datatipset kan en finne knekkfrekvensen $\omega_c = 1.248\ \frac{rad}{s}$. Fra dette kan prosessforsterkningene finnes:

$$\begin{aligned} K_1 &= 10^{\frac{K_{1,dB}}{20}} = 10^{\frac{17.05}{20}} = 7.12\ \tfrac{m}{s} \\ K_2 &= 10^{\frac{0}{20}} = 1.00\ \tfrac{m}{s} \\ T_z &= \frac{1}{\omega_c} = \frac{1}{1.248} \approx 0.80\ s \end{aligned} \tag{8.17}$$

Basert på standardformen til et 2.ordens system og konstantene i ligning 8.17 kan en finne overføringsfunksjonen ved arbeidspunktet $v_{z,A} = 0.2\ \frac{m}{s}$:

$$\begin{aligned} H_{p,z} &= \frac{K_2}{s} \cdot \frac{K_1}{T_z s + 1} = \frac{K_2}{s} \cdot \frac{K_1}{\frac{1}{\omega_c} s + 1} \\ H_{p,z} &= \frac{1.00}{s} \cdot \frac{7.12}{0.80s + 1} = \frac{7.12}{s(0.80s + 1)} \end{aligned} \tag{8.18}$$

Som nevnt i innledningen til dette delkapittelet er Simulink kun en kompleks kalkulator som regner det samme som ble gjort teoretisk. For å sammenligne prosesskonstantene fra lineæriseringen av den matematiske modellen fra ligning 8.14-15 med

lineæriseringen ved bruk Simulink innsatt arbeidspunktet $v_{z,A} = 0.20\ \frac{m}{s}$:

$$\begin{aligned} K_z\Big|_{v_z,A=0.20\frac{m}{s}} &= \frac{312}{219 \cdot 0.20} = 7.12\ \frac{m}{s} \\ T_z\Big|_{v_z,A=0.20\frac{m}{s}} &= \frac{35}{219 \cdot 0.20} = 0.80\ s \end{aligned} \tag{8.19}$$

Det konkluderes til slutt med at det er lagt et godt grunnlag med den matematiske modellen både teoretisk og ved bruk av Simulink til å finne overføringsfunksjonene og prosesskonstantene. Videre i kapittel 9 skal dette grunnlaget brukes til å simulere reguleringssystemet og finne reguleringsparametrene med den gitte modellen for dybdeprosessen i Simulink.

8.4 Rull- og stampprosessene

Modellen og fremgangsmåten vil være omtrent lik for både rullprosessen og stampprosessen. For å ikke repetere en hel prosess som kun består av en enkel variabelendring, vil det hovedsaklig bli gjennomgått for rullprosessen, mens stampprosessen vil bli nevnt og oppsummert underveis. Variabelendringer vil også bli vist i situasjonene det gjelder.

I fremgangsmåten for å finne overføringsfunksjonen vil det bli brukt momentbalansen for dreiemomentene [64].Rullprosessen består av rullvinkel[1] ϕ med rullrate p[2], og stamprosessen består av stampvinkel θ med stamprate q. Summen av dreiemomentene som påvirker ROVen består av:

$$\sum \tau = J\ddot{\phi} = \tau_u - \tau_o - \tau_D \tag{8.20}$$

Hvor:
- J er treghetsmomentet til ROVen.
- $\ddot{\phi}$ er den dobbel deriverte av rullvinkelen lik akselerasjon.
- τ_u er dreiemomentet fra pådraget til motorene.
- τ_o er dreiemomentet fra oppretningsbidraget til ROVen.
- τ_D er dreiemomentet fra vannmotstanden som virker *mot* rotasjonsbevegelser.

Hver av disse dreiemomentene vil bli forklart dypere videre. τ_u er allerede forklart i delkapittelet om motorkarakteristikk, 8.2.

ROVens treghetsmomenter, J

Treghetsmoment er ROV-ens evne til å motstå endring i rotasjonshastighet om en akse. Utformingen av ROVen som er et rektangel bestående av lengden d = 0.545 m, bredden w = 0.500 m, høyden h = 0.390 m og massen, m = 35 kg. Treghetsmomentet kan slås opp i tabell og ligningene for rektangelet kan finnes fra [66],

[1] ϕ er vinkelen ROV-en er rotert om x-aksen og θ er vinkelen ROV-en er rotert om y-aksen.
[2] p er hastigheten ROV-en roterer om x-aksen og q er hastigheten ROV-en roterer om y-aksen.

treghetsmomentet for de ulike frihetsgradene vises i ligning 8.21:

$$J_d = \frac{1}{12}m(w^2 + h^2) = \frac{1}{12} \cdot 35\ kg(0.500^2\ m + 0.390^2\ m) = 1,173\ kg \cdot m^2$$

$$J_w = \frac{1}{12}m(d^2 + h^2) = \frac{1}{12} \cdot 35\ kg(0.545^2\ m + 0.390^2\ m) = 1,310\ kg \cdot m^2 \quad (8.21)$$

$$J_h = \frac{1}{12}m(w^2 + d^2) = \frac{1}{12} \cdot 35\ kg(0.500^2\ m + 0.545^2\ m) = 1,596\ kg \cdot m^2$$

Hvor:
- J_d er treghetsmomentet for rullvinkelen (om x-aksen).
- J_w er treghetsmomentet for stampvinkelen (om y-aksen).
- J_h er treghetsmomentet for girvinkelen (om z-aksen).

J_d er dermed relevant for rullprosessen, og J_w er relevant for stampprosessen. For treghetsmomentet for rullprosessen brukes bredden og høyden, mens for stampprosessen brukes lengden og høyden. Forskjellen er en konstant som lett kan byttes ut videre for å få den riktige modellen for den spesifikke prosessen.

Dreiemoment fra oppretningsbidraget, τ_o

For at det skal være et dreiemoment τ_o som gir et naturlig oppretningsbidrag, blir ROV-en designet slik at ROV-ens massesenter ligger under volumsenteret. Med dette designet vil ROV-en rotere om volumsenteret slik at rull- og stampvinkelen ≈ 0 og ROV-en ligger i vater med vannet. Dette illustreres på figur 8.8. Det er avstanden mellom masse- og volumsenter som bestemmer hvor stort det naturlige oppretningsbidraget er. Større avstand gir større dreiemoment, og mindre avstand gir mindre dreiemoment.

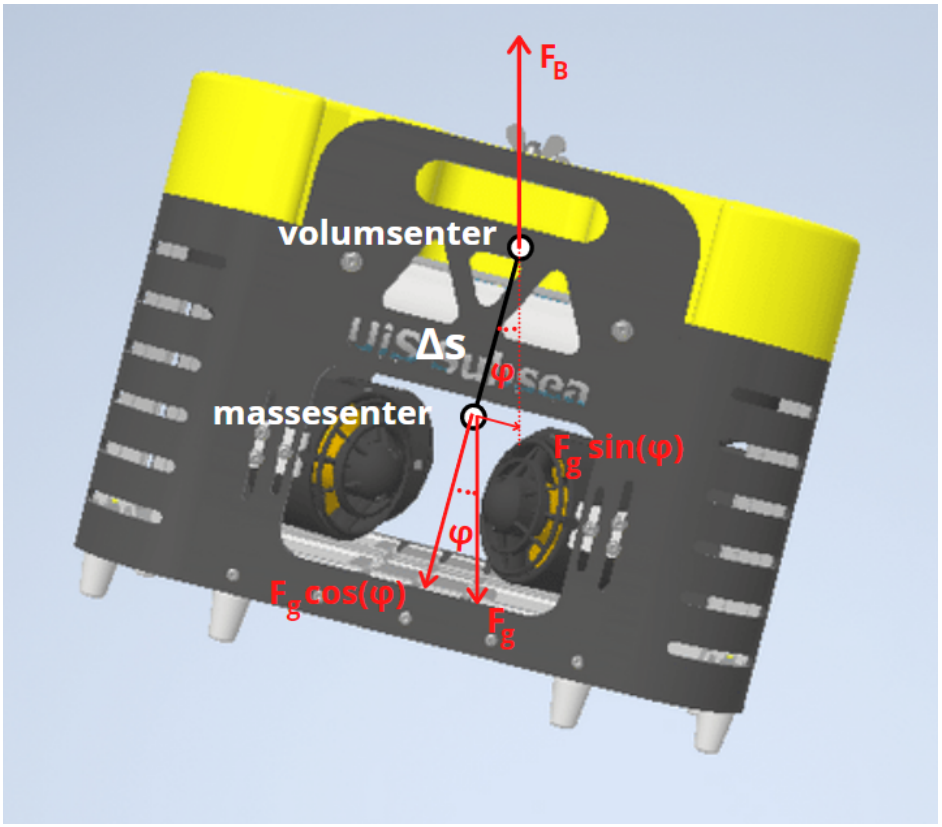
Dreiemomentet fra oppretningsbidrag kan ledes fra den generelle ligningen for dreiemoment, gitt av kraften F multiplisert med en momentarm med lengde r, multiplisert med sinus til vinkelen mellom kraften og momentarmen [67]:

$$\tau = F \cdot r \cdot sin(\phi) \quad (8.22)$$

I dette tilfellet er det gravitasjonskraften F_g som påvirker oppretningsbidraget. Denne består av to komponenter; gravitasjonskraften multiplisert med cosinus til vinkelen ϕ og gravitasjonskraften multiplisert med sinus til vinkelen ϕ. Det er antatt at det ikke er noe translatorisk bevegelse, som fører til at $F_g cos(\phi) = 0$. Videre vil det med det kun bli brukt $F_g sin(\phi)$.

Sammenhengen mellom ligningen over og dreiemomentet for oppretningsbidraget illustreres på figuren under. Kraften som påvirker oppretningsbidraget er gravitasjonskraften F_g. Armen i dette tilfellet er avstanden mellom massesenter og volumsenter, Δs og sinus til rullvinkelen ϕ (for stampprosessen vil dette være for sinus til stampvinkelen, θ), som illustrert på figuren under. Det gjør at ligningen for dreiemomentet τ_o blir:

$$\tau_o = F_g \Delta s sin(\phi) = mg\Delta s sin(\phi) \quad (8.23)$$

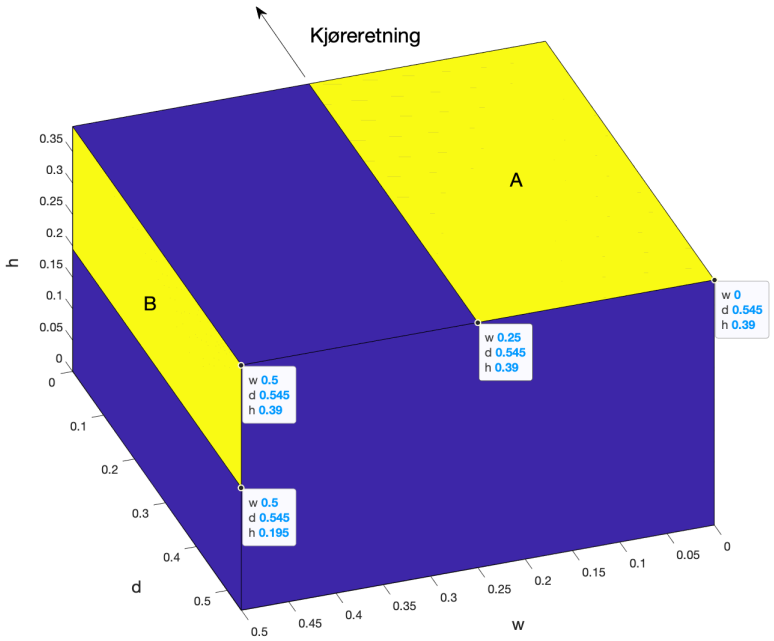


Figur 8.8: Figuren viser hvordan dreiemomentet fra oppretningskraften er bygget opp. Δs er ikke så stor i virkeligheten, men er en overdrevet størrelse for å vise hensikten i figuren. Det er denne avstanden som hovedsaklig bestemmer hvor stor oppretningskraften vil være.

Det kan være en fordel å ta i bruk små-vinkel teoremet for $sin(\phi)$ slik at dreiemomentet fra oppretningsbidraget blir forenklet for den matematiske modellen [68]. Ved bruk at dette teoremet kan en fastslå for rullprosessen at for små vinkler er $sin(\phi) \approx \phi$ og for stampprosessen er $sin(\theta) \approx \theta$. For å kunne bestemme hva som er små nok vinkler brukes et relativt avvik for å fastslå hvor grensen går før teoremet ikke er gyldig. Det relative avviket kan variere stort, men for å få en mer nøyaktig grense velges det et relativt avvik på 1 %, som vil gi en god tilnærming. Den maksimale vinkelen før teoremet ikke er gyldig vil da være $\approx 13.99° = 0.2442\ rad$, dette er gitt av [68].

Dreiemoment fra vannmotstanden, τ_D

På figur 8.9 vises utformingen til ROV-en. Områdene markert i gult er hvor vannmotstanden påvirker ROV-en ved rotasjon mot klokken om kjøreretning. Det er totalt fire av seks sider på ROV-en som blir påvirket av vannmotstand, på figuren vises to av de sett ovenfra, henholdvis de gule områdene merket med **A** og **B**. De to resterende sidene vil være ekvivalent til område A og B, men på under- og høyresiden av ROV-en, og derfor multipliseres de to gule feltene med 2 for å få alle fire sidene med i utregningene videre.



Figur 8.9: Figuren viser utformingen til ROVen, med lengde d=0.545m, bredde w = 0.500m og høyde h = 0.390m. Sett ovenfra dreier ROV-en om kjøreretning mot klokken. De gule rektanglene A og B er overflatene som treffer vann.

Dreiemomentet fra vannmotstanden for et rektangel kan ledes ut fra kryss-produktet mellom kraft og arm vist i ligning 8.24 [67]. Armen i dette tilfellet vil være annerledes for hver av de fire sidene. For det gule området markert med A, vil lengden av armen være halvparten av bredden, mens for det gule området markert med B, vil lengden av armen være halvparten av høyden. Kraften vil være lik F_g som vist i figur 8.8, og denne kraften står vinkelrett på armen slik at $sin(\theta = \frac{\pi}{2}) = 1$, og derfor kan ligningen forenkles som vist under:

$$\tau_D = F \times r = F \cdot r \cdot sin(\theta) = F \cdot r \tag{8.24}$$

Først vises kun utregningen for vannmotstanden som treffer det gule rektangel merket med **A** i ligning 8.25. Ettersom vannmotstanden er lik for oppsiden og undersiden av ROV-en, ganges den med 2. For å finne det totale dreiemomentet for hele rektangel A, må en summere opp hvert lille areal dA som blir truffet av vannmotstanden dF på det gule området. Her er F kraften for vannmotstanden lik den som ble brukt for dybdeprosessen, hentet fra [61] og armen for det gule området A lik bredden w,

hvor armen går fra 0 til halvparten av den totale bredden.

$$\tau_{D_A} = 2\int_{w'=0}^{w'=w/2} d\tau \;=\; 2\int_{w'=0}^{w'=w/2} w' \cdot dF \;=\; 2\int_{w'=0}^{w'=w/2} w' \cdot \Big(\frac{1}{2}\rho C_d (p \cdot w')^2 dA\Big)$$

$$\tau_{D_A} = \rho C_d |p| p \int_{w'=0}^{w'=w/2} (w')^3 \; dA \;=\; \rho C_d |p| p \int_{w'=0}^{w'=w/2} (w')^3 \cdot d \; dw'$$

$$\tau_{D_A} = d\rho C_d |p| p \int_{w'=0}^{w'=w/2} (w')^3 \Big|_{w'=0}^{w'=w/2} \;=\; d\rho C_d |p| p \cdot \frac{1}{4} \cdot \frac{w^4}{2}$$

$$\tau_{D_A} = \frac{d}{8}\rho C_d |p| p w^4 \tag{8.25}$$

Hvor:
- ρ er tettheten til omgivelsen objektet er i, vann har tetthet på 1000 $\frac{kg}{m^3}$.
- C_d er drag koeffisienten til ROVen.
- A er arealet til overflaten som treffer vannet. $A = d \cdot w \rightarrow dA = d \; dw$.
- p er rullraten til ROVen. Gjort om fra $p^2 = |p|p$ slik at p bestemmer kraftretningen.
- w er bredden på ROVen lik 0.500 m.
- d er lengden på ROVen lik 0.545 m.

Dette er da altså dreiemomentet fra vannmotstanden for oppsiden og undersiden. For å finne dreiemomentet fra vannmotstanden for høyre- og venstresiden, må det gule rektangelet merket med **B**, ganges med 2. Videre brukes ligning 8.25, men der $A = d \cdot w \rightarrow dA = d \; dw$ nå byttes ut med $A = d \cdot h \rightarrow dA = d \; dh$. Dette er fordi nå skal hvert lille areal dA som blir truffet av vannmotstanden dF på det gule rektangelet merket B summeres opp. Med lik fremgangsmåte ender en opp med ligning 8.26 vist under:

$$\tau_{D_B} = \frac{d}{8}\rho C_d |p| p h^4 \tag{8.26}$$

Det totale dreiemomentet fra vannmotstanden for hele ROV-en ved rull blir da $\tau_{D_A} + \tau_{D_B}$ og en ender opp med:

$$\begin{aligned} \tau_{D,r} &= \frac{d}{8}\rho C_d |p| p w^4 + \frac{d}{8}\rho C_d |p| p h^4 \\ \tau_{D,r} &= \frac{d}{8}\rho C_d |p| p \Big(w^4 + h^4\Big) \end{aligned} \tag{8.27}$$

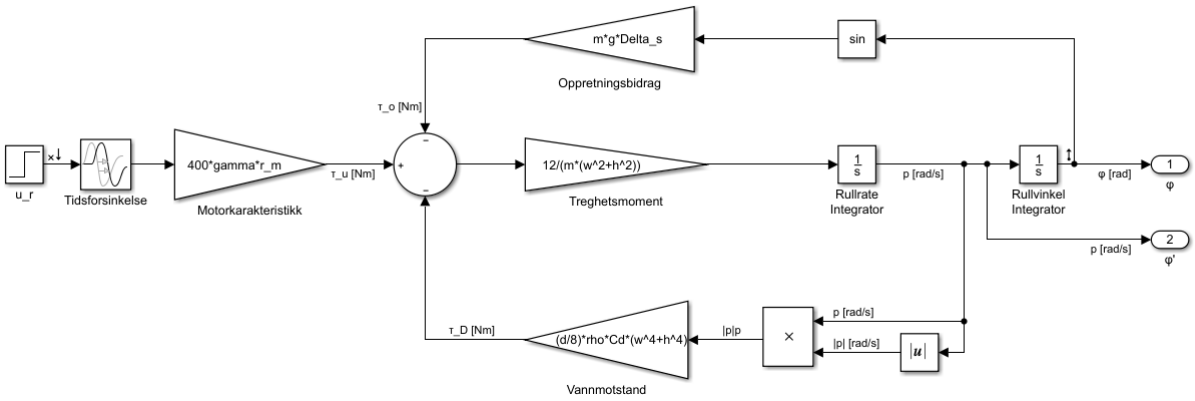
For stampprosessen blir dette løst på samme måte som for rullprosessen, men vannmotstanden vil treffe et annet areal på ROV-en. Da vil det gule rektangelet merket med **A** bli rotert 90° og ha arealet $dA = w \; dd$. Det gule rektangelet merket med **B** vil være fremme og bak og ikke på høyre- og venstresiden, slik at arealet blir,

$dA = w\ dh$. Dersom en løser det likt som for rullprosessen ender en opp med det totale dreiemomentet fra vannmotstand for stampprosessen:

$$\tau_{D,s} = \frac{w}{8}\rho C_d|q|qd^4 + \frac{w}{8}\rho C_d|q|qh^4$$
$$\tau_{D,s} = \frac{w}{8}\rho C_d|q|q\Big(d^4 + h^4\Big) \tag{8.28}$$

8.4.1 Matematisk modell av rull- og stampprosessen

I figur 8.10 vises blokkskjema for rullprosessen. Blokkskjemaet består av de ulike dreiemomentene og faktorene som er tatt hensyn til i delkapitlene over, og blir utgangspunktet for modelleringen og reguleringssystemet for rull- og stampprosessen. For stampprosessen ville dette blokkskjema hatt tre forandringer: r_m ville blitt byttet fra 0.17 m til 0.18 m, treghetsmomentet ville endret slik som vist i ligning 8.21 og vannmotstanden ville vært byttet til slik som forklart i ligning 8.28



Figur 8.10: Figuren viser blokkskjema over rullprosessen.

Basert på momentbalansen for rotasjonsprosesser, små-vinkel teoremet, treghetsmomentet fra ligning 8.21, $\tau_{u,r}$ fra ligning 8.1.2, τ_o fra ligning 8.23, $\tau_{D,r}$ fra ligning 8.27 kan ligning 8.20 brukes for å finne den matematiske modellen til rullprosessen. Denne fremgangsmåten vil være relativ lik som for dybdeprosessen:

$$\sum \tau = J\ddot{\phi} = \tau_{u,r} - \tau_o - \tau_{D,r}$$
$$\frac{1}{12}m(w^2 + h^2)\ddot{\phi}(t) = 400\gamma u(t) r_{m,r} - mg\Delta s sin(\phi(t)) - \frac{d}{8}\rho C_d|p(t)|p(t)(w^4 + h^4)$$

$$\ddot{\phi}(t) = \frac{4800\gamma r_{m,r}}{m(w^2 + h^2)}u(t) - \frac{12mg\Delta s}{m(w^2 + h^2)}\phi(t) - \frac{3d\rho C_d(w^4 + h^4)}{2m(w^2 + h^2)}|p(t)|p(t) \tag{8.29}$$

Modellen i ligning 8.29 er ulineær på grunn av $|p(t)|p(t)$ og det er derfor gunstig å lineariseres modellen rundt et arbeidspunkt. Ved å gi en liten endring Δ rundt ar-

beidspunktet A, kan vi linearisere modellen på samme måte som for dybdeprosessen, men nå for $\frac{\partial f}{\partial(u,\phi,p)}\Big|_A$.

$$\begin{aligned}\Delta\dot{\phi}(t) &= \Delta p(t)\\ \Delta\dot{p}(t) &= \frac{\partial f}{\partial u}\Big|_A \Delta u(t) + \frac{\partial f}{\partial \phi}\Big|_A \Delta\phi(t) + \frac{\partial f}{\partial p}\Big|_A \Delta p(t)\end{aligned} \tag{8.30}$$

En kan så partiell derivere de ulike leddene i arbeidspunktene basert på funksjonen $\frac{\partial f}{\partial(u,\phi,p)}\Big|_A$.

$$\begin{aligned}\frac{\partial f}{\partial u}\Big|_A &= \frac{\partial}{\partial u}\Big(\frac{4800\gamma r_{m,r}}{m(w^2+h^2)}\ u(t) - \frac{12mg\Delta s}{m(w^2+h^2)}\phi(t) - \frac{3d\rho C_d(w^4+h^4)}{2m(w^2h^2)}|p(t)|p(t)\Big)\Big|_{u=u_A}\\ &= \frac{4800\gamma r_{m,r}}{m(w^2+h^2)}\\ \frac{\partial f}{\partial \phi}\Big|_A &= \frac{\partial}{\partial \phi}\Big(\frac{4800\gamma r_{m,r}}{m(w^2+h^2)}\ u(t) - \frac{12mg\Delta s}{m(w^2+h^2)}\phi(t) - \frac{3d\rho C_d(w^4+h^4}{2m(w^2h^2)}|p(t)|p(t)\Big)\Big|_{\phi=\phi_A}\\ &= -\frac{12g\Delta s}{(w^2+h^2)}\\ \frac{\partial f}{\partial p}\Big|_A &= \frac{\partial}{\partial p}\Big(\frac{4800\gamma r_{m,r}}{m(w^2+h^2)}\ u(t) - \frac{12mg\Delta s}{m(w^2+h^2)}\phi(t) - \frac{3d\rho C_d(w^4+h^4}{2m(w^2h^2)}|p(t)|p(t)\Big)\Big|_{p=p_A}\\ &= -\frac{3d\rho C_d p_A(w^4+h^4)}{2m(w^2+h^2)}\end{aligned} \tag{8.31}$$

Deretter kan de partiellderiverte i ligning 8.31 settes inn i ligning 8.30:

$$\begin{aligned}\Delta\dot{\phi}(t) &= \Delta p(t)\\ \Delta\dot{p}(t) &= \frac{4800\gamma r_{m,r}}{m(w^2+h^2)}\Delta u(t) - \frac{12mg\Delta s}{(m(w^2+h^2)}\Delta\phi(t) - \frac{3d\rho C_d p_A(w^4+h^4)}{2m(w^2+h^2)}\Delta p(t)\end{aligned} \tag{8.32}$$

Ved å bruke sammenhengen $\Delta\phi = \int \Delta p(t)\ dt \rightarrow \Delta\phi(s) = \frac{\Delta p(s)}{s}$ kan en Laplace-

transformere ligningsettet og skrive om $\phi(t)$ under Laplacetransformasjonen:

$$\begin{aligned}
&\mathcal{L}\Big\{\Delta\dot{\phi}(t) = \Delta p(t)\Big\} \\
&\mathcal{L}\Big\{\Delta\dot{p}(t) = \frac{4800\gamma r_{m,r}}{m(w^2+h^2)}\Delta u(t) - \frac{12mg\Delta s}{m(w^2+h^2)}\Delta\phi(t) - \frac{3d\rho C_d p_A(w^4+h^4)}{2m(w^2+h^2)}\Delta p(t)\Big\}
\end{aligned} \tag{8.33}$$

$$\Downarrow$$

$$\begin{aligned}
&s\Delta\phi(s) = \Delta p(s) \\
&s\Delta p(s) = \frac{4800\gamma r_{m,r}}{m(w^2+h^2)}\Delta u(s) - \frac{12mg\Delta s}{m(w^2+h^2)}\frac{\Delta p(s)}{s} - \frac{3d\rho C_d p_A(w^4+h^4)}{2m(w^2+h^2)}\Delta p(s)
\end{aligned} \tag{8.34}$$

For å finne overføringsfunksjonen $H_{p,p}$ løses ligningen for $\frac{\Delta p(s)}{\Delta u(s)}$ og settes opp på standardform:

$$\begin{aligned}
s\Delta\phi(s) &= \Delta p(s) \\
\Big(s + \frac{12mg\Delta s}{m(w^2+h^2)s} + \frac{3d\rho C_d p_A(w^4+h^4)}{2m(w^2+h^2)}\Big)\Delta p(s) &= \frac{4800\gamma r_{m,r}}{m(w^2+h^2)}\ \Delta u(s)
\end{aligned} \tag{8.35}$$

$$\frac{\Delta p(s)}{\Delta u(s)} = \frac{\frac{4800\gamma r_{m,r}}{m(w^2+h^2)}}{s + \frac{12mg\Delta s}{m(w^2+h^2)s} + \frac{3d\rho C_d p_A(w^4+h^4)}{2m(w^2+h^2)}}$$

$$\frac{\Delta p(s)}{\Delta u(s)} = \frac{4800\gamma r_{m,r}}{m(w^2+h^2)s + 12mg\Delta s\ \frac{1}{s} + \frac{3}{2}d\rho C_d p_A(w^4+h^4)}$$

$$\frac{\Delta p(s)}{\Delta u(s)} = \frac{\Big(\frac{4800\gamma r_{m,r}}{12mg\Delta s}\Big)s}{\Big(\frac{(w^2+h^2)}{12g\Delta s}\Big)s^2 + \Big(\frac{3d\rho C_d p_A(w^4+h^4)}{24mg\Delta s}\Big)s + 1}$$

Til slutt kan en fjerne Δ-notasjonen og legge til dødtiden slik som forklart i delkapittelet om dødtid som $e^{-\tau s}$. I ligning 8.36 vises forholdet mellom vinkelhastigheten og motorpådrag for rullprosessen. For å finne forholdet mellom rullvinkelen og motorpådrag vist i ligning 8.37 brukes sammenhengen $s\Delta\phi(s) = \Delta p(s)$:

$$H_{p,p} = \frac{p(s)}{u(s)} = \frac{\Big(\frac{400\gamma r_{m,r}}{mg\Delta s}\Big)s}{\Big(\frac{w^2+h^2}{12g\Delta s}\Big)s^2 + \Big(\frac{3d\rho C_d p_A(w^4+h^4)}{24mg\Delta s}\Big)s + 1}e^{-\tau s} \tag{8.36}$$

$$H_{p,\phi} = \frac{\phi(s)}{u(s)} = \frac{\Big(\frac{400\gamma r_{m,r}}{mg\Delta s}\Big)}{\Big(\frac{w^2+h^2}{12g\Delta s}\Big)s^2 + \Big(\frac{3d\rho C_d p_A(w^4+h^4)}{24mg\Delta s}\Big)s + 1}e^{-\tau s} \tag{8.37}$$

Disse to overføringsfunksjonene er omtrent tilsvarende for stampprosessen. Som forklart er forskjellen mellom to variabler for treghetsmomentet og vannmotstanden annerledes. Det vil si at momentbalansen i ligning 8.29, så vil variablene byttes for treghetsmomentet fra $\frac{1}{12}m(w^2+h^2)$ til $\frac{1}{12}m(d^2+h^2)$ og vannmotstanden fra $\frac{d}{8}\rho C_d(w^4+h^4)$ til $\frac{w}{8}\rho C_d(d^4+h^4)$ for å få momentbalansen for stampprosessen. Deretter finner en de lineære overføringsfunksjone $H_{p,q}$ og $H_{p,\theta}$ på samme måte som for rullprosessen fra ligning 8.29 til ligning 8.35. Da ender en til slutt opp med to lineære overføringsfunksjoner, der den første er forholdet mellom vinkelhastigheten og motorpådrag, og den andre er forholdet mellom stampvinkelen og motorpådrag, vist i ligningsettet under:

$$H_{p,q} = \frac{q(s)}{u(s)} = \frac{\left(\frac{400\gamma r_{m,s}}{mg\Delta s}\right)s}{\left(\frac{d^2+h^2}{12g\Delta s}\right)s^2 + \left(\frac{3w\rho C_d q_A(d^4+h^4)}{24mg\Delta s}\right)s + 1}e^{-\tau s} \tag{8.38}$$

$$H_{p,\theta} = \frac{\theta(s)}{u(s)} = \frac{\left(\frac{400\gamma r_{m,s}}{mg\Delta s}\right)}{\left(\frac{d^2+h^2}{12g\Delta s}\right)s^2 + \left(\frac{3d\rho C_d q_A(d^4+h^4)}{24mg\Delta s}\right)s + 1}e^{-\tau s} \tag{8.39}$$

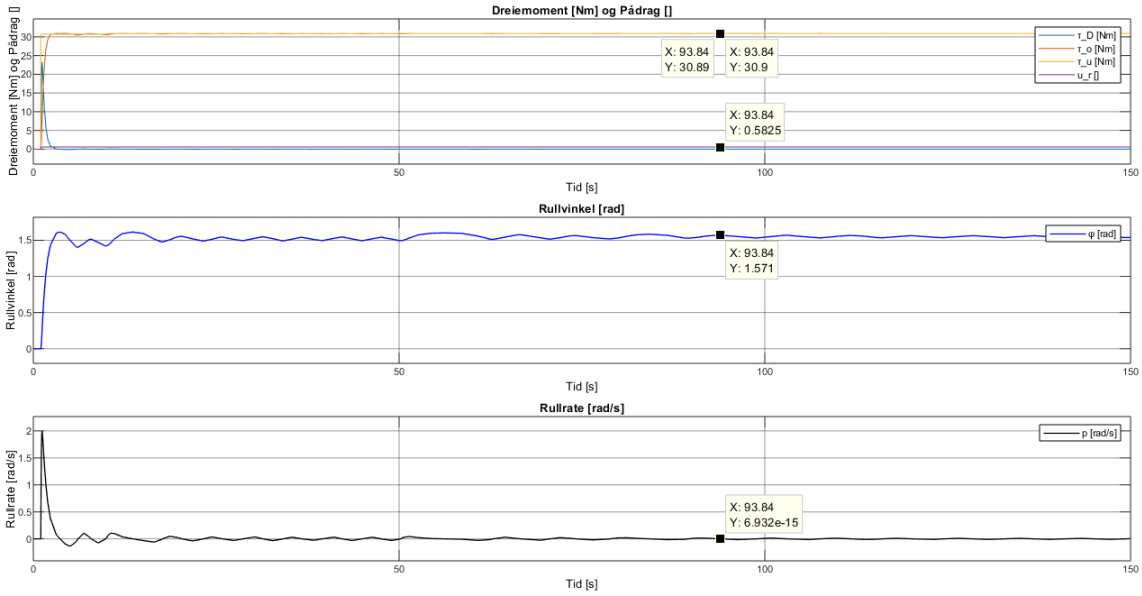
En kan merke seg at i overføringsfunksjonen mellom vinkelhastigheten for rull- og stampprosessen og motorpådraget, er det et nullpunkt i origo (i s-planet), på grunn av det er en s i telleren, en derivator. ifølge Finn Haugen [[69], side 130] vil dette nullpunktet gjøre slik at vinkelhastigheten ender opp tilbake på null stasjonært. Dette vil stemme så lenge dreiemomentet fra motorpådraget ikke blir større enn dreiemomentet fra det naturlige oppretningsbidraget, altså må $\tau_u < \tau_{o,maks}$ være oppfylt. Dersom denne ulikheten blir brutt vil det føre til at ROVen roterer rundt x-aksen for rull og rundt y-aksen for stamp, ettersom dreiemomentet fra motorpådraget er større enn det naturlige oppretningsbidraget. En kan finne den maksimale grenseverdien for når pådraget u gir et større dreiemoment enn oppretningsbidraget. En kan anta at maksimal vinkel er $\phi = \frac{\pi}{2}$, da er ROV-en rotert mest mulig om x-aksen, slik at dreiemomentet fra oppretningsbidraget blir størst mulig:

$$\tau_{u,r} < \tau_{o,maks} \rightarrow 4\gamma r_{m,r} u100\% < mg\Delta s sin(\phi)$$

$$|u| < \frac{mg\Delta s sin(\phi)}{4\gamma r_{m,r}100\%} = \frac{35\ kg \cdot 9.81\ \frac{m}{s^2} \cdot 0.09\ m \cdot sin(\frac{\pi}{2})}{4\ \cdot 0.78\ \frac{N}{\%} \cdot 0.17\ m \cdot 100\%} \tag{8.40}$$

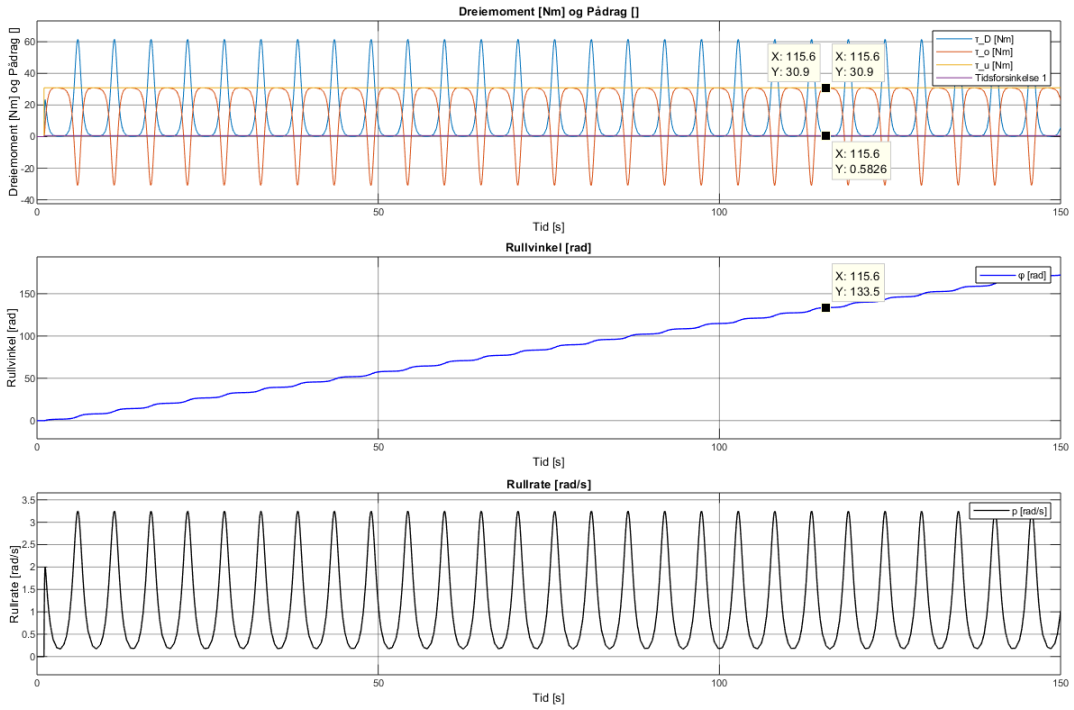
$$|u| < \frac{30.9015\ Nm}{53.0400\ Nm} = 0.5826$$

Dette kan visualiseres i Simulink ved å gi et sprang fra u_r = 0 til u_r = 0.5825 ved tiden t = 1 s på Simulink modellen vist på figur 8.10. Det som er ønskelig da er å se at ved et fast motorpådrag u_r skal vinkelhastigheten p være tilnærmet null, slik at ROV-en holder en fast vinkel ϕ. Slik som figur 8.11 viser i det øverste plottet oppfylles ulikheten ved at $\tau_{u,r} = 30.89\ Nm < \tau_{o,maks} = 30.90\ Nm$. På det nederste plottet vises det at vinkelhastigheten p, er tilnærmet 0 $\frac{rad}{s}$. Vinkelen ϕ holder seg på maksimal vinkel om x-aksen, her ved simuleringen ser en at den vinkelen varierer rundt $\phi \approx 1.571\ rad = 90°$



Figur 8.11: Figuren viser at ROVen vil holde en gitt vinkel ved et gitt pådrag slik at ROVen står stabilt i den vinkelen. Fra figuren vises det at ROVen vagger litt, men differansen utgjør < 1 ° som praktisk sett ikke vil ha stor betydning.

Så kan det være interessant å se hva som skjer når gir et sprang på u_r = 0 til u_r = 0.5826 ved tiden t = 1 s på samme Simulink modell, resultatet er vist i figur 8.12. Det en fort kan legge merke til er at vinkelen ϕ som er vist i det midterste plottet, fortsetter å stige. Etter omtrent 94 sekund har ROVen altså rotert $\frac{133.5\ rad}{2\pi} \approx 21.25$ ganger rundt x-aksen. En kan også legge merke til at vinkelhastigheten p aldri blir null og dreiemomentene fra vannmotstand og det naturlige oppretningsbidraget aldri oppnår en stasjonær verdi. I det øverste plottet kan en også se at $\tau_{u,r} = 30.90\ Nm$ ikke er mindre enn $\tau_{o,maks} = 30.90\ Nm$. Disse simuleringene viser at $\tau_u < \tau_{o,maks}$, for at ROV-en ikke skal fortsette å rotere rundt x-aksen i evig tid.



Figur 8.12: Figuren viser at dersom dreiemomentet fra motorpådraget er større en dreiemomentet fra oppretningsbidraget vil ROV-en fortsette å rulle rundt x-aksen og klarer ikke å holde en gitt vinkel.

Det nederste plottet viser at vinkelhastigheten p varierer med jevne mellomrom. Dette har en direkte sammenheng med dreiemomentet fra oppretningsbidraget. Oppretningsbidraget vil jobbe *imot* rullretningen helt til ROV-en har rotert 180°, så vil oppretningsbidraget jobbe *med* rullretningen. Når ROV-en har rotert 360° så vil oppretningsbidraget jobbe *imot* rullretningen igjen. Dette fører til at vinkelhastigheten øker hver gang ROV-en har rotert 180°, og minker hver gang ROV-en har rotert 360°.

For å finne prosesskonstantene sammenlignes overføringsfunksjonen $H_{p,\phi}$ for rull fra ligning 8.36 med standardformen for en 2.ordens overføringsfunksjon:

$$H_p = \frac{K}{\frac{1}{w_0^2}s^2 + \frac{2\zeta}{\omega_0}s + 1} \tag{8.41}$$

$$K_r = \frac{400\gamma r_{m,r}}{mg\Delta s} = \frac{400\ \% \cdot 0.78\ \frac{N}{\%} \cdot 0.17\ m}{35\ m \cdot 9.81\ \frac{m}{s^2}\ 0.09\ m} = 1.7216\ rad$$

$$\omega_{0,r} = \sqrt{\frac{12g\Delta s}{w^2 + h^2}} = \sqrt{\frac{12 \cdot 9.81\ \frac{m}{s^2} \cdot 0.09\ m}{0.500^2\ m + 0.390^2\ m}} = 5.1331\ \frac{rad}{s}$$

$$T_{r,r} \approx \frac{1}{\omega_{0,r}} = \frac{1}{\sqrt{\frac{12g\Delta s}{w^2+h^2}}} = \frac{1}{5.1331\ \frac{rad}{s}} = 0.1948\ s$$

$$\zeta_r = \frac{1}{2}\ \frac{3d\rho C_d(w^4 + h^4)}{24mg\Delta s} \cdot \sqrt{\frac{12g\Delta s}{w^2 + h^2}}\ |p_A|$$

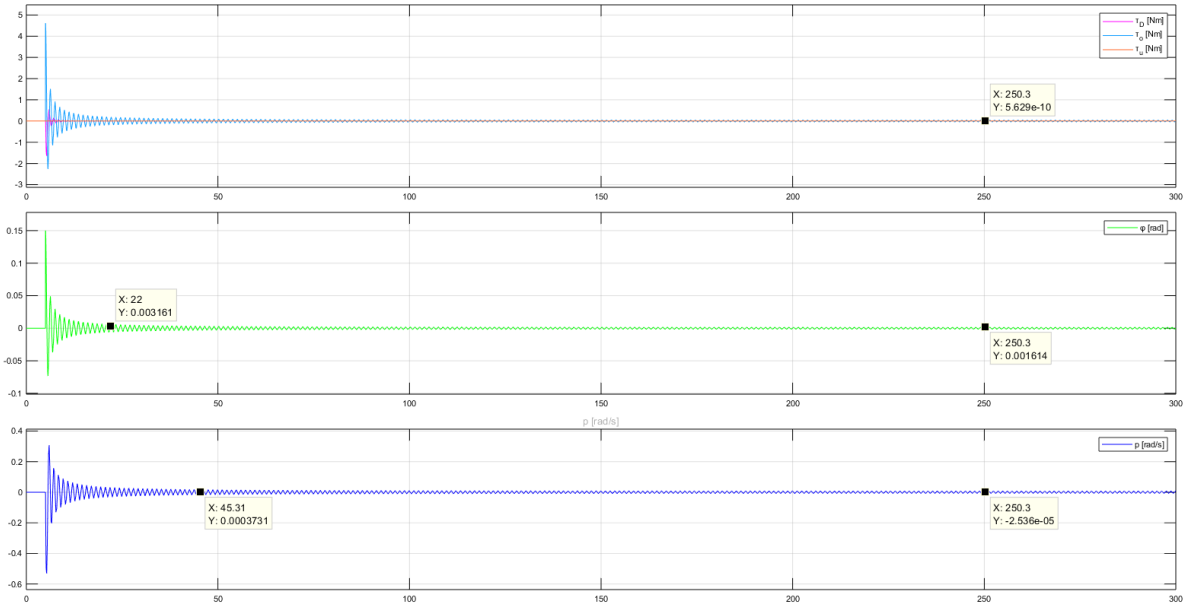
$$\zeta_r = \frac{1}{2}\ \frac{3 \cdot 0.545\ m \cdot 1000\ m^3 \cdot 1 \cdot (0.500^4\ m + 0.390^4\ m)}{24 \cdot 35\ m \cdot 9.81\ \frac{m}{s^2} \cdot 0.09\ m} \cdot 5.1331 \cdot p_A\ =\ 0.4845 \cdot |p_A| \tag{8.42}$$

Hvor:
- K_r er prosessforsterkningen for rullprosessen.
- ω_0 er resonansfrekvensen for rullprosessen.
- T_r er responstiden for rullprosessen.
- ζ_r er dempningsfaktoren for rullprosessen, det merkes at denne er proporsjonal med vinkelhastigheten.

Disse prosesskonstantene finnes på samme måte for stampprosessen og da får man:

$$K_s = 1.8217\ rad \qquad \omega_{0,s} = 4.8569\ \frac{rad}{s} \qquad T_s = 0.2059\ s \qquad \zeta_s = 0.5781 \cdot |q_A| \tag{8.43}$$

Dempningsfaktoren ζ er proporsjonal med vinkelhastigheten. Det vil si at dersom hastigheten til ROV-en er null, vil også dempningsfaktoren være null og udempet. Fra dempningsfaktoren vil rullprosessen være underdempet når $|p_A| < 2.0640$ og stampprosessen vil være underdempet når $|q_A| < 1.7299$. Systemene vil være overdempet dersom vinkelhastighetene er høyere enn disse verdiene. Siden dempningsfaktoren er proporsjonal med vinkelhastigheten vil det ta lang tid før systemet retter seg inn til en stasjonær verdi, ettersom når vinkelhastigheten synker, synker også dempningsfaktoren. Det er vist i figur 8.13 at systemet bruker lang tid å svinge seg inn til 0° når ROV-en blir sluppet fra vinkelen $\phi = 0.15\ rad$, som også gir et godt grunnlag for behovet til regulator.

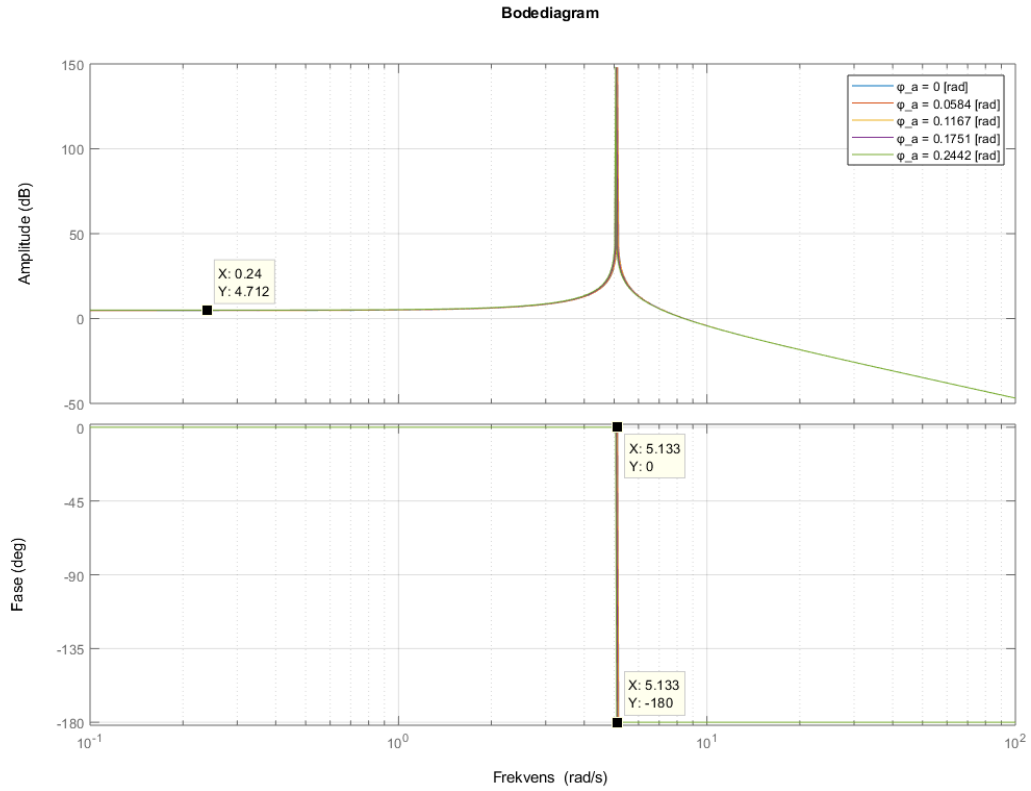


Figur 8.13: Figuren viser vannmotstand, vinkel og vinkelhastighet etter å ha sluppet ROV-en fra en vinkel $\phi = 0.15\ rad = 8.59°$.

På figuren kan en se at dreiemomentet fra vannmotstanden er avhengig av vinkelhastigheten. Etter ca 22 sekund observeres det at rullvinkelen $\phi \approx 0.003\ rad = 0.172°$. Det merkes at vinkelen aldri stabiliserer seg helt til null grader, men da den har en så lav verdi som $0.0016\ rad = 0.09°$ er en tilnærming på at vinkelen har stabilisert seg rundt 0° rimelig.

8.4.2 Simulering av rullprosessen

På lik måte som det ble gjort for dybdeprosessen i delakpittel 8.3.2 kan rullprosessen med *Linear Analysis Tool* simuleres i Simulink. Inngangen u_r settes til *open-loop input* og utgangen for vinkelen, ϕ, til *open-loop output*. Resultatet vises i figur 8.14.



Figur 8.14

Fra datatipset øverst til venstre i figuren kan en se at den stasjonære forsterkningen, K_r er 4.712 dB = 1.72 rad. Amplituden stiger veldig brått og fasen synker veldig brått ved knekkfrekvensen som kan måles ved datatipset på nederste plott. Knekkfrekvensen er relativ lik fra 0° til -180° og leses av i -135°, $\omega_{0,r} = 5.133\ \frac{rad}{s}$.

Grunnen til det er en brå stigning i amplituden og et stort hopp i fasen fra 0° til -180° ved knekkfrekvensen, skyldes at dempningfaktoren er proporsjonal med vinkelhastigheten. Siden vinkelhastigheten er veldig lav i simuleringen i Bodediagrammet vil også dempningsfaktoren være veldig lav. Når vinkelhastigheten er omtrent null vil også dempningsfaktoren være omtrent null, dette gir oversving på omtrent 100 %.

8.4.3 Konklusjon for rull- og stampprosessen

Det er gunstig at ROV-en ligger i vater horisontalt når piloten kjører, slik at piloten slipper å fokusere på å justere de ulike vinklene til ROVen. Derfor er det fornuftig å gjøre en antagelse på at rull- og stampvinkelen = 0 °og vinkelhastigheten for rull- og stampprosessen ≈ 0 $\frac{rad}{s}$. Dreiemomentet fra oppretningsbidraget vil dermed ikke gi noe bidrag ettersom $sin(0) = 0$. Dreiemomentet fra oppretningsbidraget er avhengig av den kvadrerte vinkelhastigheten, antagelsen om at vinkelhastigheten er null vil dermed gi at det ikke vil være noe vannmotstand. Dette vil si at det kun er dreiemomentet fra pådraget som vil ha en innvirkning på prosessene. Derfor kan en ut ifra disse antagelsene gjøre en tilnærming for hele rull- og stampprosessen som vil

forenkle reguleringssystemet, men samtidig gjøre det presist. Denne tilnærmingen vil dermed bli:

$$\sum \tau = J\ddot{\phi}(t) = \tau_u = 400\gamma r_{m,r} u(t) \tag{8.44}$$

Ut ifra momentbalansen kan en altså se at det ikke er noe i utrykket som gjør denne prosessen ulineær. Derfor trenger ikke dette utrykket å lineariseres rundt et arbeidspunkt slik som det ble gjort når alle dreiemomentene hadde innvirkning. Utrykket kan derfor Laplacetransformeres med en gang:

$$\begin{gathered} \mathcal{L}\Big\{ J\ddot{\phi}(t) = 400\gamma r_{m,r} u(t) \Big\} \\ \frac{1}{12} m(w^2 + h^2)\phi(s)s^2 = 400\gamma r_{m,r} u(s) \end{gathered} \tag{8.45}$$

For å finne overføringsfunksjonen for rullprosessen løses ligningen for $\frac{\phi(s)}{u(s)}$. Denne overføringsfunksjonen vil være lik som for stampprosessen der kun variabelen for bredden, w blir byttet med lengden, d. Det observeres at disse prosessene er dobbelintegrerende med s^2 i nevneren.

$$\begin{aligned} H_{p,r} &= \frac{\phi(s)}{u(s)} = \frac{\frac{4800\gamma r_{m,r}}{m(w^2+h^2)}}{s^2}\ e^{-\tau s} \\ H_{p,s} &= \frac{\theta(s)}{u(s)} = \frac{\frac{4800\gamma r_{m,s}}{m(d^2+h^2)}}{s^2}\ e^{-\tau s} \end{aligned} \tag{8.46}$$

Prosessforsterkningene kan finnes på samme måte som for dybdeprosessen med overføringsfunksjonen på standardform i ligning 8.13. Det settes inn verdier for variablene og prosessforsterkningen blir dermed:

$$\begin{aligned} K_r &= \frac{4800 \cdot 0.78\ N \cdot 0.17\ m}{35\ kg \cdot (0.500^2\ m + 0.390^2\ m)} = 45.22\ rad \\ K_s &= \frac{4800 \cdot 0.78\ N \cdot 0.18\ m}{35\ kg \cdot (0.545^2\ m + 0.390^2\ m)} = 42.87\ rad \end{aligned} \tag{8.47}$$

Det konkluderes med at disse antagelsene er mer gunstig da det skal lages et reguleringssystem for rull- og stampprosessen. Dette vil også forenkle prosessen for å lage reguleringssystemet. Se delkapittel 9.3 for utdypning av hvordan dette blir implementert.

Kapittel 9

Reguleringssystem

Innhold

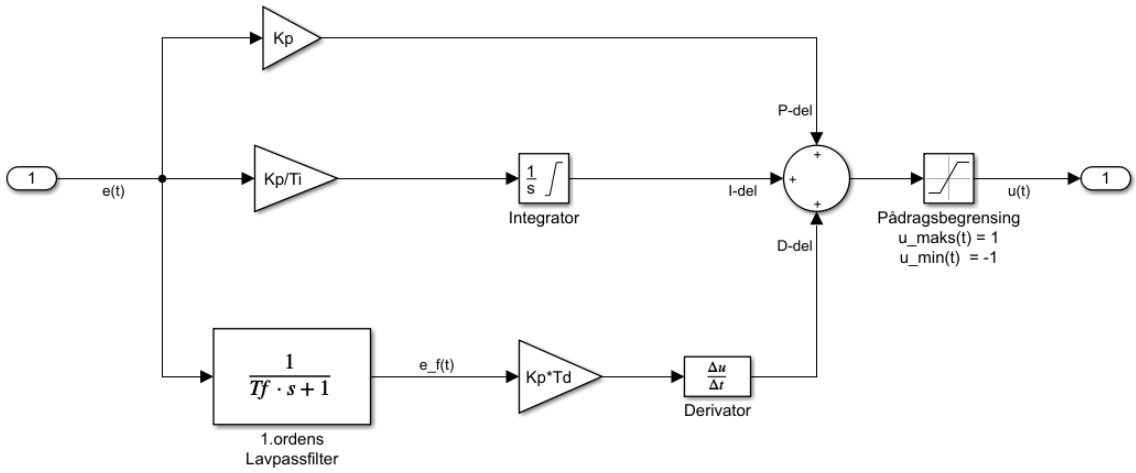
9.1 Valg av reguleringsmetode 123
9.2 Regulering av dybdeprosessen 125
9.2.1 Konklusjon av dybderegulator 135
9.3 Regulering av rull- og stampprosessen 135
9.3.1 Regulering av rullprosessen 136
9.3.2 Regulering av stampprosessen 138
9.3.3 Konklusjon av rull- og stampprosessen 139
9.4 Implementering på mikrokontroller 140
9.4.1 Implementering av dybde-, rull- og stampprosessen 143
9.4.2 Konklusjon av implementering på mikrokontroller 145

Dette kapittelet tar for seg hvilken type regulator som blir implementert for å regulere dybde-, rull- og stampprosessen. Videre blir det gjennomgått metoder for å finne de ulike reguleringsparametrene som brukes i regulatorene basert på de matematiske modellene. Det blir også utført simuleringer for å teste ulike reguleringsparametre opp mot en annen for å finne det beste alternativet.

9.1 Valg av reguleringsmetode

Det vil totalt bli brukt tre PID-regulatorer, henholdvis hver sin til dybde-, rull- og stampprosessen. PID-regulatoren er godt dokumentert og det finnes flere forskjellige metoder for å finne gode verdier for reguleringsparametrene, både teoretisk og eksperimentelt. I figur 9.1 under vises et blokkskjema på hvordan en PID-regulator ser ut på parallellform[1]. De ulike leddene blir forklart dypere under figuren.

[1] Parallellform viser innholdet til PID-regulatoren på en mer oversiktlig måte enn seriell, men har samtidig samme funksjon.



Figur 9.1: Figuren viser oppbygningen av en PID-regulator på parallellform. Inn kommer avviket e(t). det øverste leddet viser P-delen, det midterste leddet viser I-delen og det nedeste leddet viser D-delen. Disse leddene blir forklart dypere i neste avsnitt. Det er lagt inn pådragsbegrensning på maksimum 1 fremover og -1 bakover. Dette tilsvarer at motorene gir maksimalt pådrag både fremover og bakover.

P-delen beregnes av en proporsjonalforsterkning, K_p, som er proporsjonal med avviket, $e(t)$. Det vil si at ved store avvik reagerer P-leddet mye, mens med små avvik reagerer den lite. P-leddet vil minke jo nærmere en kommer ønsket verdi, og i tilfeller hvor en har et tap eller en motstand vil dette føre til at en ender opp med et stasjonært avvik.
I-delen sin oppgave er å integrere over et sett med tidligere målte verdier. Ved et stasjonært avvik vil dermed I-leddet øke pådraget nok til at avviket blir null, som igjen fører til at P-leddet blir null. Det tar nytte av intregaltiden T_i, som bestemmer hvor fort reguleringsavviket skal fjernes.
D-delen brukes til å forutse fremtidig trend i avviket, basert på avvikets endringsrate. Dette gjøres ved å derivere over en viss tid T_d, som blir multiplisert med forsterkningsfaktoren K_p.

Dersom det oppstår et stort avvik, eller om dette avviket ikke blir rettet opp i løpet av kort tid, vil I-leddet integrere til en veldig stor verdi, som igjen vil føre til at når avviket endelig nærmer seg null igjen vil I-leddet være så stort, at pådraget ikke stopper tidsnok. Dette kan i værste fall føre til veldig store oversving[2]. En måte å forhindre dette er ved noe som kalles 'Anti windup'. Det funker ved at en setter en grense på integratoren, slik at dette integralet ikke får muligheten til å gå over en satt verdi. På integratoren i I-leddet i figur 9.1 kan en i Simulink sette denne grensen ved å dobbelklikke på integratoren, trykke på *'Limit output'* og sette øvre og nedre integralgrense. Disse grensene blir normalt sett beregnet ved $integratorbegrensning = \pm\frac{Maksimaltpdrag}{T_i}$, men kan også finnes ved testing for ulike størrelser på avviket.

[2]Oversving betyr at den målte verdien svinger over referansen, dersom den målte verdien svinger under referansen kalles det undersving.

D-leddet er veldig utsatt for brå endringer, og en ulempe med dette er at det vil påvirke systemet på en negative måte dersom det kommer brå feilmålinger eller støy. Det som skjer da er at D-leddet deriverer over et stort avvik som vil føre til veldig store verdier, mens det i realiteten ikke er et stort avvik, men støy. Dette gjør at reguleringssystemet kan bli ustabilt. For å unngå er det vanligst å filtrere og så derivere målesignalet, men Finn haugen [[64], side 96] anbefaler å bruke et 1. ordens lavpassfilter[3] som skal filtrere ned de uønskede feilmålingene og støyet direkte fra avviket, slik som vist i figur 9.1 i det nederste leddet. Finn Haugen anbefaler en tidskonstant for filteret, T_f, som er proporsjonal med T_d med en faktor fra 0.05 til 0.2. Gjennom lavpassfilteret gis det ut et filtrert avvik, $e_f(t)$ som kun gjelder for D-delen.

P-I-D leddene blir tilsammen addert, slik at det kommer et nytt utgangsignal for pådraget, $u(t)$, slik som vist i ligning 9.1 under.

$$u(t) = K_p \cdot e(t) \; + \; \frac{K_p}{T_i} \int_0^t \; e(t)dt \; + \; K_p \cdot T_d \cdot \frac{d}{dt}(e_f(t)) \tag{9.1}$$

Ved bruk av alle tre P-I-D leddene og sammenhengen $e_f(s) = \frac{1}{T_f s+1} e(s)$ kan en skrive opp overføringsfunksjonen for regulatoren:

$$H_r(s) = \frac{u(s)}{e(s)} = K_p \; + \; \frac{K_p}{T_i s} \; + \; \frac{K_p T_d s}{T_f s + 1} \tag{9.2}$$

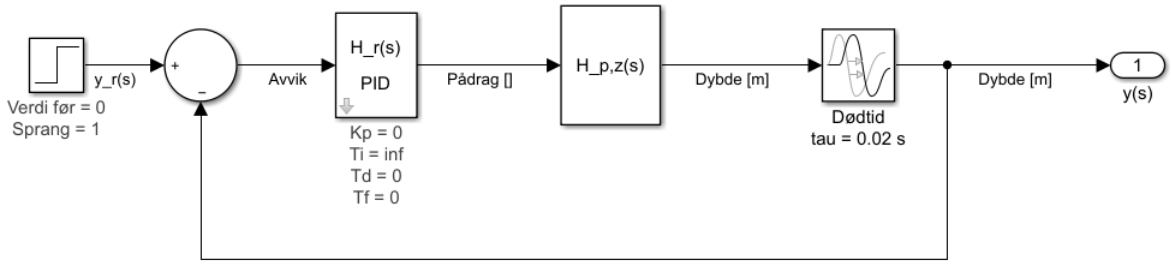
9.2 Regulering av dybdeprosessen

For dybdeprosessen brukes Skogestads metode for å finne reguleringsparametrene [64]. For denne metoden kreves en relativt enkel matematisk modell, som ble funnet i ligning 8.12. Fra denne ligningen vil det bli funnet reguleringsparametre, og disse parametrene vil bli brukt til å simulere reguleringssystemets oppførsel på det ulineære systemet (systemet vist i figur 8.5). Skogestads metode baserer seg på følgeforholdet $M(s)$ som vist i ligning 9.3 under. Følgeforholdet vil si hvor godt utgangen $y(t)$ klarer å følge referansen $y_r(t)$ ved et sprang eller forstyrrelse inn til systemet, slik at differansen (avviket) mellom de blir null.

$$M(s) = \frac{y(s)}{y_r(s)} = \frac{1}{T_c s + 1} e^{-\tau s} \; = \frac{H_r(s) \cdot H_p(s)}{1 + H_r(s) \cdot H_p(s)} \tag{9.3}$$

Hvor:
- T_c er tidskonstanten det regulerte systemet, M(s).
- τ er dødtiden i systemet.
- $H_r(s)$ er overføringsfunksjonen for regulatoren.
- $H_p(s)$ er overføringsfunksjonen for prosessen.

[3]Lavpassfilter filtrerer signaler over en viss frekvens, typisk støy.



Figur 9.2: Figuren viser blokkskjema av Skogestads metode. Dette blokkskjemaet er ekvivalent til følgeforholdet M(s) vist i ligning 9.3.H_r(s) er PID-regulatoren vist i figur 9.1, og H_p,z(s) er den ulineære modellen vist i figur 8.5

Skogestads metode baserer seg på standardformen til overføringsfunksjonen for prosessen, i dette tilfellet vil det være $H_{p,z}(s)$ fra ligning 8.12. I $H_{p,z}(s)$ kan en se at det er en prosessforsterkning i telleren, K_z, en integrator (en ekstra s under brøkstreken), en tidskonstant (T_z) og en dødtid (τ). Det vil si at overføringsfunksjonen for prosessen er på denne formen:

$$H_p(s) = \frac{K}{(Ts+1)s+1} e^{-\tau s} \tag{9.4}$$

Fra den matematiske modellen ble det vist at prosesskonstantene K_z og T_z er avhengig av arbeidspunktet $v_{z,A}$. Det som er viktig for regulatorene er å velge det mest kristiske arbeidspunktet for ROV-en, altså i den farten den kommer til å operere mest i. Med hensyn til MATE-konkurransen vil dette arbeidspunktet være en lav hastighet med tanke på oppgaver skal utføres i et basseng. Det vil også være en fordel å finne reguleringsparametre for et lavt arbeidspunkt, slik at dersom ROV-en befinner seg i et høyere arbeidspunkt vil ikke stabiliteten bli redusert. Dersom en velger å finne reguleringsparametre for for eksempel $v_{z,A} = 1\ \frac{m}{s}$ og ROV-en opererer kun i $v_{z,A} = 0.1\ \frac{m}{s}$ vil det bli veldig mye oversving og ustabilitet i systemet. Ved å definere det mest kritiske arbeidspunktet til $v_{z,A} = 0.1\ \frac{m}{s}$, vil prosesskonstantene dermed bli; $K_z = 14.25\ m$ og $T_z = 1.60\ s$.

Fra Skogestad sin tabell [[64], Kap. 10.3.2] kan reguleringsparametrene basert på standardformen for prosessen og prosesskonstantene finnes for en PID-regulator på seriellform. Ettersom vi har en regulator på parallellform kan parametrene først finnes for seriellform, for å derretter gjøre de om til parallellform med formlene vist

under [[64], Kap. 10.3.4]:

$$
\begin{aligned}
K_{p,s} = \frac{1}{K_z(T_c + \tau)} \;&\longrightarrow\; K_{p,p} = K_{p,s}\Big(1 + \frac{T_{d,s}}{T_{i,s}}\Big) \\
T_{i,s} = c(T_c + \tau) \;&\longrightarrow\; T_{i,p} = T_{i,s}\Big(1 + \frac{T_{d,s}}{T_{i,s}}\Big) \\
T_{d,s} = T_z \;&\longrightarrow\; T_{d,p} = T_{d,s}\left(\frac{1}{1 + \frac{T_{d,s}}{T_{i,s}}}\right)
\end{aligned}
\tag{9.5}
$$

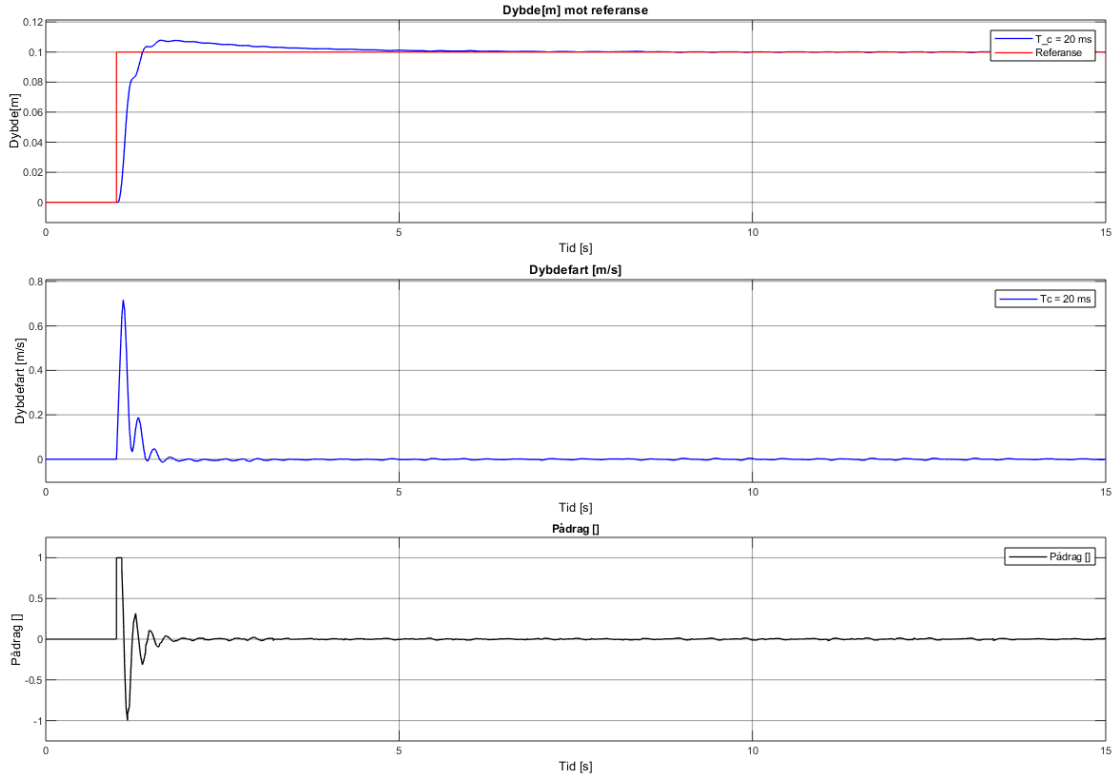
Konstanten c er en faktor som blir anbefalt av Skogestad for å gi reguleringssystemet bedre følgeegenskaper[4]. Skogestad anbefaler denne faktoren til å være $c = 4$. Dersom c blir satt til en lavere verdi vil systemet få bedre kompenseringsegenskaper[5], men vil da også føre til mer oversving i systemet og er ikke like robust mot eventuelle endringer for prosessparametrene.

Tidskonstanten T_c er en selvvalg konstant fra brukeren, men Skogestad foreslår å bruke $T_c = \tau$, som i dette systemet er lik 20 ms (ref delkapittel 8.1.2). Tidskonstanten settes for å velge hvor kjapt systemet skal få avviket til null. Til lavere T_c er, desto hissigere respons og mer oversving vil en få og til høyere T_c er, desto lenger tid tar det før systemet når en stasjonær verdi. Dersom $T_c = \tau = 20\ ms$, vil det bli en veldig lav tidskonstant for gjeldende system. Reguleringsparametrene med $T_c = \tau = 20\ ms$ blir; $K_p = 19.294 \quad T_i = 1.760 \quad T_d = 0.145$ basert på ligning 9.5.

Figuren under viser simulering av systemet med $T_c = \tau = 20\ ms$. Det som er verdt å merke seg er at reguleringssystemet har problem med å holde referansen slik at avviket aldri blir helt null. I det nederste plottet vises det at responsen er svært hissig ved å gi maksimum pådrag i begge retninger for å prøve å få ned avviket, samt at det fortsetter å gi pådrag gjennom hele simulering ettersom avviket aldri blir null.

[4]Følgeegenskapen forteller hvor godt regulatoren klarer å følge referansen.

[5]Kompenseringsegenskapen forteller hvor godt regulatoren klarer å kompensere bort effekten av forstyrrelser.



Figur 9.3: Ved å gi et sprang fra u=0 til u=0.1 ved t=1 viser figuren oppførselen til regulatoren med $T_c = \tau = 20\ ms$. Regulatoren klarer ikke å gjøre jobben sin, og avviket fortsetter å være stort med tiden. At pådraget går i metning betyr i praksis at det vil bli gitt fullt pådrag.

Det tas med at $T_c = \tau = 20\ ms$ ikke er brukbart for dette systemet og at $T_{d,s} = T_z$ ikke er svært gunstig siden det resulterer i stor proporsjonalforsterkning som påvirker oppførselen på en negativ måte. Det ble testet med flere verdier for T_c når $T_{d,s} = T_z$, resultatene var at pådraget gikk i metning hver gang, og det er ikke ønskelig i dette tilfellet at det skal være så rask forandring i pådraget på kort tid. Dette er både for regulatorens skyld, samt for å unngå store forandringer i strømtrekket til motorene.

Videre vil derfor $T_{d,s} = T_{i,s}$ som vil resultere i bedre parametre for dette systemet. Basert på det grunnlaget vil det derfor bli testet med ulike verdier for T_c. Som nevnt er T_c er en selvvalgt tidskonstant, og det kan være ideelt å simulere med verdier basert på tidskonstanten for prosessen, T_z. $T_z = 1.60\ s$ er veldig mye høyere enn $T_c = \tau = 0.02\ s$, derfor er det gunstig å simulere og teste ulike verdier for å finne den responsen som er mest ønsket for ROV-en. Oppførselen bør være responsiv og kjapp, men samtidig har lite oversving i forhold til referansen. Dersom en bruker en lav verdi for T_c fører det til mer oversving, og dersom T_c er en høyere verdi fører det til at systemet blir tregere. Det testes først med tre sett med reguleringsparametrene for å få indikasjon på hvilket verdiområdet for T_c som gir best resultat, parametrene er vist i tabellen under:

T_c	$K_{p,d}$	$T_{i,d}$	$T_{d,d}$
$T_c = T_z/8 = 200\ ms$	0.638	1.760	0.440
$T_c = T_z/4 = 400\ ms$	0.334	3.360	0.840
$T_c = T_z/1.77 \approx 900\ ms$	0.153	7.360	1.840

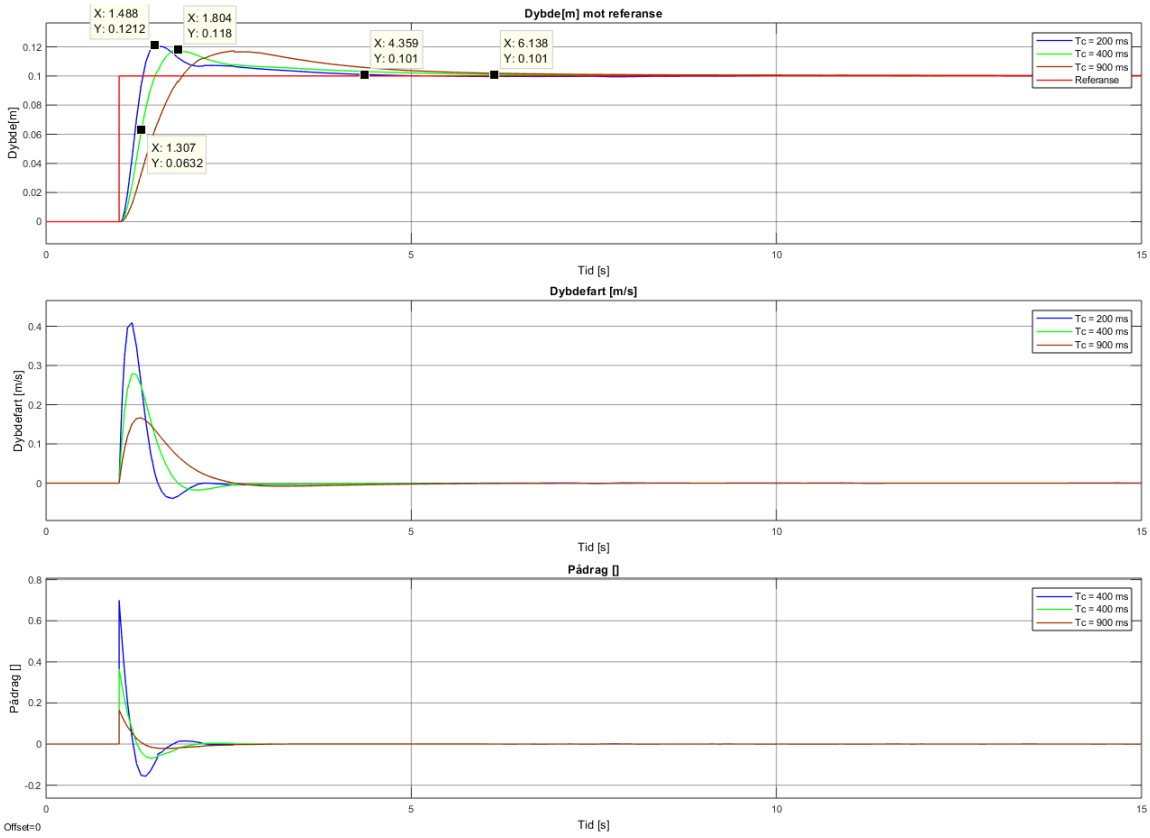
Tabell 9.1: Tabellen viser de ulike reguleringsparametrene for dybdeprosessen med ulike verdier for T_c. Reguleringsparametrene er funnet ved bruk av ligning 9.5. Det ble testet med ulike verdier for T_c, men område fra $T_c = T_z/8 = 200\ ms$ til $T_c = T_z/1.77 \approx 900\ ms$ ga reguleringsparametrene med best resultater.

I figur 9.4 under vises responsen for de ulike verdiene av T_c fra tabell 9.1. Simulering gjort i figuren tilsvarer at en sender inn et ønsket om at ROV-en skal forflytte seg 0.1 meter i dybderetning. Forskjellen for en T_c med for lav eller for høy verdi vises i figuren. Den brune responsen har omtrent like mye oversving som de to andre, men særdeles større responstid[6] og stabiliseringstid[7] og ses derfor bort ifra. Den blå responsen viser at en lav T_c gir en hissig respons med mye pådrag som utgjør en raskere responstid, men gir mer litt mer oversving. Den grønne responsen har vesentlig mindre pådrag (rundt 50% mindre), som resulterer til litt høyere responstid i forhold til den blå (sett på datatipset nede til venstre, i dette tilfellet har grønn respons en responstid på 1.307 - 1.000 = 0.307 s)[8] for å få mindre oversving. Den blå responsen har 3.2 % mer oversving enn den grønne responsen. Den grønne har en 1.779 sekund mer stabiliseringstid mer enn den blå responsen. (merk: fortsatt 0.001 m i avvik).

[6]Responstid er tiden det tar for å nå 63.2 % av stasjonær verdi.

[7]Stabiliseringstid er tiden det tar for å nå stasjonær verdi lik referansen. I dette tilfellet regnes 0.101 m som tilnærmet stasjonær verdi

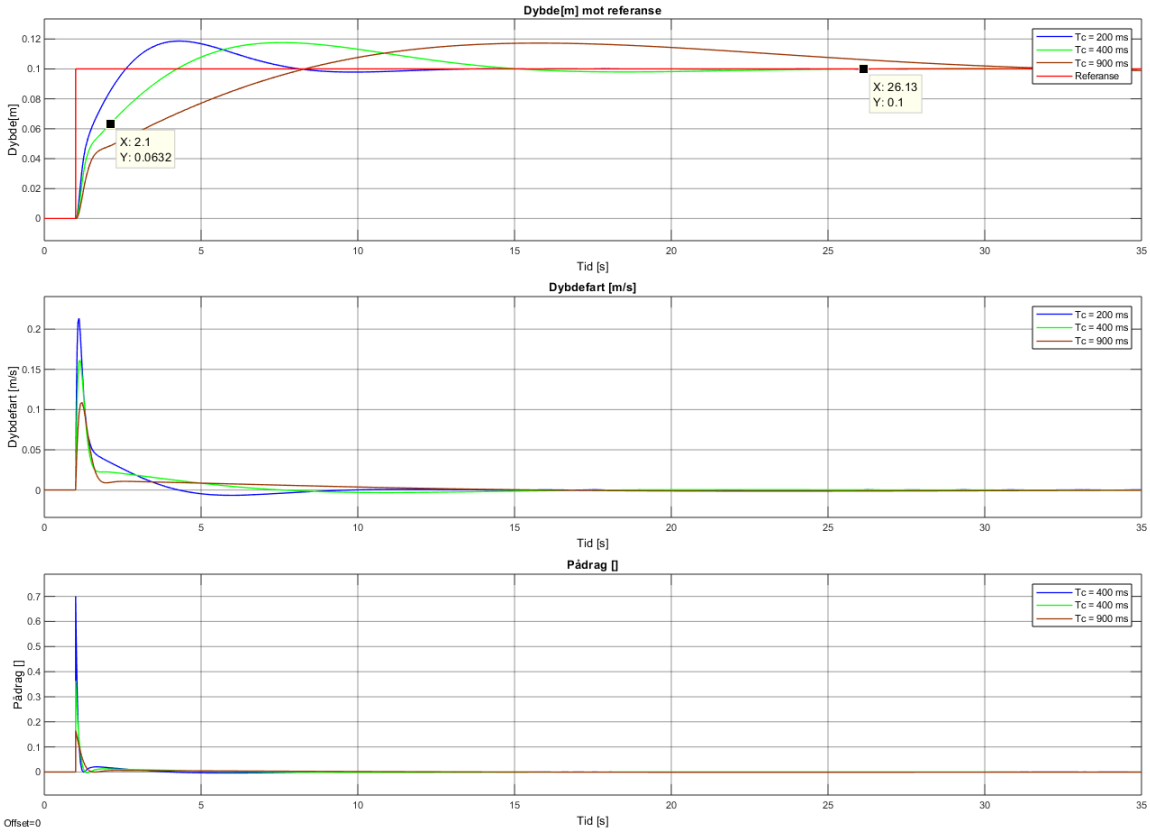
[8]Riktig responstid finnes ved å ta tiden ved 63.2% av spranget, minus tiden spranget startet på. Dette gjelder også for stabiliseringstiden.



Figur 9.4: Figuren over viser responsen fra de ulike reguleringsparametrene vist i tabell 9.1. Det gis her et sprang fra u=0 til u=0.1 ved t=1. Figuren viser godt responsen for både en for lav og for høy verdi av T_c. Den grønne responsen har en responstid på 0.986 s, et oversving på 12.4 % og en stabiliseringstid på 8.051 s. Det midterste plottet viser også at dybdefarten holder en mye jevnere og stabil fart gjennom responsen.

Fra figuren er det konkludert med at den grønne responsen er mest gunstig av de tre responsene. Dette er hovedsaklig fordi pådraget er halvert fra den blå responsen. som fører til at responstiden bli omtrent lik den blå og at oversvinget minker. Det er merkbart at stabiliseringstiden på 6.138 - 1.000 = 5.138 er svært rimelig. Dybdefarten er jevn og behagelig som alt i alt resulterer til en ønskelig oppførsel.

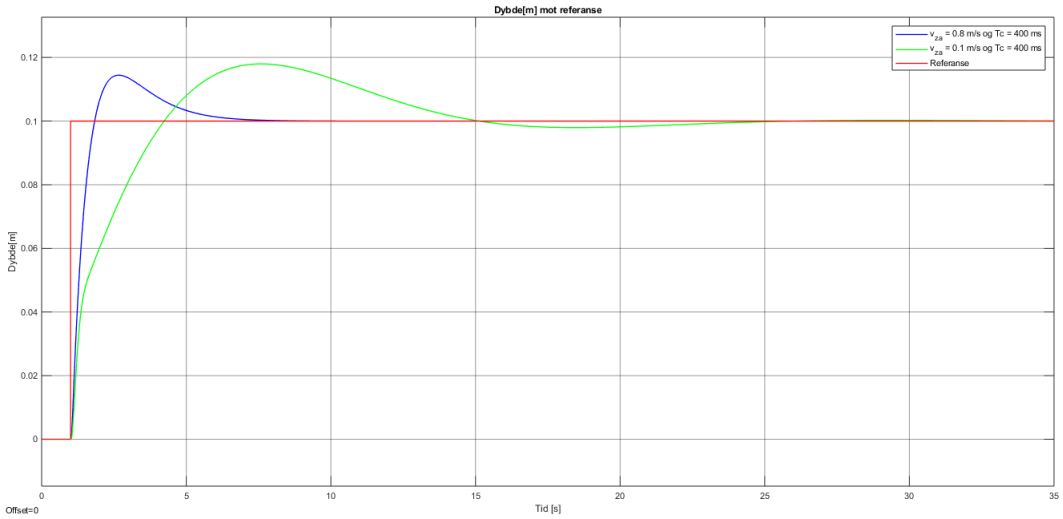
Et problem med å velge disse reguleringsparametrene er at dybdeprosessen er linearisert rundt *et* arbeidspunkt avhengig av dybdefarten $v_{z,A}$. Det er allerede forklart at reguleringsparametrene over er funnet med $v_{z,A} = 0.10\ \frac{m}{s}$. I figur 9.5 under vises oppførselen i et annet høyere arbeidspunkt, $v_{z,A} = 0.8\ \frac{m}{s}$, med de samme reguleringsparametrene funnet i tabell 9.1. Ved å sammenligne den grønne responsen fra figur 9.4 og figur 9.5 vises det at responstiden er omtrent 3.5 ganger så mye og stabiliseringstiden er rundt 5 ganger større.



Figur 9.5: Figuren viser oppførselen til regulatoren med parametrene funnet i tabell 9.1 med et større arbeidspunkt for ROV-en.

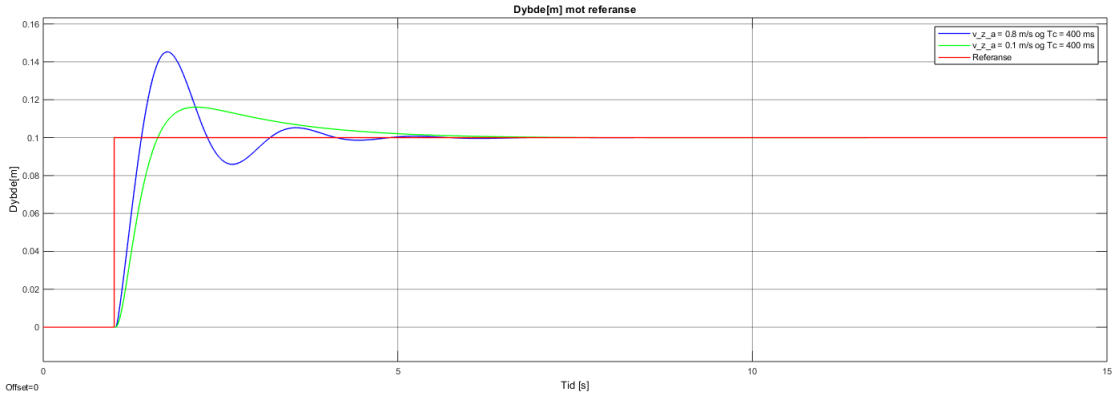
Figuren viser at det kan være gunstig å ha ulike reguleringsparametre i forhold til arbeidspunktet ROV-en befinner seg i. Hovedgrunnen til dette er at de dynamiske egenskapene til prosessen er avhengige av farten til ROV-en. En måte å implementere dette på er en metode som kalles parameterstyring. Denne metoden tar hensyn til ROV-ens arbeidspunkt som hele tiden måles og justerer reguleringsparametrene i forhold til det. Grunnen for at dette er praktisk er siden den matematiske modellen kun er linearisert rundt et arbeidspunkt, og ved å endre arbeidspunktet endres også de dynamiske egenskapene til systemet (K_z og T_z). Disse dynamiske egenskapene blir brukt til å finne reguleringsparametrene, det vil si at det er en rød tråd gjennom hele prosessen. I teorien vil parameterstyring gjøre slik at systemet tilpasser seg mye bedre i forhold til forstyrrelser i det gitte arbeidspunktet, og dermed forbedrer hele reguleringssystemet.

Med parameterstyring vil reguleringsparametrene endres fortløpende, på figuren under kan en se en sammenligning mellom responsen for $T_c = 400\ ms$ funnet med arbeidspunktet$v_{z,A} = 0.1\ \frac{m}{s}$ mot reguleringsparametre funnet med ligning 9.5, men for arbeidspunktet $v_{z,A} = 0.8\ \frac{m}{s}$ og innsatt $T_c = T_z = 400\ ms$. Parametrene som blir sammenlignet i figuren er reguleringsparametrene følgende;
$v_{z,A} = 0.8\ \frac{m}{s}$: $T_c = 400\ ms$: $K_{p,d} = 1.487$, $T_{i,d} = 1.879$ og $T_{d,d} = 0.177$
$v_{z,A} = 0.1\ \frac{m}{s}$: $T_c = 400\ ms$: $K_{p,d} = 0.334$, $T_{i,d} = 3.360$ og $T_{d,d} = 0.840$
Figuren under blir simulert med arbeidspunktet $v_{z,A} = 0.8\ \frac{m}{s}$.



Figur 9.6: Systemet blir nå simulert med arbeidspunktet for $v_{z,A} = 0.8\ \frac{m}{s}$. Figuren viser at det er stor forskjell dersom en bruker reguleringsparametrene som er best egnet for et lavt arbeidspunkt mot et høyt arbeidspunkt. Den blå responsen er parametre som er tiltenkt for det simulerte arbeidspunktet, mens den grønne er for et lavere arbeidspunkt.

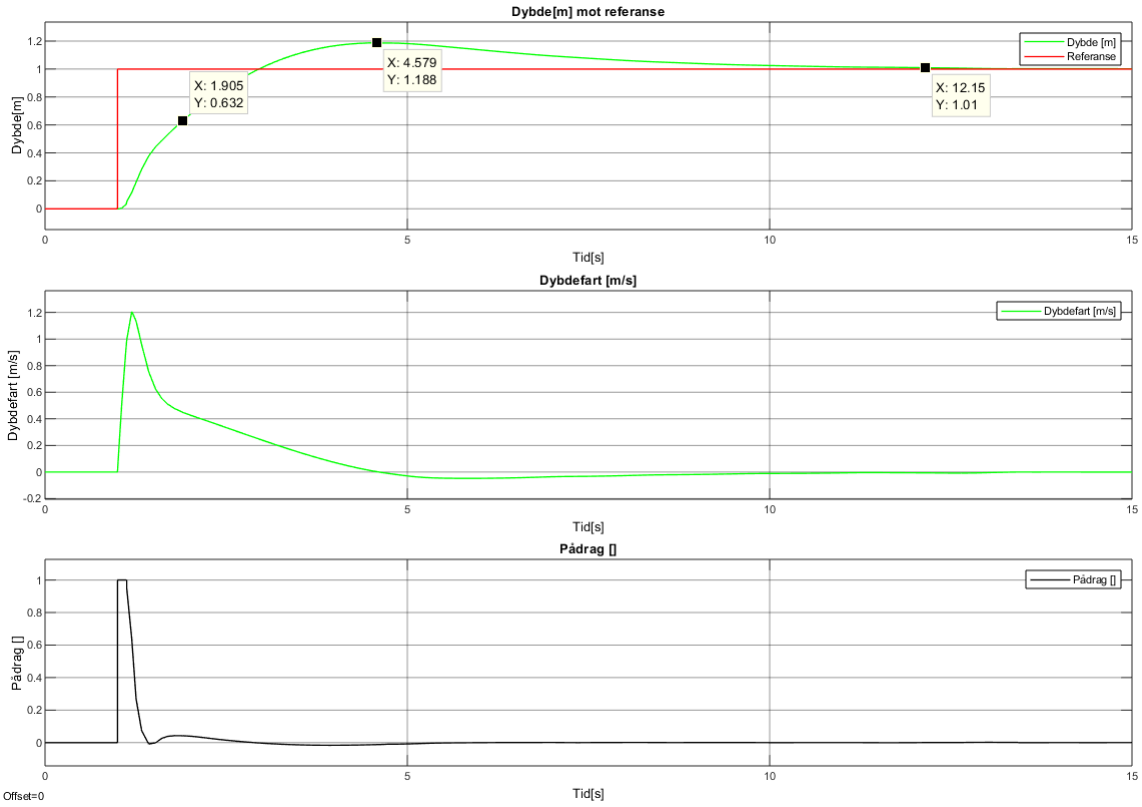
Dersom en snur simuleringen, og sammenligner samme reguleringsparametrene for et lavere arbeidspunkt, denne gang for arbeidspunktet $v_{z,A} = 0.1\ \frac{m}{s}$. Det er merkbart at reguleringsparametre tiltenkt et høyt arbeidspunkt blir mer ustabilt i et lavere arbeidspunkt og at reguleringsparametre tiltenkt et lavt arbeidspunkt blir mer stabilt i et høyere arbeidspunkt.



Figur 9.7: Systemet blir nå simulert med arbeidspunktet for $v_{z,A} = 0.1\ \frac{m}{s}$.

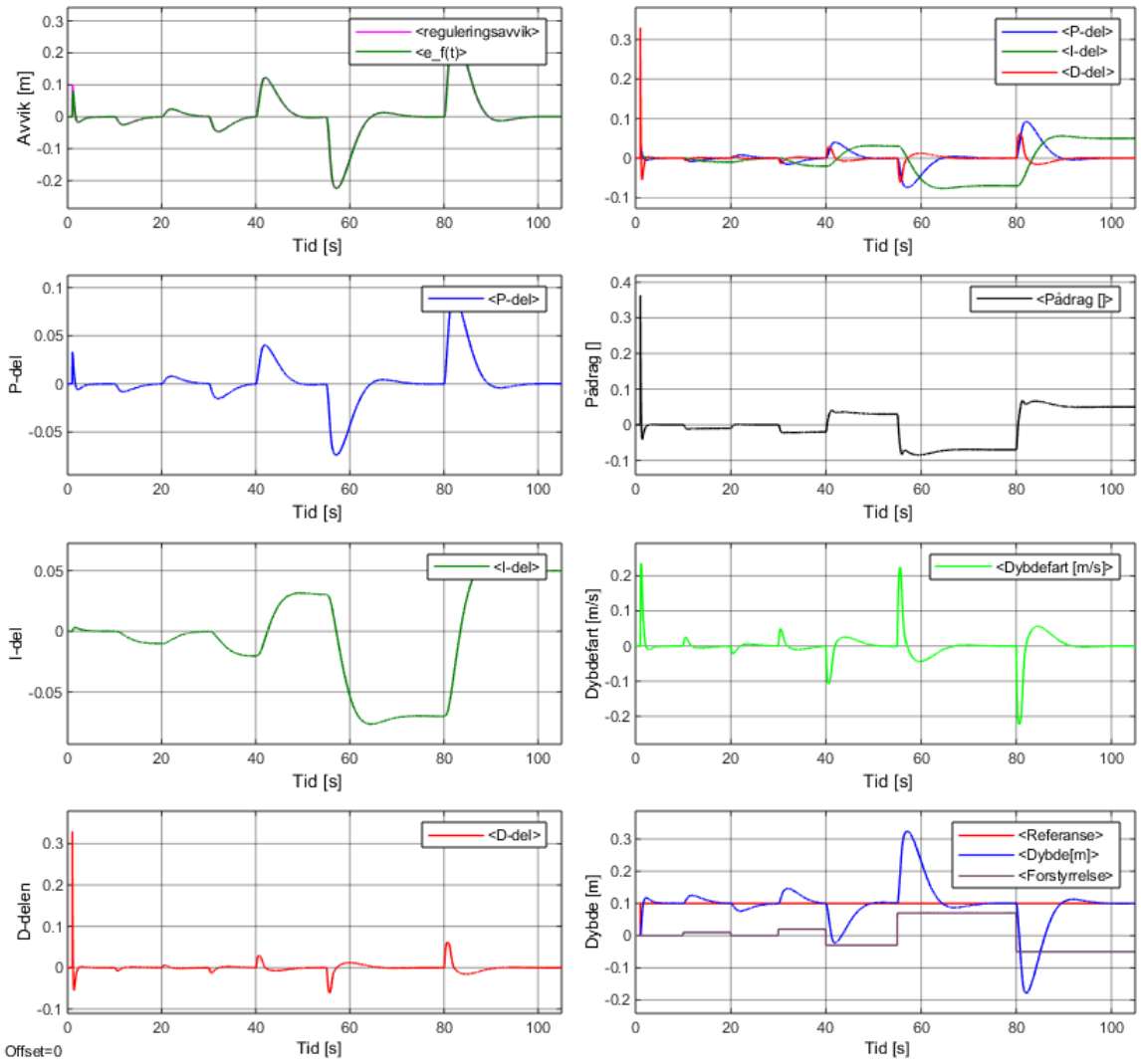
Figur 9.6 viser hva som skjer dersom en er i et høyt arbeidspunkt med parametre tiltenkt for lavere arbeidspunkt og Figur 9.7 viser hva som skjer dersom en er i et lavere arbeidspunkt med parametre tiltenkt for høyere arbeidspunkt. Figurene viser altså at det kan være en fordel å ha parameterstyring som kan hjelpe regulatorene å bli bedre og tilpasse parametrene i forhold til arbeidspunktet slik at de reguleringsparametrene passer til de dynamiske egenskapene til prosessen. Det kan likevel være lurt å stille spørsmålet: Hvor ofte kommer ROV-en til å ha et arbeidspunkt på 0.8 $\frac{m}{s}$?

Hvis en går tilbake til systemet med det lave arbeidspunktet og responsen med $T_c = 400\ ms$ kan det være interessant å se hvordan oppførselen til regulatoren blir dersom referansen endres fra 0.1 m til 1 m. I figur 9.8 under vises det at responstiden er relativ rask, og er målt med datatipset til å være 1.905 - 1.000 = 0.905 s. Oversvinget er 18.8 %, oversvinget er relativt likt, mens responstiden er omtrent 3 ganger større. Stabiliseringstiden er relativ rask, og etter 11.15 sekunder er det kun 0.01 m i avvik. Det vil si at disse reguleringsparametrene egner seg godt for større sprang i referansen.



Figur 9.8: Figuren viser at regulatoren er responsiv for større sprang i referansen. Regulatoren utnytter motorens kraft godt ved å gi et større pådrag med en gang, og dette resulterer i at dybdefarten også reagerer hurtig samtidig som det er stabilt.

Det kan også være interessant å se hvordan systemet takler flere forstyrrelser, mens ROV-en prøver å holde en gitt referanse. Her er det brukt reguleringsparametrene for $T_c = 400\ ms$. I figur 9.9 under vises det i plottet nederst til høyre hvordan regulatoren oppfører seg i forhold til at forstyrrelser ved ulike tidspunkt. De første referanse endringene som er relativt små kan sammenlignes med at manipulatoren for eksempel plukker opp gjenstander, mens de siste forstyrrelsene er i overkant overdrevet og kan gjenskape situasjoner hvor ROV-en blir påvirket av ytre uforutsigbare faktorer som påvirker i større grad eller at manipulator plukker opp tyngre gjenstander.



Figur 9.9: Figuren viser hvordan regulatoren oppfører seg i forhold til flere endringer i referansen. Figuren oppsummerer hvordan de ulike delene av regulatoren påvirker systemet totalt sett.

Figuren viser også at ved de små forstyrrelsene er det mest P- og I-delen som påvirker regulatoren, mens når de større forstyrrelsene kommer hjelper D-delen godt til. Det filtrerte avviket viser ikke helt hensikten i dette tilfellet, da det ikke er noe støy eller feilmålinger, men viser at det demper ned litt av de større forstyrrelsene. I starten av kapittelet ble det forklart hvordan 'Anti windup' kan brukes, fra plottet om I-delen vises det at integralet får en verdi på 10^{-2} og dette er ikke stort nok til at det blir et problem i dette tilfellet. Dersom avviket hadde brukt lang tid på å rette seg inn, hadde I-delen bygget seg opp til en betraktelig større verdi. Det ble testet med å sette en referanse på 100 meter, som resulterte med at systemet ga pådrag lenge etter at ønsket dybde var oppnådd, grunnet at I-leddet var blitt så stort. Det er derfor nødvendig med en integratorbegrensning. Denne blir funnet ved $integratorbegrensning = \frac{u_{maks}}{T_{i,d}} \longrightarrow \frac{+/-1}{3.360} = +/- 0.298$ for simuleringen.

9.2.1 Konklusjon av dybderegulator

Fra simulering i Simulink ble det funnet tilfredstillende resultater for en regulator som er beregnet for arbeidspunktet $v_{z,A} = 0.1\ \frac{m}{s}$. Skogestad sin metode[9] ga flere ulike resultater som ble testet på samme måte som i figur 9.9 med forstyrrelser. Fra tabell 9.1 ga $T_c = 400\ ms$ indikasjoner på en god respons, og det ble testet med $T_c = 400 +/- 100\ ms$ uten at responsene ble mer tilfredstillende. Figur 9.9 viste også at de reguleringssystemet er robust mot tiltenkte forstyrrelsene ROV-en kommer til å oppleve fra omtrent 0.01 m til 0.1 m.

Det ble også bevist at disse reguleringsparametrene ikke var svært effektive dersom arbeidspunktet ble økt til $v_{z,A} = 0.8\ \frac{m}{s}$. For å svare på spørsmålet som ble stilt under figur 9.7 så vil det mest sannsynlig ikke være et så høyt arbeidspunkt i MATE-konkurransen hvor alt skal foregå i et basseng. Dybderegulatoren sin største oppgave i forhold til konkurransen vil å holde en gitt referanse selv om det kommer en forstyrrelse, for eksempel om manipulatoren plukker opp en gjenstand. Dersom manipulatorarmen er i bruk vil også arbeidspunktet til ROV-en ligge i nærheten av null ettersom ROV-en står i ro. Med prosessmodellen endres arbeidspunktet og de dynamiske egenskapene seg fortløpende, og dersom ROV-en faktisk er i et høyt arbeidspunkt og bruker en regulator tiltenkt et lavt arbeidspunkt som $v_{z,A} = 0.1\ \frac{m}{s}$, så vil det bli et tregt system med en større responstid og stabiliseringstid.

Dersom ROV-en skulle bli brukt til større operasjoner der distansen ROV-en skal kjøre i er større, og muligens ytre uforutsigbare forstyrrelser var mer relevant, ville det vært gunstig å implementert parameterstyring. Det ses ikke på som et behov for årets ROV i forhold til MATE-konkurransen.

Med dette i forbehold vil det derfor bli tatt utgangspunkt for følgende reguleringsparametre for dybderegulatoren, inntill testing og tuning av dybderegulatoren blir gjort på en ferdig ROV:

$$K_{p,d} = 0.334 \qquad T_{i,d} = 3.360 \qquad T_{d,d} = 0.840 \qquad T_{f,d} = 0.084$$

9.3 Regulering av rull- og stampprosessen

For å unngå gjentakelse av lik fremgangsmåte for rull- og stampprosessen, vil det i dette kapittelet hovedsaklig bli forklart, beregnet og simulert for rullprosessen. Stampprosessen vil også bli noe gjennomgått, men ikke like detaljert. For rull- og stampprosessen vil det bli brukt Skogestads metode for å finne parametrene, ettersom den ga gode resultater for dybdeprosessen. Den foretrukne responsen fra Skogestads metode vil også bli etterjustert og sammenlignet for å se om responsen kan forbedres ytterlige.

For Skogestads metode vil fremgangsmåten være relativ lik i dette tilfellet som det var for dybdeprosessen. Den samme PID-regulatoren blir brukt som vist i blokkskjemaet i (figur 9.1), og M(s) vil være lik, men prosessmodellen H_p vil være lik som figur

[9]Med forbehold om endring fra $T_{d,s} = T_z$ til $T_{d,s} = T_{i,s}$

8.10 for rullprosessen. For stampprosessen brukes prosessmodellen lik figur 8.10, men med variabelendringen forklart over figuren.

9.3.1 Regulering av rullprosessen

Skogestads metode tar i bruk den matematiske modellen for prosessen og for rullprosessen vil det bli brukt den forenklede modellen som forklart i delkapittel 8.4.3. Denne modellen er på formen:

$$H_{p,r} = \frac{K_r}{s^2}\, e^{-\tau s}\ ,\ \text{der } K_r = 45.22 \text{ og } \tau = 0.02 \tag{9.6}$$

Overføringsfunksjonen er på en annen form enn dybdeprosessen, men de samme anbefalingene om konstanter gjelder fortsatt. Dette inngår anbefalingene; $T_c = \tau$, $c = 4$ og filterkoeffisienten $T_f = 0.05 - 0.2 \cdot T_d$. Fra Skogestad sin tabell [[64], Kap. 10.3] vises det hvordan reguleringsparametrene finnes basert på formen til den forenklede matematiske modellen, ved formlene gjengitt i ligning 9.7. Disse må også gjøres om fra seriellform til parallellform ettersom regulatoren er på formen vist i figur 9.1.

$$\begin{aligned}
K_{p,s} = \frac{1}{4K_r(T_c+\tau)^2} \ &\longrightarrow\ K_{p,p} = K_{p,s}\Big(1 + \frac{T_{d,s}}{T_{i,s}}\Big) \\
T_{i,s} = c(T_c+\tau) \ &\longrightarrow\ T_{i,p} = T_{i,s}\Big(1 + \frac{T_{d,s}}{T_{i,s}}\Big) \\
T_{d,s} = c(T_c+\tau) \ &\longrightarrow\ T_{d,p} = T_{d,s}\left(\frac{1}{1 + \frac{T_{d,s}}{T_{i,s}}}\right)
\end{aligned} \tag{9.7}$$

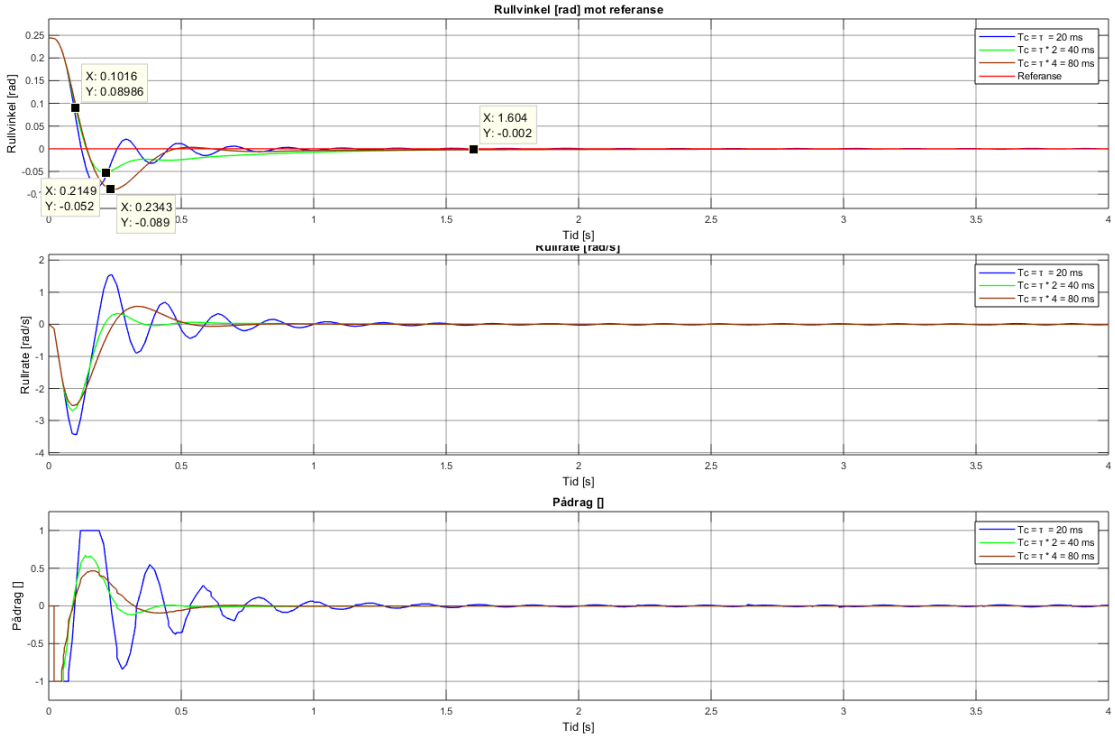
Med henhold til overføringsfunksjonen og ligningene finnes det tre sett med reguleringsparametre for å sammenligne hvilken respons som kan foretrekkes:

T_c	$K_{p,r}$	$T_{i,r}$	$T_{d,r}$
$T_c = \tau = 20\ ms$	6.911	0.320	0.080
$T_c = \tau \cdot 2 = 40\ ms$	3.071	0.480	0.120
$T_c = \tau \cdot 4 = 80\ ms$	1.728	0.800	0.200

Tabell 9.2: Tabellen viser de ulike reguleringsparametrene med ulike verdier for T_c. En lavere verdi for T_c ville gitt en veldig lav verdi for $T_{i,r}$ og høy verdi for K_p som hovedsaklig ikke er ønskelig, da risikoen er lik responsen fra figur 9.3

Disse reguleringsparametrene ble alle simulert samtidig på den ulinære rullmodellen vist i figur 8.10, ved å slippe ROV-en fra en vinkel $\phi = 0.2442\ rad$. Resultatet er vist i figur 9.10. Den blå responsen viser at systemet har store endringer i pådraget som resulterer i ujevnt rullrate og mye svingninger i selve responsen, dette er ikke er ønskelig. Alle tre responsene har omtrent like stor responstid, avlest på de øverste

datatipset til omtrent 1 sekund. Den brune responsen har 36.4% oversving, mens den grønne har 21.3 % oversving. Dette ses på som den faktoren som er viktigst å ta hensyn til i disse responsene, da forskjellen er så stor. Den grønne responsen er å foretrekke, ettersom den har mye mindre oversving, samt utnytter motorens kraft bra, ved å gi et jevnere pådrag som igjen resulterer til en jevn rullrate, som er veldig positivt.



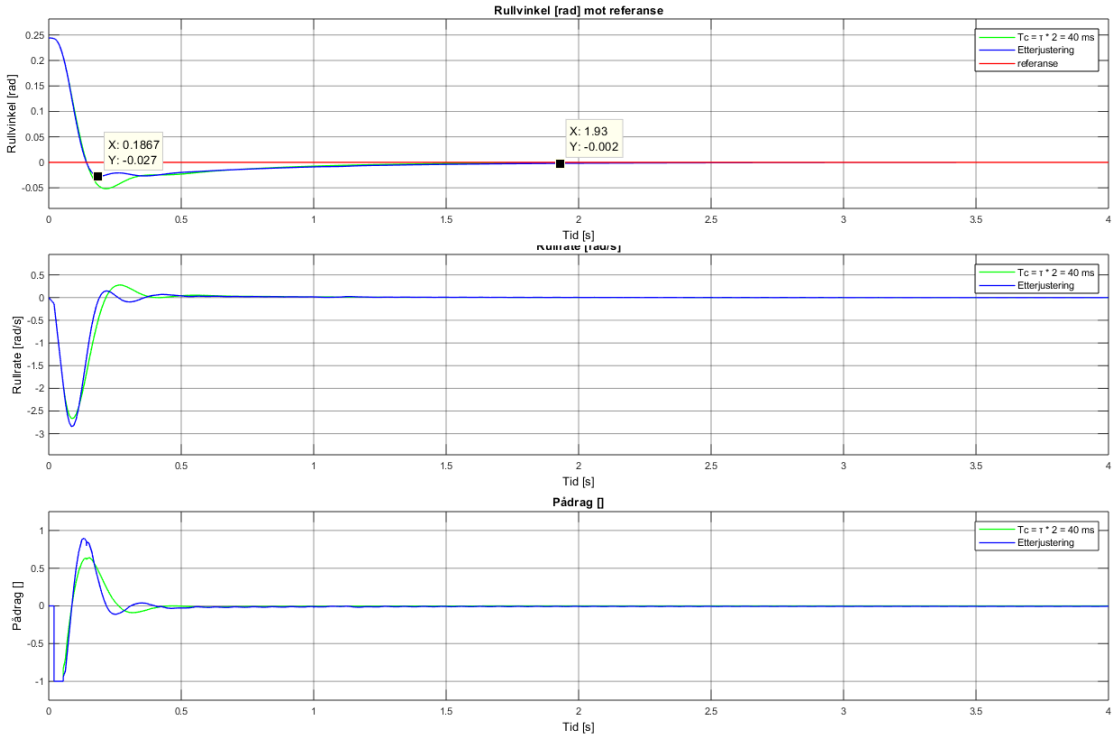
Figur 9.10: Figuren viser en sammenligning av de ulike reguleringsparametrene funnet ved forskjellige verdier for T_c fra tabell 9.2. Fra disse tre er det den grønne responsen som er til å foretrekke, selv om den brune også er brukbar. Den brune har for stort oversving til å dette systemet. Den blå responsen er alt for ujevnt samt alt for mye svingninger til å bli foretrukket.

Det er interessant å se om etterjustering av reguleringsparametrene funnet ved bruk av Skogestads metode kan forbedre responsen ytterligere. Dette gjøres ved å endre verdier for $K_{p,r}$, $T_{i,r}, T_{d,r}$ og $T_{f,r}$ for å så simulere og sammenligne ulike responser. Ved å øke verdien for $T_{i,r}$ slik at I-delen har mindre innvirkning, kan oversvinget minkes mye, men allikevel være raskt og responsivt. Problemet med dette er at dersom det kommer en ytterligere forstyrrelse vil systemet bruke lengre tid på å redusere avviket, og derfor foretrekkes det litt oversving for å kunne ha bedre kompenseringsegenskaper. De endelige parametrene ble funnet til å være: $K_{p,r} = 5.550, T_{i,r} = 0.650, T_{d,r} = 0.090$ og $T_{f,r} = 0.004$.

Figur 9.11 viser sammenligning mellom responsen fra Skogestads metode[10] (grønn) og med etterjusteringer (blå). Det er verdt å legge merke til at responsen er relativ lik når ROV-en slippes fra en vinkel, $\phi = 0.2442\ rad$. Den etterjusterte responsen

[10]Responsen med reguleringsparametrene funnet i tabell 9.2 for $T_c = 40\ ms$

har 0.3 sekund lengre stabiliseringstid, men klarer å redusere oversvinget fra 21.3 % til 11 %. Avviket er fremdeles 0.002 rad etter 1.93 sekund, men det er veldig lite og ikke merkbart praktisk, derfor anses det som at den etterjusterte responsen er å foretrekke.



Figur 9.11: Figuren viser en sammenligning av responsen funnet med Skogestads metode med bruk av $T_c = \tau \cdot 2 = 40\ ms$ og med en etterjustering av disse reguleringsparametrene. Figuren viser at etterjusteringen gir mindre oversving mot lengre stabiliseringstid. Begge responsene har relativ lik rullrate, men den etterjusterte responsen bremser litt ned på pådraget for å minke oversvinget.

Det er merkbart at pådraget går rett til -1 når ROV-en blir sluppet, dette hjelper til å nå referanse verdien raskere. Når referanse verdien er passert og oversvinget har begynt, så går pådraget motsatt vei for å stabilisere ROV-en mot referansen. Den etterjusterte regulatoren bruker ikke like mye pådrag som responsen fra Skogestads metode, men det resulterer i omtrent like jevnt pådrag som gir omtrent lik rullrate.

9.3.2 Regulering av stampprosessen

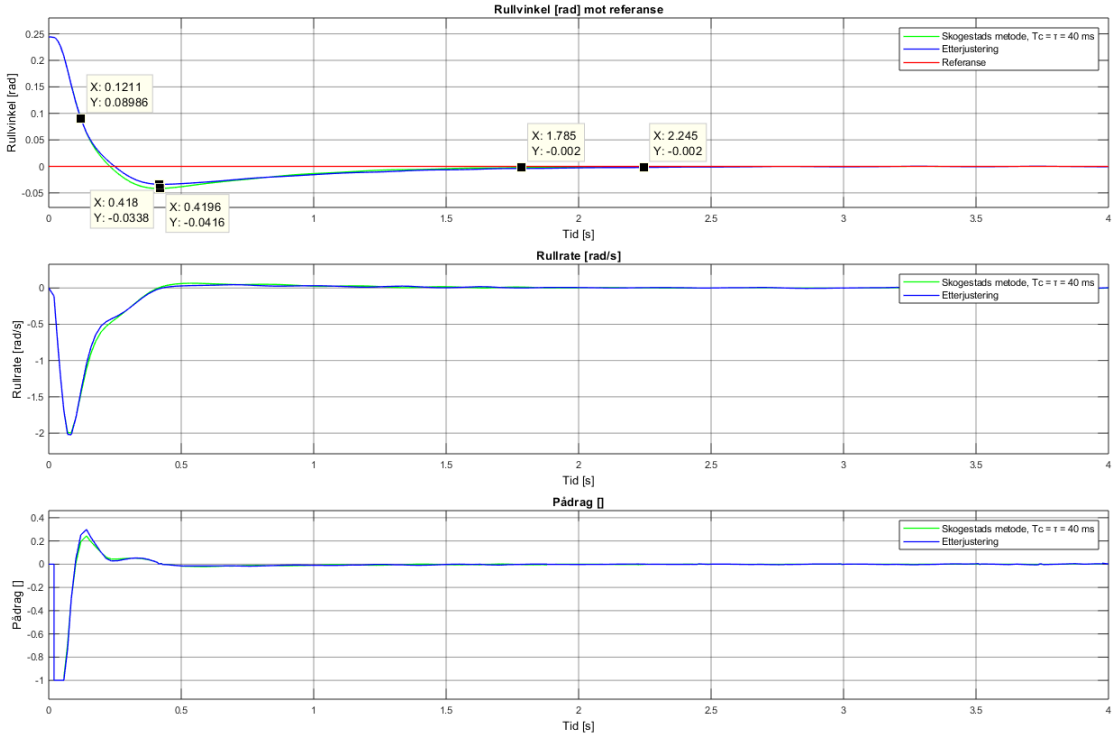
Stampprosessen har samme fremgangsmåte som rullprosessen, men den matematiske modellen har små forskjeller som gjør at reguleringsparametrene og oppførselen i stampretning blir annerledes. Den forenklede matematiske modellen som brukes for å finne reguleringsparametrene gjengis:

$$H_{p,s} = \frac{K_s}{s^2}\, e^{-\tau s} \;=\; \frac{42.87}{s^2}\, e^{-0.02s} \tag{9.8}$$

Ved bruk av ligning 9.7 kan reguleringsparametrene for stampprosessen bli funnet. Det gjøres simuleringer for ulike verdier for T_c i dette tilfellet også. Ettersom etterjus-

teringen av parametrene funnet fra Skogestads metode for rullprosessen ga gode resultater, er det ønskelig å teste det samme for stampprosessen. Reguleringsparametrene for $T_c = \tau \cdot 2 = 40\ ms$ ga parametrene; $K_{p,s} = 3.240, T_{i,s} = 0.480, T_{d,s} = 0.120$ og $T_{f,s} = 0.012$. Disse verdiene ga en veldig grei respons og et godt utgangspunkt for etterjustering.

I figur 9.12 vises det sammenligning av den foretrukne responsen fra Skogestads metode (grønn) med den etterjusterte responsen (blå) med reguleringsparametrene; $K_{p,s} = 3.490, T_{i,s} = 0.600, T_{d,s} = 0.120$ og $T_{f,s} = 0.012$. Pådraget og stampraten er relativ lik i begge tilfellene, og dette resulterer i at responstiden er veldig lav og tilnærmet lik for begge responsene, som vist i det øverste datatipset til venstre. Den etterjusterte responsen har 3 % lavere oversving, men 0.46 sekund lenger stabiliseringstid.



Figur 9.12

Den etterjusterte responsen ga en grei respons, men svært lik responsen funnet fra Skogestad metode. Begge disse responsene er å foretrekke, men responsen fra Skogestads metode er nok den mest ideele. At responsen har 3 % mer oversving for å nå en stasjonær verdi 0.46 sekund raskere er svært ønskelig.

9.3.3 Konklusjon av rull- og stampprosessen

Det ble vist i kapittelet om den matematiske modellen (ref. figur 8.13) at uten en regulator vil ROV-en svinge seg inn slik at rullvinkelen blir omtrent lik 0 rad, men aldri helt null. Videre ble det antatt at når piloten kjører ROV-en ønsker han å kjøre med en rull- og stampvinkel lik 0 rad og rull- og stamprate lik 0 $\frac{rad}{s}$. Det betyr at disse regulatorene ofte kommer til å regulere små avvik, men det er alltid en

fordel å ha gode kompenseringsegenskaper slik at det er robusthet mot større avvik. Det er dette som er utgangspunktet for konklusjonen for regulatorene for rull- og stampprosessene.

For rullprosessen konkluderes det med at Skogestads metode ga tilfredstillende responser for reguleringsparametre rundt $T_c = \tau \cdot 2 = 40\ ms$, men at etterjustering av parametrene ga en totalt sett bedre respons for både mindre og større avvik. Det vil videre bli tatt utgangspunkt i reguleringsparametrene fra den etterjustere responsen før testing og tuning på en ferdig ROV. Reguleringsparametrene er som følger:

$$K_{p,r} = 5.550 \qquad T_{i,r} = 0.650 \qquad T_{d,r} = 0.095 \qquad T_{f,r} = 0.004$$

For stampprosessen konkluderes det med at det hovedsaklig ikke var stor forskjell mellom de to responsene, utenom at responsen fra Skogestads metode ga omtrent 3 % mer oversving og 0.46 sekund raskere stabiliseringstid enn den etterjusterte. Med tanke på dette så vil det videre bli tatt utgangspunkt i reguleringsparametrene funnet fra Skogestads metode før testing og tuning på en ferdig ROV. Reguleringsparametrene er som følger:

$$K_{p,s} = 3.240 \qquad T_{i,s} = 0.480 \qquad T_{d,s} = 0.120 \qquad T_{f,s} = 0.012$$

Dette vil si at disse reguleringsparametrene er kun et utgangspunkt for videre testing og tuning. Ved testing og tuning vil en kunne justere parametrene dersom en ønsker en annerledes respons eller om reguleringsparametrene ikke gir den responsen som er mest egnet for ROV-en.

9.4 Implementering på mikrokontroller

For å implementere PID-regulatoren som beskrevet tidligere i kapittelet på en mikrokontroller, er det nødvendig å ta i bruk forskjellige metoder for diskretisering av regulatoralgoritmen. Analytisk kan PID-regulatoren skrives som vist i ligning 9.1:

$$u(t) = K_p \cdot e(t) \ + \ \frac{K_p}{T_i} \int_0^t e(t)dt \ + \ K_p \cdot T_d \cdot \frac{d}{dt}(e_f(t)) \tag{9.9}$$

P-leddet

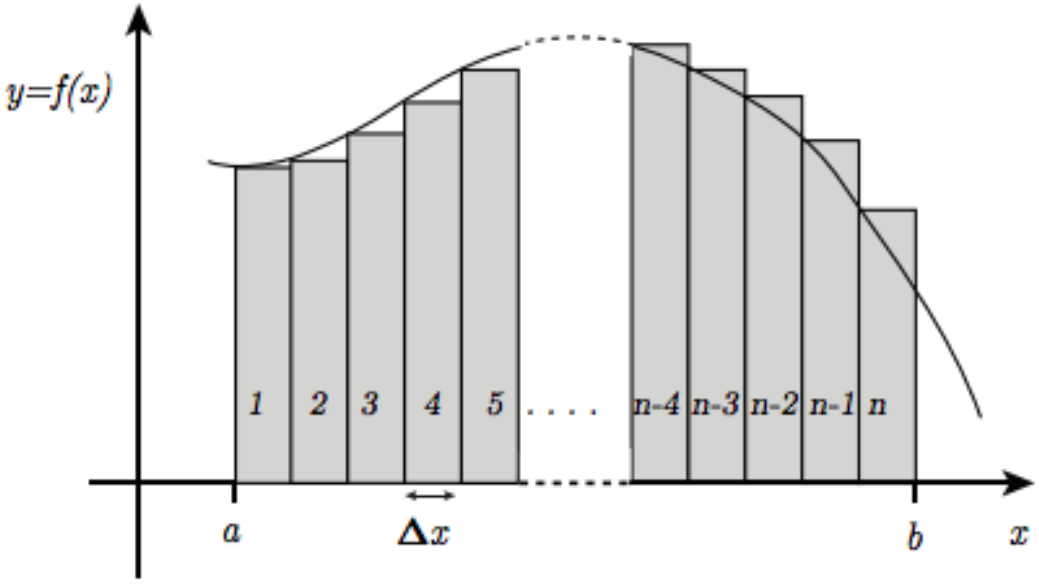
Fra ligning 9.9 over kan en se at P-leddet, også kalt proporsjonalleddet, består kun av å multiplisere K_p-verdien med avviket. Det er derfor ikke nødvendig å diskretisere mer enn det allerede er, og pådraget $u_p(t)$ kan dermed skrives:

$$u_p(t) = K_p \cdot e(t) \tag{9.10}$$

I-leddet

Et integral er arealet under en funksjon f(x) fra x = a til x = b. Når man beregner integral i matematikken, vil en ha eksakte (analytiske) løsninger. Som vist i ligning

9.9 er dette lik $\frac{K_p}{T_i}\int_0^t e(t)dt$. I praksis er dette vanskelig å få til, og en kan da heller bruke en metode som kalles *numerisk integrasjon*. Det går ut på å dele arealet opp i n rektangler med lik bredde og summere arealene av disse.



Figur 9.13: Prinsippet for numerisk integrasjon.

For at dette skal fungere i praksis når en skal beregne integralet av avvikene for dybde, rull og stamp, kan en enkelt bestemme at $u_I(k-1)$ er summen av alle tidligere verdier for u_I, for så å addere med u(k). Da tidsskrittet mellom målingene i våres tilfelle alltid vil være like, er det ikke nødvendig å ta hensyn til tidsskritt. Pådraget $u_I(t)$ kan dermed skrives:

$$u_I(t) = u_I(t-1) + e(t) \tag{9.11}$$

Videre blir dette integralet begrenset til å ikke overstige $integratorbegrensning = \frac{u_{maks}}{T_i}$, som forklart tidligere i kapittelet.

D-leddet

D-leddet finnes ved å ta et estimat av endringsraten til e(t). Som vist i ligning 9.9 er dette $K_p \cdot T_d \cdot \frac{d}{dt}(e_f(t))$. Da oppdateringsfrekvensen på PID-regulatoren i mikrokontrolleren som sagt har en fast verdi er det ikke nødvendig å diskretisere algoritmen for å estimere denne verdien. Istedenfor kan en enkelt beregne endringsraten ved å subtrahere det forrige beregnede avviket fra det nye målte avviket, og får dermed pådraget $u_D(t)$:

$$u_D(t) = e(t) - e(t-1) \tag{9.12}$$

Videre blir avviksverdiene først filtrert ved bruk av et IIR-filter før det deriveres, for å unngå store hopp ved bråe endringer og/eller støy. Et IIR-filter tar inn forrige filtrerte verdi og ny målt verdi, og setter en prioritering på hvilke av verdiene som skal ha høyest betydning for den filtrerte verdien. Et godt utgangspunkt er å sette

denne parameteren til 0.1 som beskrevet i delkapittel 9.1, som dermed beregner filtrert verdi ved å multiplisere ny måling med 0.1, og addere med forrige filtrerte målingen med 1-0.1.

Oppsummert kan alle leddene bli implementert ved følgende kode:
filnavn: main.c

```
1  // PID-regulatoren returnerer alle de tre P, I, og D paadragene
       summert. Paadragsomraade for selve PID-regulatoren er -1 til
       1, mens paadragsomraade for thrusterne er 11000-19000 med
       nullpunkt i 15000, altsaa 4000 opp/ned fra middelverdi, som
       beskrevet i kapittel 7 i rapporten. Dermed multipliseres PID-
       regulatorens nye verdi med 4000.
2  float pid_regulator(float error, float integral, float derivat,
       float KP, float KI, float KD) {
3    return 4000*(KP*error)+(KI*integral)+(KD*derivat); // ny
       paadragsverdi.
4    // KP = Kp
5    // KI = Kp/Ti
6    // KD = Kp*Td
7  }
8
9  // Integral estimerer integralet ved aa addere nytt avvik med
       summen av alle tidligere avvik.
10 float integral(float siste_integral, float error, float
       integratorbegrensning) {
11   float integral;
12   integral = siste_integral + error;
13
14   // Begrens integralverdien til integratorbegrensning = P_maks
       / K_I
15   if (integral > integratorbegrensning) {
16     integral = integratorbegrensning;
17   } else if (integral < -integratorbegrensning) {
18     integral = -integratorbegrensning;
19   }
20   return integral;
21 }
22
23 // Derivat estimerer endringsraten til avviket ved aa subtrahere
       forrige avvik fra nytt maalt avvik. En finner dermed hvor
       mye avviket har endrer seg siden sist maaling.
24 float derivat(float error, float siste_error) {
25   return error - siste_error;
26 }
27
28 // Error finner avviket ved aa subtrahere maalt verdi fra onsket
       verdi.
29 float error(float settpunkt, float maalt_verdi) {
30   return settpunkt - maalt_verdi;
31 }
32
33 // IIR_filter filtrerer avviket for aa fjerne store hopp som
       resultat av stoy og eller braae bevegelser.
34 float IIR_filter(float forrige_filtrerte_avvik, float nytt_avvik
       ) {
35   return 0.1*nytt_avvik + (1-0.1)*forrige_filtrerte_avvik;
36 }
```

9.4.1 Implementering av dybde-, rull- og stampprosessen

Implementeringen av dybde-, rull- og stampprosessen er som vist i koden under, se forklarende kommentarer:

filnavn: main.c

```
// Sjekk om dybderegulatoren er aktiv
if (dybde_regulator == 1) {
    // Finner avvik, integral og derivat
    dybde_error = error(settpunkt_dybde, dybde);
    dybde_integral = integral(dybde_integral, dybde_error, integratorbegrensning_dybde);

    // Filtrer avviket
    dybde_filtrert_error = IIR_filter(siste_dybde_filtrert_error, dybde_error);
    dybde_derivat = derivat(dybde_filtrert_error, siste_dybde_filtrert_error);

    // Setter siste_dybde_error lik nytt maalt avvik og siste_dybde_filtrert_error lik nytt maalt avvik filtrert, til neste iterasjon.
    siste_dybde_filtrert_error = dybde_filtrert_error;

    // Beregner ny paadragsverdi for dybderegulatoren vha. regulator.
    dc_dybde = pid_regulator(dybde_error, dybde_integral, dybde_derivat, KP_dybde, KI_dybde, KD_dybde);
} else {
    dc_dybde = 0;
}

// Lik fremgangsmaate som for dybdeprosesen, med stampprosessens variabler og parametrer.
if (stamp_regulator == 1) {
    stamp_error = error(settpunkt_stamp, stamp);
    stamp_integral = integral(stamp_integral, stamp_error, integratorbegrensning_stamp);

    stamp_filtrert_error = IIR_filter(siste_stamp_filtrert_error, stamp_error);
    stamp_derivat = derivat(stamp_filtrert_error, siste_stamp_filtrert_error);

    siste_stamp_filtrert_error = stamp_filtrert_error;

    dc_stamp = pid_regulator(stamp_error, stamp_integral, stamp_derivat, KP_stamp, KI_stamp, KD_stamp);
} else {
    dc_stamp = 0;
}

// Lik fremgangsmaate som for dybdeprosesen, med rullprosessens variabler og parametrer.
```

```
36 if (rull_regulator == 1) {
37     rull_error = error(settpunkt_rull, rull);
38     rull_integral = integral(rull_integral, rull_error,
   integratorbegrensning_rull);
39
40     rull_filtrert_error = IIR_filter(siste_rull_filtrert_error,
   rull_error);
41     rull_derivat = derivat(rull_filtrert_error,
   siste_rull_filtrert_error);
42
43     siste_rull_filtrert_error = rull_filtrert_error;
44
45     dc_rull = pid_regulator(rull_error, rull_integral,
   rull_derivat, KP_rull, KI_rull, KD_rull);
46 } else {
47     dc_rull = 0;
48 }
```

Videre blir disse pådragsverdiene sammenslått med korrekte fortegn for hver av de vertikale thrusterne, og sendt inn i **max_change_fn** som beskrevet i kapittel 7.6.1. Setting av korrekte fortegn for vertikale thrustere blir bestemt av følgende logiske faktum, gitt av at alle vertikale thrustere har samme kraftretning for et positivt pådrag:

- Alle thrusterne skal ha samme fortegn for bevegelse i dybderetning.
- Sett ovenfra med foroverretning pekende rett frem, skal thrusterne på venstre side ha motsatt fortegn enn thrusterne på høyre side, for bevegelse i rullretning.
- Sett ovenfra med foroverretning pekende rett frem, skal thrusterne fremme ha motsatt fortegn enn thrusterne bak, for bevegelse i stampretning.

En får dermed ligningssettet:

$$\begin{aligned} u_{vfv} &= u_{dybde} + u_{stamp} + u_{rull} \\ u_{vfh} &= u_{dybde} + u_{stamp} - u_{rull} \\ u_{vbh} &= u_{dybde} - u_{stamp} - u_{rull} \\ u_{vbv} &= u_{dybde} - u_{stamp} + u_{rull} \end{aligned} \tag{9.13}$$

Dette blir videre implementert på mikrokontroller ved følgende kode:
filnavn: main.c

```
1 dc_vfv = max_change_fn(TIM2->CCR1, (dc_dybde + dc_stamp +
     dc_rull) + standby, max_change, standby);
2 dc_vfh = max_change_fn(TIM2->CCR2, (dc_dybde + dc_stamp -
     dc_rull) + standby, max_change, standby);
3 dc_vbh = max_change_fn(TIM2->CCR3, (dc_dybde - dc_stamp -
     dc_rull) + standby, max_change, standby);
4 dc_vbv = max_change_fn(TIM2->CCR4, (dc_dybde - dc_stamp +
     dc_rull) + standby, max_change, standby);
```

9.4.2 Konklusjon av implementering på mikrokontroller

Hensikten med dette delkapittelet var å implementere automatisk justering av pådrag for å holde en ønsket referanseposisjon/vinkel ved hjelp av en PID-regulator, basert på P-, I- og D-parametrene funnet tidligere i kapittelet. Ved bruk av CubeIDE's feilsøkingsverktøy kan en overvåke programmets verdier etter en ønsket tid, og dermed bekrefte om programmet fungerer som det skal. Ved å legge inn et konstant avvik på 0.3 meter for ønsket dybde, får en etter 600 iterasjoner (tilfeldig valgt) som tilsvarer 6.0 sekunder ved 100 iterasjoner per sekund, følgende pådragsverdier for vertikale motorer:

(x)= counter	double	600
(x)= dybde_error	float	-0.300000191
(x)= siste_dybde_error	float	-0.300000191
(x)= dybde_integral	float	-179.401062
(x)= dc_vfv	float	13934.6357
(x)= dc_vfh	float	13934.6357
(x)= dc_vbv	float	13934.6357
(x)= dc_vbh	float	13934.6357
(x)= dc_dybde	float	-1065.36389

Figur 9.14: Skjermbilde av variabler i CubeIDE's feilsøkingsverktøy.

I figur 9.14 ser en de relevante variablene og deres verdi for regulering i dybdeprosessen, med betydningene:
- **counter** er en teller som inkrementeres for hver gang PID-regulatoren beregner ny pådragsverdi.
- **dybde_error** er avviket i dybderetning.
- **siste_dybde_error** er avviket fra forrige iterasjon.
- **dybde_integral** er oppbygd integralverdi for I-leddet til PID-regulatoren i dybderetning. Integratorbegrensning er i dette tilfellet $\frac{4000}{3.36} = 1190$.
- **dc_...** er ny ønsket CCRx verdi for de korresponderende timerkanalene. Disse blir rundet av til nærmeste heltall, i dette tilfellet 13935.
- **dc_dybde** er pådraget for PID-regulatoren for dybderetning.

Middelverdi for motorene er som tidligere forklart 15000, og maksimal revers er 11000. Dette gir at PID-regulatoren gir $\frac{1065}{4000} \cdot 100\% = 26.6\%$ pådrag etter 6.0 sekunder. Det kan derfor konkluderes med at PID-regulatoren ser ut til å fungere som den skal, og gir et godt utgangspunkt for eventuell finjustering og tuning ved testing av en fysisk ROV i reelle omgivelser med reell sensordata. Det er også implementert en pådragsbegrensing som gjør at dersom regulatoren(e), for eksempel grunnet et veldig stort avvik i referansen får et utslag større enn maksimalt pådrag på ± 4000, vil pådraget bli satt til 4000. Dette sørger for at PWM-signalet alltid er innenfor tillat på-tid på 1100 til 1900 μs.

Kapittel 10

Testing og tuning

Innhold

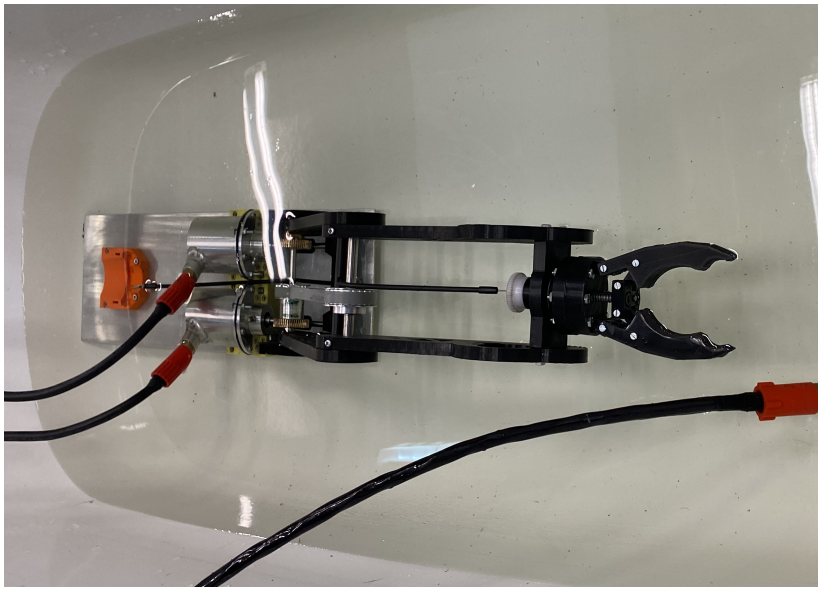
10.1 Testing av manipulatorarm 146
10.2 Testing av styrings- og reguleringssystem 148

Ved hjelp av Cube-IDE's feilsøkingsverktøy er det verifisert at alle funksjoner beskrevet i rapporten gir korrekte resultater, samt at setting av PWM- og PFM-signaler fungerer som de skal, ref. delkapittel 6.4 og delkapittel 9.4.2. Sensorgruppen [2] har verifisert at SPI-kommunikasjon fungerer, og programstrukturen for styring og regulering er nøye gjennomgått.

Testing av manipulatorarm er gjennomført, og blir beskrevet i dette kapittelet. Testing av hele systemet satt sammen på den fysiske ROV-en er dessverre ikke blitt gjennomført, da det i skrivende stund mangler kritiske deler for at dette skal være mulig. Det blir derfor lagt en plan for hva som skal testes og hvordan det skal testes, samt hvordan regulatorene eventuelt skal tunes, for å oppnå best mulig resultat.

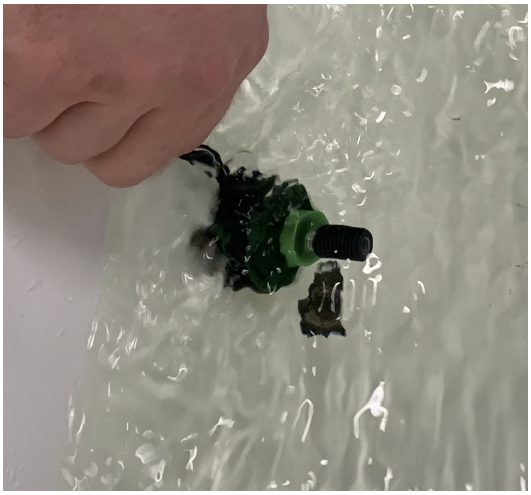
10.1 Testing av manipulatorarm

Testing av manipulatorarmen ble gjennomført sammen med Manipulatorgruppen [17]. Dette ble gjort i et badekar på verkstedet ved Universitetet i Stavanger. Hensikten med testen var å funksjonsteste de ulike motorene på ulike ledd, samt få en indikasjon på vanntettingen til motorkapslingene, som Manipulatorgruppen hadde ansvar for at var i orden. Motorene for klyp-funksjonen og rotasjon av klypen ble ikke testet på selve manipulatorarmen i denne testen, da det var noen mekaniske deler på disse som fortsatt var under produksjon. Disse ble dog testet i ettertid når de var ferdig montert på manipulatoren. Se figur av oppsett under:



Figur 10.1: Manipulatorarmen plassert under vann, i et badekar på verkstedet ved Universitetet i Stavanger. Motor for klype og rotasjon av klype er ikke montert, da det manglet noen fysiske deler for at disse kunne monteres. Testen gir likevell en god indikasjon på vanntetthet, samt bevegelse av øvre og nedre ledd.

Figur 10.2: Motor for klype-funksjon, plassert under vann i samme test som figur over.



Figur 10.3: Motor for rotasjon av klype, plassert under vann i samme test som figur over.

Ved å sette opp PFM-signaler som beskrevet i kapittel 6 samt å sette en GPIO-pinne høy/lav, fikk vi testet alle motorene i begge retninger, ved å sende høyt/lavt signal til DIR-pinnen på motorkontrolleren som bestemmer retning, og PFM-signaler til PUL-pinnen på motorkontrolleren som gir et steg på motoren per puls på signalet. Motorene ble testet i forskjellige hastigheter, og med noe løsning/stramming på skruer på selve manipulatoren for å gi mindre friksjon i girene, fikk vi mer enn tilfredstillende resultater. I kapittel 1 ble det spesifisert at det tyngste objektet manipulatorarmen skal løfte er 0.5 kg. Ved å plassere en gjenstand med tyngde 2.5 kg i klypen, fikk vi også testet bevegelse i alle ledd med $\frac{2.5}{0.5} \approx 5$ ganger så mye motsand som en vil møte i MATE-konkurransen.

Med disse testresultatene konkluderes det med at motorene ga tilfredsstillende resul-

tater for bevegelse i alle ledd. Ved å teste med gjenstanden med tyngde 2.5 kg, fikk vi også bekreftet at alle ledd har mer enn nok holdemoment for de ulike oppgavene manipulatorarmen skal gjennomføre. I kapittel 5 ble det også forklart at alle leddene kunne kjøres til endeposisjon uten at det ville bli noe form for skader. Dette ble også testet og resultatet var som forventet. Alt i alt har testingen av manipulatorarmen gitt oss resultatene vi ønsket, og kan til slutt konkludere med at den fungerer til sin hensikt.

Til videre arbeid er det ønskelig å teste manipulatoren på ferdig montert ROV. Testingen forklart over gir oss et godt grunnlag for at manipulatorarmen skal fungere godt når den blir montert på, men dette er ønskelig å teste når hele systemet er komplett. Videre testing vil da bestå av generell bruk av manipulatorarmen rettet mot MATE-konkurransen.

10.2 Testing av styrings- og reguleringssystem

For å få et godt utgangspunkt til MATE-konkurransen og eventuelle andre bruksområder, er det viktig å teste systemet og alle dets funksjoner så godt som mulig. Det tas her utgangspunkt i at selve koden blir avluset[1] for syntaksfeil og lignende i forkant av test for styrings- og reguleringssystemene, og det blir derfor videre kun gjennomgått testplan for dette.

Det er i prosessen for beregninger av reguleringsparametrer gjort en del antagelser om den ferdige ROV-en, som spiller inn på resultatet. Dette gjelder hovedsaklig dragkoeffisienten $C_d = 1$, ROV-ens masse $m = 35kg$ og avstand mellom massesenter og volumsenter, $\Delta s = 0.09m$. Det er også antatt at massesenter og volumsenter er direkte overfor hverandre, som i teorien vil føre til at opprettningskraften alene vil føre ROV-en til en helt horisontal posisjon. I virkeligheten vil disse etter all sannsynlighet være noe annerledes, og en vil derfor få noe ulike dynamiske egenskaper enn det som er simulert i denne rapporten. Ved ulik plassering i x- og y-planet for masse- og volumsenter, vil en måtte gi et konstant pådrag i ulike retninger på vertikale motorer for å holde ROV-en horisontal. Hvor stort avvik det er i disse antagelsene er interessant å vite, og det vil derfor bli målt før selve testen av regulatorene. Dersom det er store avvik vil det bli funnet nye parametrer, ved å endre relevante variablene i formler funnet i kapittel 8 og kapittel 9.

Dragkoeffisienten vil være ganske vanskelig å måle, og minst like vanskelig å finne eksperimentelt uten veldig avansert utstyr. Det er derfor noe usikkerhet i parametrene funnet, selv med reelle målinger som forklart over. Det er derfor sannsynlig at reguleringsparametrene må endres eksperimentelt under testing. På grunn av dette er reguleringsparametrene implementert på mikrokontrolleren definert som variabler, og vil være mulige å oppdatere fra datamaskin på toppsiden av systemet. Dette er mulig grunnet kommunikasjonssystemet utviklet av Kommunikasjonsgruppen [59]. En slipper da å måtte ta opp ROV-en, demontere elektronikkhuset, koble seg til mikrokontrolleren og oppdatere programvaren, for å endre reguleringsparametrer. Dette vil gi mulighet for å teste mange flere reguleringsparametrer i løpet av kortere

[1] 'Debugged' / feilsøkt

tid, som vil føre til et større grunnlag for valg av endelige parametrer.

Ved bruk av samme kommunikasjonssystem som nevnt over, vil alle sensordata, pådragsverdier og ellers interessante data, bli sendt kontinuerlig til toppsiden. Dette blir logget i sanntid på et grafisk brukergrensesnitt utviklet av Bildegjenkjenning og autonom kjøring gruppen [70], som gir god innsikt i hvor godt en regulator virker etter sin hensikt. På denne måten kan det sammenlignes med simuleringer gjort i Simulink, samt at en kan sammenligne målinger ved kjøring uten regulering, P-regulering, PI-regulering og PID-regulering.

Kapittel 11

Diskusjon

Som deltaker i et studentprosjekt som UiS Subsea er det flere verv som skal utføres utover det en vanlig bacheloroppgave krever. I den sammenheng ble Markus Lægreid Haldorsen utnevnt som lagleder elektro og Edmond Baloku Vigre som økonomiansvarlig. Som resultat av dette har det medgått mye arbeid utover det som er dokumentert i denne rapporten. I tillegg til dette er det mye som må gjøres for at så mange grupper skal ende opp med et enkelt produkt der alle systemene virker sammen. Det er derfor gått mye tid til å planlegge og samarbeide, assistere hverandre der det trengs, testing av systemer og lignende.

11.1 Motorer og motorkontrollere

Det ble valgt å bruke 8 børsteløse likestrømsmotorer av typen P1000 THRUSTME [16] for fremdrift av selve ROV-en. De er mye kraftigere enn motorer brukt i tidligere år, samt at de har tilhørende vanntette motorkontrollere, som gir mulighet for å plassere disse på utsiden av elektronikkhuset. Det gir fordeler for både plass og varmgang i elektronikkhuset, samt mye mer effektiv kjøling for motorkontrollerne selv, da de er i direkte kontakt med vann.

Videre ble det valgt å gjenbruke 2 stegmotorer av typen Nema 17, en stegmotor av typen Nema 14 og en børsteløs likestrømsmotor av typen Multistar Elite 3508 for styring av manipulatorarm. Stegmotorene styrer henholdsvis manipulatorens nedre arm 5.1.1, øvre arm 5.1.2, klype 5.1.3 og rotasjon av klypen 5.1.4.

UiS Subsea har ved tidligere år hatt en rigg som har blitt montert ved en bassengkant, for å gjøre en fullverdig test av motorenes oppførsel under vann. Denne riggen var ikke å finne etter stor leteaksjon, og på grunn av tidsbegrensningen i prosjektet ble det ikke fokusert på å lage en ny rigg. En slik fullverdig test ville gitt oss et større og bredere bilde på motorenes oppførsel og kapasitet i forhold til testingen som ble gjort i badekaret ved verkstedet på UiS. Det ble dog gjort en test som forklart i kapittel 3, som gav tilfredsstillende resultater, men med større usikkerhetsmomenter.

11.2 Styringssystem

Manuell styring av ROV-en gjennomføres ved bruk av to Xbox360-kontrollere [53] - en for styring av fremdrift, og en for styring av manipulator. Styringsdata fra disse blir lest, behandlet og videresendt fra toppsiden av systemet. De blir så mottatt av kraftforsyningskortet i selve ROV-en, der kode for styring og regulering er implementert. Koden tolker signalene mottatt fra toppsiden, beregner motorpådrag og generer PWM- og PFM-signaler for styring av motorene.

11.3 Modellering og regulering

Piloten har mulighet for å styre ROV-en i fire av seks frihetsgrader, henholdsvis x-, y- og z-retning for translatorisk bevegelse, og gir-retning for rotasjonsbevegelse. Det er ønskelig å legge til rette for piloten slik at hen kan kjøre ROV-en på en optimal og enklest mulig måte, og det er dermed ikke noe behov for at piloten skal måtte håndtere bevegelse i rull- og stampretning. Dette var grunnlaget for at det ble valgt tre PID-regulatorer for henholdsvis dybde-, rull- og stamp. Fra sensorkortet får regulatorene tilsendt dybde, rullvinkel og stampvinkel, slik at de har et avvik å følge.

11.3.1 Modell og regulator for dybdeprosessen

Den matematiske modellen ga overføringsfunksjonen mellom dybde og pådrag gjengitt under:

$$H_{p,z}(s) = \frac{z(s)}{u(s)} = \frac{\frac{400\gamma}{\rho C_d A|v_{z,A}|}}{s\left(\frac{m}{\rho C_d A|v_{z,A}|}s + 1\right)} e^{-\tau s} \Bigg|_{v_{z,A}=0.10} = \frac{14.25}{s(1.60s+1)} e^{-0.02s}$$

Det ble bevist ved to metoder at dybdeprosessen er meget ulineær, da vannmotstanden er avhengig av den kvadrerte hivfarten. Det var derfor gunstig å lage et reguleringssystem i det mest kritiske arbeidspunktet ROV-en vil arbeide i, som i dette tilfellet er antatt å være $v_{z,A} = 0.10\frac{m}{s}$. Med dette arbeidspunktet blir prosessforsterkningen $K_z = 14.25\ m$ og tidskonstanten $T_z = 1.60\ s$.

Basert på at dybdeprosessen var svært ulineær og at prosesskonstantene dermed varierer med arbeidspunktet $v_{z,A}$, ble det drøftet om det ville være gunstig å ta i bruk metoden kalt *parameterstyring*, for å tilpasse reguleringsparametrene i forhold til arbeidspunktet ROV-en befinner seg i. Det ble konkludert med at det ikke vil bli implementert parameterstyring, ettersom ROV-ens arbeidspunkt under MATE konkurransen mest sannsynlig ikke vil være mye høyere enn 0.1 $\frac{m}{s}$, samt at det ville vært unødvendig tidskrevende å implementere en såpass avansert metode, for minimale forbedringer i systemet.

Med Skogestads metode ble det funnet tilfredstillende reguleringsparametre etter en etterjustering av den anbefalte tidskonstanten T_c og en ny vurdering av ligningene i ligning 9.5. Reguleringsparametrene ble funnet med bakgrunn fra den matematiske modellen for dybdeprosessen. Følgende parametre vil bli implementert på mikrokontrolleren som et utgangspunkt:

$$K_{p,d} = 0.334 \qquad T_{i,d} = 3.360 \qquad T_{d,d} = 0.840 \qquad T_{f,d} = 0.084$$

Ved et sprang i referansen på 0.1 m ga disse parametrene en responstid på 0.307 sekund, et oversving på 18 % og en stabiliseringstid på 5.138 sekunder.Ved et sprang på 1 meter i referansen ble responstiden 0.905 sekund, oversvinget 18.8 % og stabiliseringstiden omtrent 11.15 sekund. Alt i alt ga dette tilfredstillende oppførsel for dybderegulatoren. Det presiseres at dette baseres på simuleringer, og at en vil få andre resultater i virkeligheten. Hva som forventes av virkelig oppførsel og tiltak for dette ble tatt rede for i kapittel 10.

11.3.2 Modell og regulator for rull- og stampprosessen

Den matematiske modellen ga overføringsfunksjonen mellom rull- og stampvinkel og pådrag gjengitt under, der overføringsfunksjonnen for rull er øverst og stamp nederst:

$$H_{p,\phi} = \frac{\phi(s)}{u(s)} = \frac{\left(\frac{400\gamma r_{m,r}}{mg\Delta s}\right)}{\left(\frac{w^2+h^2}{12g\Delta s}\right)s^2 + \left(\frac{3d\rho C_d p_A (w^4+h^4)}{24mg\Delta s}\right)s + 1} e^{-\tau s}$$

$$H_{p,\theta} = \frac{\theta(s)}{u(s)} = \frac{\left(\frac{400\gamma r_{m,s}}{mg\Delta s}\right)}{\left(\frac{d^2+h^2}{12g\Delta s}\right)s^2 + \left(\frac{3d\rho C_d q_A (d^4+h^4)}{24mg\Delta s}\right)s + 1} e^{-\tau s}$$

Basert på antagelsene som ble gjort under konklusjonen om rull- og stampprosessene i delkapittel 8.4.3 ble overføringsfunksjonene forenklet. Antagelsene baserte seg på at det naturlige oppretningsbidraget og vannmotstanden ikke vil spille stor rolle når piloten styrer ROV-en i små hastigheter, ettersom vinkelen og vinkelhastigheten er tilnærmet null. Overføringsfunksjonene ble derfor forenklet som vist under:

$$\begin{aligned} H_{p,r} = \frac{\phi(s)}{u(s)} &= \frac{\frac{4800\gamma r_{m,r}}{m(w^2+h^2)}}{s^2} e^{-\tau s} = \frac{45.22}{s^2} e^{-0.02s} \\ H_{p,s} = \frac{\theta(s)}{u(s)} &= \frac{\frac{4800\gamma r_{m,s}}{m(d^2+h^2)}}{s^2} e^{-\tau s} = \frac{42.87}{s^2} e^{-0.02s} \end{aligned} \tag{11.1}$$

De forenklede modellene var et godt grunnlag til å bruke Skogestads metode for rull- og stampprosessen ettersom denne metoden ga god og tilfredstillende oppførsel for dybderegulatoren. Det var ønskelig å ta i bruk reguleringsparametrene funnet med den mest ønskelige oppførselen funnet ved Skogestads metode, og teste om etterjustering av disse parametrene ga en mer tilfredstillende oppførsel.

For rullprosessen ga en etterjustering av parametrene en mer tilfredstillende respons enn med parametrene funnet fra Skogestads metode. De nye og etterjusterte reguleringsparametrene som blir tatt utgangspunkt i ved implementering på mikrokontroller er:

$$K_{p,r} = 5.550 \qquad T_{i,r} = 0.650 \qquad T_{d,r} = 0.095 \qquad T_{f,r} = 0.004$$

Ved å slippe ROV-en fra en rullvinkelen $\phi = 0.2442\ rad$, fikk regulatoren en responstid på 0.102 sekund, et oversving på 11 % og en stabiliseringstid på omtrent 1.930 sekund, i simuleringer. Igjen presiseres det at dette er basert på simuleringer, og at en vil få andre resultater i virkeligheten, som forklart i kapittel 10.

For stampprosessen var oppførselen fra reguleringsparametrene funnet ved Skogestads metode og oppførselen fra de etterjusterte reguleringsparametrene relativ lik. Det ble dog et mindre oversving med den etterjusterte oppførselen, og dette var å

foretrekke. De etterjusterte parametrene som blir tatt utgangspunkt i på mikrokontrolleren er:

$$K_{p,s} = 3.240 \qquad T_{i,s} = 0.480 \qquad T_{d,s} = 0.120 \qquad T_{f,s} = 0.012$$

Ved å slippe ROV-en fra en stampvinkelen $\theta = 0.2442\ rad$, fikk regulatoren en responstid på 0.121 sekund, et oversving på 13.8 % og en stabiliseringstid på omtrent 1.76 sekund, i simuleringer. Igjen presiseres det at dette er basert på simuleringer, og at en vil få andre resultater i virkeligheten, som forklart i kapittel 10.

11.4 Mikrokontroller

I samarbeid med Kraftoverførings- og -fordelingsgruppen [35] ble det tatt et valg om å ta i bruk utviklingskortet NUCLEO-H7A3ZI-Q med mikrokontrolleren STM32H7A3ZI [49]. Det ble brukt mye tid på å sette seg inn i hvordan en setter opp en mikrokontroller til ens behov, samt de forskjellige funksjonene og modulene en kan ta i bruk. Det ferdige systemet for styring og regulering genererer 12 ulike PWM- og PFM-signaler på totalt 6 timere, for styring av de 12 individuelle motorene. Kommunikasjon med sensorkort blir realisert ved bruk av SPI-kommunikasjon. For å oppnå tidspresise avbrudd for beregning og setting av signaler, samt henting av sensordata, benyttes avbruddsmetoden SysTick Interrupt Handler.

Det er verifisert at alle funksjonene implementert på mikrokontrolleren fungerer som de skal, som forespørsel og mottak av sensordata, generering av nødvendige PWM- og PFM-signaler, og beregning av motorpådrag ved bruk av PID-regulator, som forklart i kapittel 6 og 9.

11.5 Videre arbeid

Som resultat av handelskrig mellom USA og Kina ble det store forsinkelser i sending av kretskort, samt manglende komponenter på flere av disse. ROV-en ble derfor dessverre ikke ferdigstilt før bacheloroppgaven skulle leveres. Montering og funksjonstesting av ROV-en er planlagt til ukene etter leveringsfrist, og det er dermed mer enn nok tid til testing og tuning før en eventuell MATE konkurranse skal gjennomføres.

Funksjonstesting og etterjustering av parametrer gjennomføres som forklart i kapittel 10. Det er forventet at dette vil være nødvendig, da reguleringssystemet kun er basert på teoretiske utregninger og simuleringer som tar utgangspunkt i noen verdier som mest sannsynlig vil være annerledes i virkeligheten. Dette gjelder hovedsaklig dragkoeffisienten $C_d = 1$, ROV-ens masse $m = 35kg$ og avstand mellom massesenter og volumsenter, $\Delta s = 0.09m$.

Kapittel 12

Konklusjon

Arbeidet utført av gruppen gir etter gruppens egne mening et godt utgangspunkt for et solid og driftssikkert styrings- og reguleringssystem. Etter nødvendig testing og tuning av reguleringsparametrer, ser vi for oss, og håper at systemet vil gi tilfredsstillende resultater, og føre til enklere styring for piloten. Systemet skal i teorien sørge for at ROV-en holder ønsket dybde, rull- og stampvinkel, samt effektiv håndtering av ytre forstyrrelser i disse. I tillegg er det utviklet et intuitivt og brukervennlig styringssystem, som kjøres parallelt med reguleringssystemet.

I sin helhet har prosjektet vært svært lærerikt og interessant. I tillegg til at kunnskap fra så og si alle fag gjennomgått på Universitetet i Stavanger har blitt tatt i bruk, har vi fått mye mer kunnskap om blant annet:

- Oppbygging og virkemåte for forskjellige typer motorer og motorkontrollere
- Programmering, testing og utvikling i Simulink, C, Python, Javascript og HTML
- Anvending av mikrokontrolleres funksjoner og oppsett av disse i utviklingsmiljøet CubeIDE
- Avansert analyse av oppførsel for objekter under vann
- Hvordan det er å arbeide i et prosjekt der mange må samarbeide for et felles mål
- Ledelse (lagleder elektro)
- Bedriftsøkonomi (økonomiansvarlig)

Videre gjenstår en del arbeid sammen med resten av gruppene fra UiS Subsea, for å ferdigstille ROV-en til en eventuell MATE konkurranse. Dette vil føre til enda mer erfaring innen tverrfaglig arbeid, både i form av tett samarbeid med andre grupper, lodding og spleising av kabler, eventuell feilsøking og etterjustering av diverse programvare og parametrer.

Bibliografi

[1] MATE Center. *MATE ROV Competition manual - Explorer*, Besøkt: 09.01.21. URL http://files.materovcompetition.org/2021/2021_EXPLORER_Manual_14Sept2020.pdf.

[2] Daniel Vasshus Espen Myrset. *Utvikling av sensorsystem og elektronikkhus for undervannsfartøy.* Bacherloppgave UiS Subsea 2021, 2021.

[3] Bluerov2, Besøkt: 08.01.21. URL https://bluerobotics.com/store/rov/bluerov2/.

[4] Seabed dredger, Besøkt: 09.01.21. URL https://www.novasub.com/project-details/seabed-dredger/#1532340251241-e0d5a862-6c83.

[5] Serpent seaview, Besøkt: 09.01.21. URL https://www.seaviewsystems.com/toolbox/serpent-2/.

[6] Mate logo, Besøkt: 06.01.21. URL https://www.facebook.com/materovcompetition/photos/1250385541654400.

[7] Mate competition logo, Besøkt: 06.01.21. URL https://files.materovcompetition.org/images/logos/MATEROVCompetition5683x2662.png.

[8] Skjermdump av konkurranseoppgaver, Besøkt: 19.01.21. URL https://www.youtube.com/watch?v=KWNBOUqVIPQ&t=6s&ab_channel=MATECenter.

[9] Allan R. Hambley Narendra Kumar, Ashish R Kulkarni. *Electrical Engineering:Principles and Applications, International Edition: Principles and Applications. Sixth edition.* Pearson Education Limited, ISBN: 027379325X.

[10] Saugstad, kjell; gunvaldsen, ivar: elektrisk maskin i store norske leksikon på snl.no, Besøkt: Januar 2021. URL https://snl.no/elektrisk_maskin.

[11] How brushless motor and esc work, Besøkt: Februar 2021. URL https://futurelab3d.com/how-brushless-motor-and-esc-work/.

[12] Grøn, Øyvind: elektromagnetisk induksjon i store norske leksikon på snl.no, Besøkt: Januar 2021. URL https://snl.no/elektromagnetisk_induksjon.

[13] How a stepper motor works, Besøkt: Februar 2021. URL https://howtomechatronics.com/how-it-works/electrical-engineering/stepper-motor/.

[14] Bluerobotics, Besøkt: Januar 2021. URL https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-asm-rotor-r3-rp/.

[15] Bluerobotics, Besøkt: Januar 2021. URL https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-thruster/.

[16] Thrustme specifications, Besøkt: Januar 2021. URL https://www.thrustme.no/eu/thruster-more-eu.

[17] Joachim Merenyi Sindre Fjermedal. *Design and control of ROV manipulators*. Bacheloroppgave UiS Subsea 2021, 2021.

[18] Datablad for nema 23hm22-2804s, Besøkt: Februar 2021. URL https://www.omc-stepperonline.com/download/23HM22-2804S.pdf.

[19] Bilde av nema 23hm22-2804s, Besøkt: Februar 2021. URL https://www.omc-stepperonline.com/nema-23-bipolar-0-9deg-1-26nm-178-4oz-in-2-8a-2-5v-57x57x56mm-4-wires.html.

[20] Datablad for nema 17hs24-2104s, Besøkt: Februar 2021. URL https://www.omc-stepperonline.com/download/17HS24-2104S.pdf?fbclid=IwAR2bH8k0U3e9RL6I-XT9vXRPHbTvNYKnNy5eS6-dXLHHj5Nv6hLRBRz5UAY.

[21] Bilde av nema 17hs24-2104s, Besøkt: Februar2021. URL https://www.omc-stepperonline.com/nema-17-bipolar-1-8deg-65ncm-92oz-in-2-1a-3-36v-42x42x60mm-4-wires-it.html?search=nema%2017%20bipolar%201.8deg%2065ncm.

[22] Datablad for nema 14hs20-1504s, Besøkt: Februar 2021. URL https://www.omc-stepperonline.com/download/14HS20-1504S.pdf?fbclid=IwAR0tYq7REBS3TWFfW6XH8WyzWrxMCeoAmFvvyYTBu29-iEf5p2o6tDBg41I.

[23] Bilde av nema 14hs20-1504s, Besøkt: Februar 2021. URL https://www.omc-stepperonline.com/nema-14-bipolar-1-8deg-40ncm-56-7oz-in-1-5a-4-2v-35x35x52mm-4-wires.html?search=nema%2014%20bipolar.

[24] Hall effect sensor and its role in a motor controller, Besøkt: Februar 2021. URL https://www.embitel.com/blog/embedded-blog/hall-effect-sensor-and-its-role-in-a-motor-controller.

[25] Hall effect sensors – work, types, applications, advantages and disadvantages, Besøkt: Februar 2021. URL https://electricalfundablog.com/hall-effect-sensors-work-types-applications-advantages-disadvantages/#Disadvantages_of_Hall_Effect_Sensors.

[26] Sensorless bldc motor control and bemf sampling methods with st7mc, Besøkt: Februar 2021. URL https://www.digikey.no/no/articles/controlling-sensorless-bldc-motors-via-back-emf.

[27] Controlling sensorless bldc motors via back emf, Besøkt: Februar.2021. URL https://www.st.com/resource/en/application_note/cd00020086-sensorless-bldc-motor-control-and-bemf-sampling-methods-with-st7mc pdf.

[28] Sensorless trapezoidal control of bldc motors, Besøkt: Februar 2021. URL https://www.ti.com/lit/an/sprabz3/sprabz3.pdf?ts=1613041289330&ref_url=https%253A%252F%252Fwww.google.com%252F.

[29] Lenz's law, Besøkt: Februar 2021. URL https://www.britannica.com/science/Lenzs-law.

[30] Field-oriented control of small dc motors put drones on a rising flight path, Besøkt: Februar 2021. URL https://www.digikey.no/no/articles/field-oriented-control-of-small-dc-motors-put-drones-on-a-rising-flight-path.

[31] Increasing motor performance with field-oriented control, Besøkt: Februar.2021. URL https://www.digikey.no/no/articles/increasing-motor-performance-with-field-oriented-control.

[32] Sensorless field oriented control (foc) for permanent magnet synchronous motors (pmsm), Besøkt: Februar.2021. URL https://www.microchip.com/stellent/groups/SiteComm_sg/documents/Training_Tutorials/en532365.pdf.

[33] Microstepping myths, Besøkt: Februar 2021. URL https://www.machinedesign.com/archive/article/21812154/microstepping-myths.

[34] What is microstepping?, Besøkt: Februar 2021. URL https://www.linearmotiontips.com/microstepping-basics/.

[35] Andrine Pedersen Anniken Hjelm. *Kraftoverføring og -fordeling*. Bacheloroppgave UiS Subsea 2021, 2021.

[36] Alexander Voerman Sigvart Rodriguez. *Design for assembly concept and construction of ROV frame and performance analysis og thrusters*. Bacheloroppgave UiS Subsea 2021, 2021.

[37] Leadshine, Besøkt April 2021. URL http://leadshine.com/.

[38] Datablad til dm320t), Besøkt: Mars 2021. URL https://www.omc-stepperonline.com/download/DM320T.pdf.

[39] Bildet som viser dm320t), Besøkt: Mars 2021. URL https://www.omc-stepperonline.com/digital-stepper-driver/digital-stepper-driver-03-22a-18-30vdc-for-nema-8-11-14-16-17-stepper-motor-c
html?mfp=46-input-voltage-v%5BDC12%20-%2040%2CDC10%20-%2040%2CDC10%20-%2028%2CDC10%20-%2030%5D.

[40] Datablad til st-7128, Besøkt: Mars.2021. URL https://www.oyostepper.com/images/upload/File/THB7128_IC_Instructions.pdf.

[41] Bildet som viser st-7128, Besøkt: Mars.2021. URL https://www.oyostepper.com/goods-116-Bipolar-Stepper-Motor-Driver-Max-3A-Current-128-High-Subdivisio
html.

[42] Bilde av subconn circular series, Besøkt: Februar 2021. URL https://www.macartney.com/what-we-offer/systems-and-products/connectors/subconn/subconn-circular-series/.

[43] Bilde av subconn micro low profile series, Besøkt: Februar 2021. URL https://www.macartney.com/what-we-offer/systems-and-products/connectors/subconn/subconn-micro-low-profile-series/.

[44] Bilde av blue robotics m10 kabelpenetrator, Besøkt: Februar 2021. URL https://bluerobotics.com/store/cables-connectors/penetrators/penetrator-10-25-a-8mm-r2/.

[45] Bilde av pinner til macartney circular series 10 pin, Besøkt: Mars 2021. URL https://www.macartney.com/what-we-offer/systems-and-products/connectors/subconn/subconn-circular-series/subconn-circular-6-8-and-10-contacts/.

[46] Bilde av pinner til macartney micro low profile series 10 pin, Besøkt: Mars 2021. URL https://www.macartney.com/what-we-offer/systems-and-products/connectors/subconn/subconn-circular-series/subconn-circular-6-8-and-10-contacts/.

[47] Just Erik Ormbostad. *Normguiden, Veiledning til NEK:400:2010.* Elforlaget, ISBN: 978-82-7345-534-5.

[48] Hvordan beregne korreksjonsfaktor, Besøkt April 2021. URL http://elfagentusiastene.blogspot.com/2015/06/hvordan-beregne-korreksjons-faktor.html.

[49] Stm32 nucleo-144 development board with stm32h7a3zi mcu, smps, supports arduino, st zio and morpho connectivity, Besøkt Januar 2021. URL https://www.st.com/en/evaluation-tools/nucleo-h7a3zi-q.html.

[50] Thor I. Fossen. *Marine control systems: guidance, navigation and control of ships, rigs and under- water vehicles.* Marine Cybernetics, Besøkt: Mars 2021.

[51] Morten Tengesdal. *Kalman-filteret: Prosessinnsyn i praksis.* Morten Tengesdal, Besøkt: Mars 2021.

[52] Morten Tengesdal. *Litt om utrekining av motorpådrag for ein ROV.* Morten Tengesdal, Besøkt: Mars 2021.

[53] Kildekode til behandling av signaler fra xbox 360 kontroller., Besøkt: Januar 2021. URL https://github.com/joncoop/pygame-xbox360controller.

[54] Gamepad api, Besøkt: April 2021. URL https://developer.mozilla.org/en-US/docs/Web/API/Gamepad_API.

[55] How to connect a gamepad to the browser [html5 gamepad api], Besøkt: April 2021. URL https://www.youtube.com/watch?v=T8vi1JZyjhs.

[56] Analog stick input traces a square, Besøkt Februar 2021. URL https://answers.unrealengine.com/questions/226365/analog-stick-input-traces-a-square.html.

[57] STMicroelectronics. 32-bit arm® cortex®-m7 280 mhz mcus, up to 2-mbyte flash memory, 1.4 mbyte ram, 46 com. and analog interfaces, smps, Besøkt: Februar 2021. URL https://www.st.com/resource/en/datasheet/stm32h7a3zi.pdf.

[58] STMicroelectronics. Rm0455 reference manual, Besøkt: Februar 2021. URL https://www.st.com/resource/en/reference_manual/dm00463927-stm32h7a37b3-and-stm32h7b0-value-line-advanced-armbased-32bit-mcus pdf.

[59] Oliver Veland Martin Hausken. *Kommunikasjon og videostrøm av fjernstyrt undervannsfartøy.* Bacheloroppgave UiS Subsea 2021, 2021.

[60] What is buoyant force?, Besøkt: Mars 2021. URL https://www.khanacademy.org/science/physics/fluids/buoyant-force-and-archimedes-principle/a/buoyant-force-and-archimedes-principle-article.

[61] Nasa - the drag equation, Besøkt: Mars 2021. URL https://www.grc.nasa.gov/WWW/k-12/airplane/drageq.html.

[62] What is drag?, Besøkt Mars 2021. URL https://www.grc.nasa.gov/WWW/k-12/airplane/drag1.html.

[63] Drag coefficient, Besøkt Mars 2021. URL https://www.engineeringtoolbox.com/drag-coefficient-d_627.html.

[64] Finn Haugen. *Basic DYNAMICS and CONTROL*. TechTeach, ISBN: 978-82-91748-13-9.

[65] Lukas Neuenschwander Heiki Henrichsen. *Design av regulerings-, styrings- og kommuni- kasjonssystem for et fjernstyrt undervannsfartøy.* Bacheloroppgave UiS Subsea 2019, 2019.

[66] Wikipedia. List of moments of inertia, Besøkt: Mars 2021. URL https://en.wikipedia.org/wiki/List_of_moments_of_inertia.

[67] Khan Academy. Torque, Besøkt: April 2021. URL https://www.khanacademy.org/science/physics/torque-angular-momentum/torque-tutorial/a/torque.

[68] Wikipedia. Small-angle approximation, Besøkt: April 2021. URL https://en.wikipedia.org/wiki/Small-angle_approximation.

[69] Finn Haugen. *Dynamiske systemer: modellering, analyse og simulering.* 2. utg. tapir akademisk forlag, ISBN: 82-519-1877-4.

[70] Bjørnar Wiik Jens Trydal. *Bildegjenkjenning og autonom kjøring.* Bacheloroppgave UiS Subsea 2021, 2021.

Vedlegg

Hele programstrukturen med tilhørende kode for styrings- og reguleringssystem, samt alle Simulink- og Matlabfiler, er vedlagt som komprimerte filer. Disse kan lastes ned ved å klikke på tilhørende 'stift' under. For både kode og Simulink/Matlab er det en stift for filtype '.zip', og en stift for filtype '.7z'. Dette er fordi forskjellige PDF-lesere støtter forskjellige filformater. En kan lett teste hvilken av de som fungerer for PDF-leseren en er i, ved å dobbelklikke på stiften og se om det fungerer eller ikke. Om ingen virker, anbefales det å laste ned **Adobe Acrobat Reader DC**, da dette er den mest vanlige PDF-leseren, og det er bekreftet at den tillater vedlegg og nedlasting av filer med filtypen '.7z'.

Kode for styrings- og regulerintssystem på mikrokontroller

Klikk på stiften for å laste ned med filtype ".zip":
Klikk på stiften for å laste ned med filtype ".7z":

Simulink og Matlabfiler for simulering av matematisk modell og reguleringssystem

Klikk på stiften for å laste ned med filtype ".zip":
Klikk på stiften for å laste ned med filtype ".7z":

E-post fra Leadshine

Bildet under viser e-post dialog med produsent for motorkontrollere til stegmotorer. Vedlegget blir brukt i delkapittel 4.3.2.

Leadshine-Jason Peng <jason@leadshine.com>

Re: Fw: Digital vs Analog stepper drivers

To: edmondbv <edmondbv@hotmail.com>, Cc: Tonny <tonny@leadshine.com>, penny <penny@leadshine.com>

Hi Baloku,
Thanks for your interests in Leadshine products.
The analog driver is realized by the hardware circuit, and the digital driver is processed and controlled by the chip DSP with high speed of the signal process ability.
Please pay more attention to our EM-S series digital stepper drives, http://www.leadshine.com/series.aspx?type=products&category=stepper-products&producttype=stepper-drives&series=EM-S

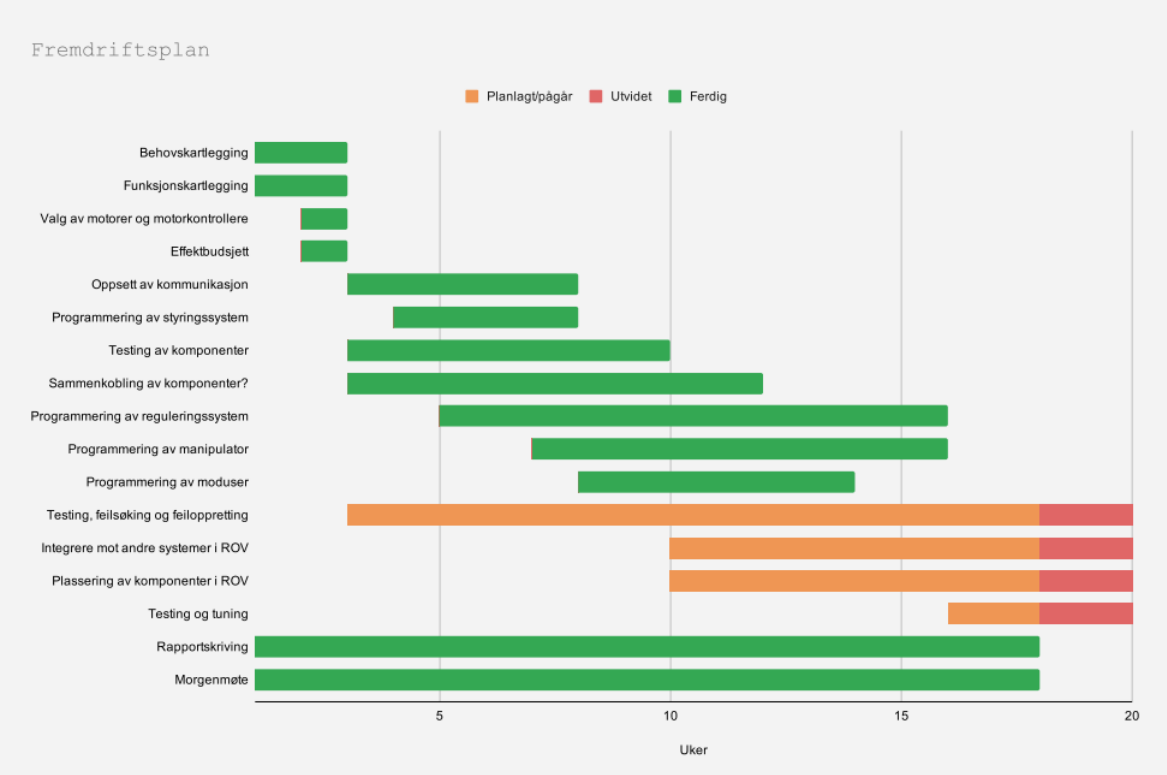
Digital drives are mainstream now on the market, here I find an artical from internet to share you something more, thanks.
https://www.quora.com/Whats-the-difference-between-digital-stepper-driver-and-analog-stepper-driver

Best regards.

Jason Peng
Senior Representative
China Leadshine Technology Co., Ltd.
Address: 11/F, Building A3, Nanshan iPark, No.1001 Xueyuan Blvd,
Nanshan District, Shenzhen, China.
TEL : +86 755 2665-5136
Skype: peng--jian@hotmail.com

Figur 13.1: Svar fra Leadshine angående forskjellen på digitale og analoge motorkontrollere.

Gantt-skjema av fremdriftsplan



Figur 13.2: Gantt-skjema av fremdriftsplan.