Universitetet i Stavanger

DET TEKNISK-NATURVITENSKAPELIGE FAKULTET

BACHELOROPPGAVE

Studieprogram/spesialisering: Bachelor i ingeniørfag / Automatisering og elektronikkdesign	Vårsemesteret 2022 Åpen eller Konfidensiell

Forfattere:

Kristian Birkeland — Otto Nessa Ljosdal — Tomas Royal Choat

Fagansvarlig: Morten Tengesdal

Veileder: Morten Tengesdal

Tittel på bacheloroppgaven: Styring og regulering av motorsystem for fjernstyrt undervannsfarkost

Engelsk tittel: Development of a control system for remotely operated underwater vehicle

Studiepoeng: 3*20

Emneord: Motor, motorkontroller, kretskort, styringssystem, reguleringssystem	Sidetall: 175 Vedlegg/annet: A: Programlisting, B: Testrapporter Stavanger 15. mai 2022

Sammendrag

Siden 2013 har studentorganisasjonen *UiS Subsea* ved Universitetet i Stavanger utviklet fjernstyrte undervannsfarkoster for å delta i den internasjonale konkurransen *MATE ROV Competition*. I år har prosjektet til sammen engasjert 21 studenter fra ingeniørfagretningene elektro, data og maskin ved Det teknisk-naturvitenskapelige fakultet, samt fra økonomi og administrasjon ved Handelshøgskolen UiS. Samtlige studenter skriver bacheloroppgave basert på prosjektet, hvor hver gruppe er tildelt sitt ansvarsområde for utvikling av farkosten.

For manøvrering av farkosten i vann, og kontroll av en påmontert manipulatorarm, kreves et velutviklet motorsystem. Dette systemet er ansvarsområdet gruppen vår er tildelt. Utviklingsprosessen innebærer utvalg av motorer og motorkontrollere, og å sette opp et styrings- og reguleringssystem for disse. Dette er derfor hovedinnholdet i denne bacheloroppgaven.

Motorutvalget er basert på teoretisk gjennomgang av virkemåte for ulike typer elektriske motorer. For manøvrering av undervannsfarkosten brukes ferdigutviklede thrustere fra *Blue Robotics*, som er designet for undervannsbruk. For å sikre tilstrekkelige bevegelsesmuligheter er det brukt åtte av disse. For styring av manipulatorarmen brukes børsteløse DC-motorer, som ble støpt i epoksy for vanntetting. Motorkontrollere fra *Blue Robotics* er tatt i bruk for alle motorer. Disse ble grundig analysert og kartlagt for å forstå virkemåten bak komponentene som driver motorene. Både motorer og motorkontrollere er testet, og fungerte godt etter sin hensikt.

For å realisere kommunikasjon med resten av elektronikken i undervannsfarkosten, er det designet et kretskort. Kortet består av kretser for CAN-kommunikasjon, LED-lys til testing og pådragssignal til motorer, samt tilkoblingspunkter for et plattformkort med mikrokontroller. Kretsene på kortet ble tegnet i *Altium*, og etter montering ble det verifisert at signalbaner var koblet som ønsket. Det er også utført konfigurering og programmering av mikrokontrolleren for å sørge for tilfredsstillende kommunikasjon og signalgenerering. Kortet er i sin helhet blitt testet sammen med resten av elektronikken, og oppfyller de kravene som på forhånd ble stilt.

For styringssystemet ble det gjort pådragsberegninger basert på styredata levert fra toppsiden, og opprettet tilhørende programkode for dette på mikrokontrolleren. Det er også gjennomført grundige kraftanalyser på undervannsfarkostens bevegelse i vann, og satt opp matematiske modeller i *MATLAB Simulink* basert på disse. Modellene la grunnlaget for videre utvikling av reguleringssystemet, som består av tre PID-regulatorer. Disse regulerer dybdeposisjon og stabilitet. Både styrings- og reguleringssystemet ble testet både på land og i vann, og gav tilfredsstillende oppførsel av ROV-en.

Under utvikling av et stort system som dette vil man alltid finne områder med forbedringspotensiale, og dette er gjort rede for i rapporten, men vi vil likevel si oss godt fornøyde med arbeidet som er lagt ned, og resultatet som er oppnådd.

Forord

Vi vil rette en stor takk til alle involvert i årets prosjekt for UiS Subsea. Prosjektet har vært preget av godt samarbeid, arbeidsvilje og ikke minst fellesskap. Å jobbe på et prosjekt av denne størrelsen har gitt mange gode erfaringer og et godt læringsutbytte. Selv om prosjektet har krevd mange timer med arbeid, har det samtidig vært engasjerende, og veldig givende.

En takk går til Daniel Vasshus og Geir Arne Solland Kindingstad, som har innhatt rollene som leder og nestleder i styret. Deres kjennskap til organisasjonen, samt deres visjon og innsats, har muliggjort det å videreutvikle organisasjonen UiS Subsea.

Universitetet i Stavanger har vært den største støttespilleren for UiS Subsea i alle år. Vi takker for denne støtten, både teknisk og økonomisk, som har lagt til rette for at et prosjekt av denne størrelsen er mulig. Tilgang til laboratorier, eget rom for organisasjonen og maskinverksted med behjelpelige ansatte, har vært uvurderlig for gjennomføringen.

Det rettes også en takk til alle bedrifter som har bidratt med sponsormidler, komponenter og rabatter. Uten disse ville det vært bortimot umulig å utvikle en ROV av denne kompleksiteten.

Sist men ikke minst, vil vi rette en ekstra stor og hjertelig takk til Morten Tengesdal, som har vært veileder for bachelorgruppen vår. Tengesdal har gjennom hele semesteret vist stort engasjement for prosjektet, og alltid vært villig og tilgjengelig for rådgivning og veiledning.

Ordliste

BaudRate: Informasjonshastighet ved signaloverføring.

BDC: Børstede DC-motorer.

BEMF: *Back Electro-Motive Force*. Indusert spenning som følge av en permanentmagnets påvirkning på en elektrisk leder.

BFF: *Body Fixed Frame*. Koordinatsystem festet til et legeme som beveger seg.

BLDC: Børsteløs DC-motor.

C: Programmeringsspråk.

CAD: *Computer Aided Design*. Design og konstruksjon ved hjelp av datamaskin.

Drain (transistor): Elektroden hvor ladningsbærere forlater en transistor.

ESC: *Electronic Speed Controller*. Elektronisk krets som styrer hastigheten til en elektrisk motor.

Float: Flyter. En autonom undervannsfarkost beregnet for innsamling av diverse data.

Gate (transistor): Elektroden som styrer ladningsstrømmen gjennom en transistor.

HAL: *Hardware Abstraction Layer*. Et programmeringsnivå som tillater å programmere maskinvare på et høyere nivå i C.

IC: *Integrated circuit*. Integrerte kretser.

IMC: *Internal Model Control*. Metode for bestemmelse av regulatorparametere.

IMU: *Inertial Measurement Unit*. Elektronisk instrument som måler et legemes akselerasjon og posisjon ved hjelp av akselerometere og gyroskop.

Inrunner: Elektrisk motor hvor roteren befinner seg innenfor statoren.

Jumper: Liten leder som brukes for å slutte en krets.

MATE: *Marine Advanced Technology Education*. Organisasjon som driver med undervannsteknologi, og arrangerer årlig en internasjonal ROV-konkurranse.

MOSFET: *Metal–Oxide–Semiconductor Field-Effect Transistor*. Metall oksid silisium-felteffekttransistor. En transistor ofte beregnet til bruk i effektkrevende kretser.

NED: *North East Down*. Koordinatsystem definert av jordens magnetiske retninger nord og øst, samt retning nedover mot jordens sentrum.

ODR: *Output Data Register*. Register som inneholder informasjon om hvilket spenningsnivå som skal settes på et sett pinner på en mikrokontroller.

Outrunner: Elektrisk motor hvor roteren befinner seg utenfor statoren.

Prescaler: Faktor som brukes til å nedskalere klokkefrekvensen.

QOL: *Quality of Life*. Funksjoner som bidrar til forenkling av visse prosesser.

Source (transistor): Elektroden hvor ladningsbærere komme inn på en transistor.

Struct: Samling av variabler av forskjellige datatyper i programmeringsspråket C.

Thruster: I denne oppgaven brukt om vanntette motorer med propell som skaper skyvekraft, på engelsk *thrust*.

Timer: Modul som teller basert på en gitt klokkefrekvens.

Toppsiden: Definert som kontrollstasjonen for ROV-en som befinner seg på land.

Innhold

Kapittel 1

Introduksjon

Kapitteloversikt

Denne bacheloroppgaven er en del av et større prosjekt, som baserer seg på design, utvikling og bygging av undervannsfarkoster. Utgangspunktet for prosjektet er deltakelse i den internasjonale konkurransen, *MATE ROV Competition* [23]. Kapittelet vil presentere organisasjonen UiS Subsea, hvilke typer undervannsfarkoster som skal utvikles og oppgavene de designes etter. Deretter vil det overordnede systemet presenteres med inndeling av ansvarsområder for de ulike bachelorgruppene. Avslutningsvis gis et mer detaljert innblikk i systemet denne bacheloroppgaven omhandler.

Introduksjonskapittelet er skrevet av gruppen med ansvar for styrings- og reguleringssystemet, altså gruppen vår, og er delt med alle gruppene i prosjektet. Kapittelet er felles for alle gruppene siden det presenterer prosjektet i sin helhet.

1.1 UiS Subsea

UiS Subsea er en studentorganisasjon ved Universitetet i Stavanger, som siden 2013 har engasjert studenter i undervannsteknologi. UiS Subsea har som mål å gi studentene erfaring med å jobbe i et større prosjekt på tvers av fagområder.

Organisasjonen har gjennom flere år bygget ROV-er for å delta i MATE-konkurransen. Dette gir grunnlag for et prosjekt som krever at de involverte løser utfordrende tverrfaglige oppgaver. Formålet med dette er å skape et miljø hvor studentene får utviklet sine tekniske ferdigheter, samt evne til å samarbeide med andre studenter. UiS Subsea er en organisasjon som muliggjør et tett samarbeid mellom studenter og næringslivet. En rekke bedrifter er involvert og svært interessert i dette prosjektet, og bidrar med utstyr og andre ressurser gjennom sponsoravtaler. For å dyrke dette samarbeidet, vil det i 2022, for første gang, arrangeres *Subsea-dagen*. Her vil næringslivet inviteres til universitetet for å presentere sine bedrifter.

I tidligere år har det manglet kontinuitet i organisasjonen, da den hvert år har blitt overtatt av et helt nytt kull med bachelorstudenter. I fjor ble organisasjonen restrukturert ved at styret ble adskilt fra selve prosjektet, med formål om å muliggjøre drift uavhengig av prosjektdeltakere. Dette åpner også mer opp for deltakelse fra studenter i andre perioder av utdanningsløpet. Fjorårets leder og nestleder har valgt å sitte en periode til, for å redusere belastningen på bachelorstudentene, og for å videreføre tidligere erfaringer.

Nytt for årets prosjekt, er at også en bachelorgruppe av økonomistudenter innehar en rolle i driften av organisasjonen. Dette bidrar med kunnskap innenfor økonomi og markedsføring i styret.

Årets styre består av følgende roller med tilhørende innehavere:

- **Styreleder:** Daniel Vasshus
- **Nestleder:** Geir Arne Solland Kindingstad
- **Økonomiansvarlig:** Sina Brunes
- **Markedsføring:** Sanna Sørskår og Åse Jortveit Sagebakken
- **Sponsoransvarlig:** Maren Lovise Jåsund og Otto Nessa Ljosdal
- **Styremedlem:** Tage Mellemstrand

Grunnet restruktureringen, er det opprettet en egen prosjektledelse med ansvar for det tekniske. Prosjektledelsen fungerer som et bindeledd mellom organisasjonen og prosjektdeltakerene.

Prosjektledelsen med roller og innehavere er:

- **Prosjektleder:** Tomas Royal Choat
- **Teknisk ansvarlig programvare:** Christoffer Næss
- **Teknisk ansvarlig maskinvare:** Mats Røste

1.2 Farkostene

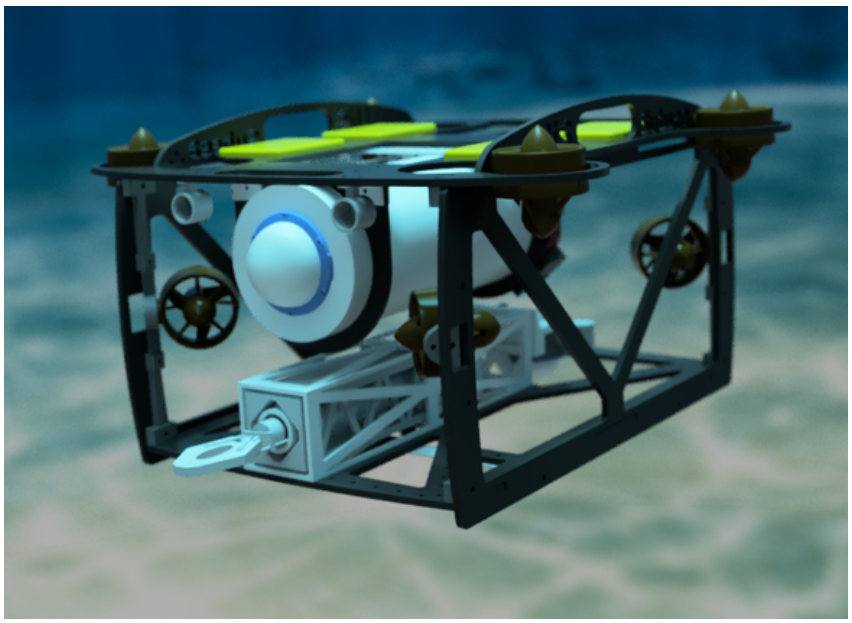
1.2.1 ROV

En ***R**emotely **O**perated Underwater **V**ehicle* (ROV) er, som navnet indikerer, en fjernstyrt undervannsfarkost. ROV-en er et hjelpemiddel som muliggjør undervannsoperasjoner uten behov for dykkere. Utviklingen av denne typen undervannsroboter har åpnet dørene for en rekke operasjoner under vann som ikke var mulig før, da man ikke lenger trenger å risikere menneskeliv. Dette utvider rekkevidden under vann, og forenkler operasjonene, siden de kan gjennomføres i mindre skala. Nyere ROV-er er også mulig å operere fra land, som reduserer kostnader av å utføre operasjoner *offshore*.

Utviklingen av ROV-er har på grunn av *offshore*-næringen vokst kraftig siden midten av 1980-tallet [61], da dette medførte utplassering av store installasjoner og behov for operasjoner på dybder man ikke kunne nå med dykkere. Eksempler på dette fremheves av årets konkurranse, med oppgaver som inspeksjon av fiskemerder og havvindmøller.

1.2.2 Fenris

ROV-en som utvikles i år, *Fenris*, er en frittsvømmende ROV med navlestreng, vist i figur 1.1.

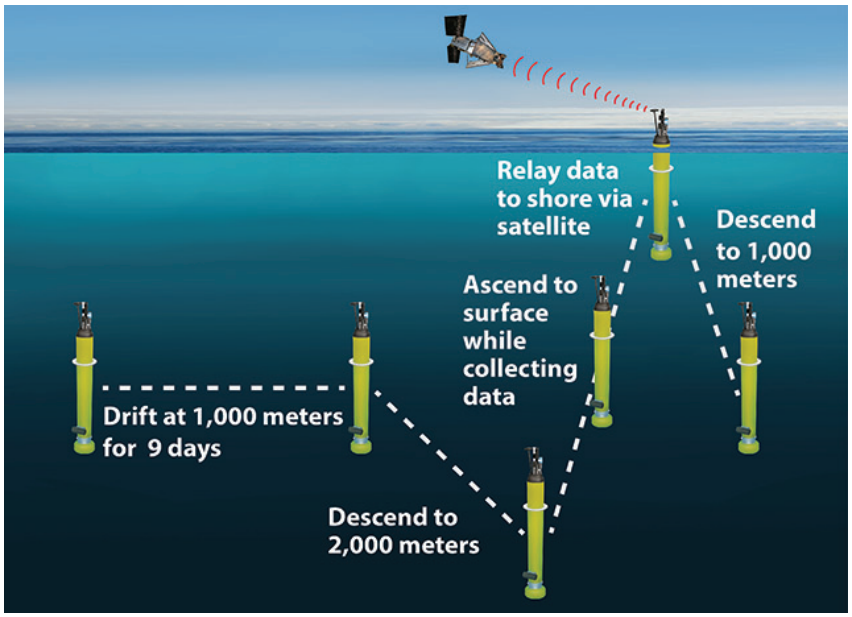


Figur 1.1: 3D-modell av Fenris.

Fenris styres fra et operatørgrensesnitt på land, og kommuniserer gjennom navlestrengen. Navlestrengen utgjør også kraftforsyningen til ROV-en. Målet er at ROV-en skal være i stand til å nå en dybde på 50 meter. Fenris er designet etter oppgavene i *MATE ROV Competition*, og er derfor utstyrt med en manipulatorarm for å løse disse. For å tilrettelegge for videreutvikling på ROV-en, er den designet modulært, slik at alt enkelt kan byttes ut. Dette gjør at Fenris kan brukes som en base for eventuell fremtidig utvikling.

1.2.3 Flyter

Nytt for årets prosjekt er at det skal bygges en *flyter*, inspirert av *float*-teknologi utviklet av forskningsprosjektet GO-BGC [15]. En flyter er en autonom undervannsfarkost som har i hovedoppgave å overvåke områder under vann. For å utføre dette, er den utstyrt med biologiske og kjemiske sensorer. Den har også en innebygd mekanisme som endrer volumet til farkosten, og dermed kan oppdriften reguleres. Et eksempel på syklusen til en flyter er vist i figur 1.2.

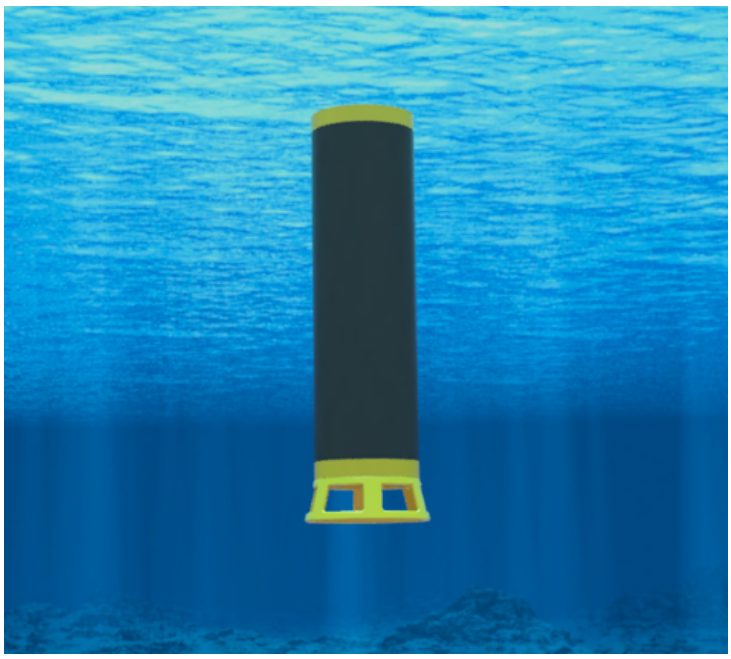


Figur 1.2: Eksempel på syklusen til en flyter. Hentet fra [28].

Flyteren kan dykke ned til flere tusen meters dybde, og returnere til overflaten på egenhånd. Når flyteren når overflaten, sendes innhentet data via satellitt, og slik skal den kunne fortsette ubemannet i opptil 5 år.

1.2.4 Frøya

Flyteren som utvikles i år, *Frøya*, er vist i figur 1.3.



Figur 1.3: 3D-modell av Frøya.

Dette er en prototype av en flyter, hvor fokuset er rettet mot den dynamiske oppdriftsmekanismen, og den er ikke utstyrt med biologiske eller kjemiske sensorer. Dybden den befinner seg på måles med en trykksensor, og reguleres gjennom en oppdriftsmotor. Oppdriftsmotoren er realisert ved hjelp av en pumpe som forflytter luft ut i en ballong på undersiden av flyteren. Av sikkerhetsmessige årsaker, må batteriene i flyteren være alkaliske. Dette begrenser hvor lenge den kan operere. Målet er at flyteren skal være i stand til å gjennomføre 2 vertikale profiler innenfor en tidsramme, etter at den er blitt plassert i vannet av ROV-en. I konkurransen tilsvarer en vertikal profil et dykk fra overflaten, ned til bunnen og tilbake.

1.3 MATE – Marine Advanced Tehchnology Education

Informasjonen under er hentet fra arrangørens hjemmeside [22].

Prosjektet gjennomføres, som tidligere nevnt, med hovedfokus rettet mot en internasjonal konkurranse innen utvikling av robotiserte undervannsfarkoster. Arrangøren bak konkurransen er forskningssenteret *The Marine Advanced Tehchnology Education (MATE) Center*, som er et samarbeid mellom flere organisasjoner i USA, med et felles mål om å forbedre utdanning innen marin teknikk. Senterets logo er vist i figur 1.4. Forskningssenteret jobber i tett samarbeid med flere skoler og universiteter, men også med bedrifter. Gjennom dette samarbeidet kan de tilby utdanning av studenter, basert på

Figur 1.4: The MATE Center sin logo. Hentet fra hjemmesiden deres.

etablerte retningslinjer fra bedriftene, og sørge for direkte
kontakt mellom student og arbeidsgiver under utdanningen. Et av hovedformålene med The MATE Center er å gi studenter et læringsutbytte, som senere kan være med på å utvikle hele næringen. I den sammenheng arrangeres konkurransen *MATE ROV Competition* hvert år. UiS Subsea stiller i *utforskerklassen*, som er klassen med høyest vanskelighetsgrad.

MATE ROV Competition

Informasjon om konkurransen er hentet fra konkurransemanualen [23]. Konkurransen har også sin egen logo, vist i figuren under.

Figur 1.5: *MATE ROV Competition* logo. Hentet fra konkurransens nettside.

Årets utgave av konkurransen vender oppmerksomheten mot FNs havforskningstiår (2021-2030), som er et initiativ fra FN med bakgrunn i de 17 bærekraftsmålene. Motivasjonen bak MATEs engasjement er å snu den nedadgående trenden til havets helse, og i den forbindelse vil de utfordre studenter globalt til å komme med nyskapende løsninger på problemet. Oppgaven går ut på å bygge en ROV som bidrar i arbeidet mot klimaendringer, legger til rette for ren energi, sørger for mat til en voksende populasjon, overvåker havets helse og bevarer maritim historie. MATE konstaterer at gjennom dette arbeid vil man skape «havet vi trenger, for fremtiden vi ønsker».

Poenggivning

Under konkurransen deles det ut poeng i tre hovedkategorier. I del 1 gjennomgår man en produktdemonstrasjon. Her kontrolleres vekt og fysiske mål. I denne delen skal også ROV-en og flyteren gjennom de praktiske oppgavene som tester deres operative egenskaper. I konkurransen er det tre separate oppgaver man skal løse, hvor hver oppgave har en tidsramme på 15 minutter. I tillegg belønnes struktur og effektivitet under demonstrasjonen. Man får 1 poeng ekstra for hvert minutt, og 0.01 poeng for hvert sekund, man fullfører før den avsatte tiden i hver oppgave. I del 2 får man poeng for teknisk dokumentasjon av farkostene, samt kommunikasjon og markedsføring fra organisasjonens (UiS Subsea) side. I del 3 får man poeng for å ta hensyn til sikkerhet. Dette innebærer skriftlig sikkerhetsdokumentasjon, men også det å vise aktsomhet under operasjoner med ROV-en.

Under gjennomgås de praktiske oppgavene, med tilhørende poengfordeling. Oppgaver, poengfordeling og restriksjoner er gjengitt fra konkurransemanualen [24].

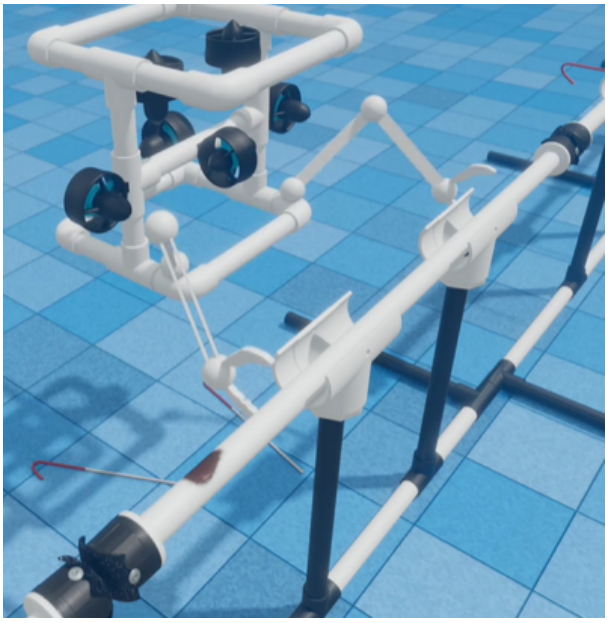
Oppgave 1: Maritim fornybar energi
FNs bærekraftsmål:
#7 Ren energi for alle
#12 Ansvarlig forbruk og produksjon

Oppgaven er utformet for å representere arbeidsoppgaver som er nødvendige for å utføre ved-

likehold på offshore havvind. Man skal blant annet reparere en skadet strømkabel, erstatte en ødelagt oppdriftsmodul og fjerne et fiskegarn som har satt seg fast på en av vindturbinenes understell. Utover dette skal det også plasseres ut en hydrofon for å overvåke tilstedeværelsen av sjøpattedyr. Til slutt skal ROV-en kjøre autonomt inn i en dokking-stasjon. Alle arbeidsoppgavene er simulert ved hjelp av konstruksjoner laget av PVC-rør. Under er oppgavebeskrivelsene gjengitt, med tilhørende poengfordeling.

1.1 Erstatte en skadet del av en strømkabel

Figur 1.6 viser hvordan deler av oppgaven kan løses.

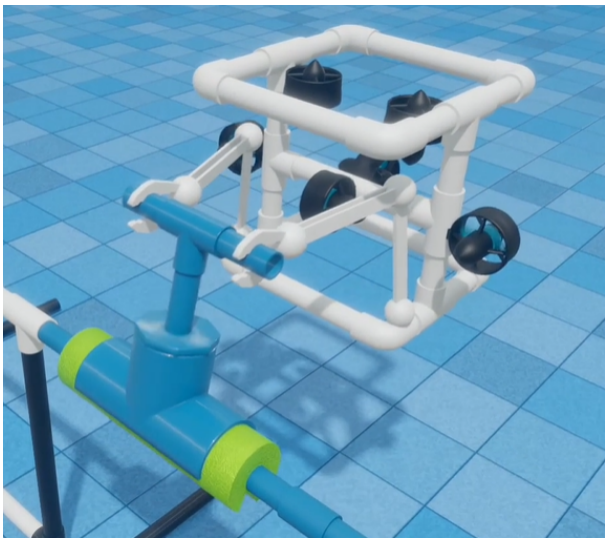


Figur 1.6: Fjerning av skadet del av kabel. Skaden er markert med en brun flekk. Låsepinnen vises rett til venstre for den brune flekken på røret. Hentet fra [26].

- Utføre en visuell inspeksjon av kabelen: **5 poeng**.
- Kutte kabelen på begge sider av den skadede delen. Simulert ved å dra ut en låsepinne på hver side: **10 poeng**.
- Fjerne den skadede delen av kabelen (PVC-rør), og bringe den til overflaten: **5 poeng**.
- Installere en ny kabelseksjon (PVC-rør), og feste denne med tilhørende festemekanisme: **5 poeng** for hver ende av røret, totalt **10 poeng**.

1.2 Erstatte en skadet oppdriftsmodul på en strømkabel tilhørende en havvindmølle

Figur 1.7 viser hvordan deler av oppgaven kan løses.

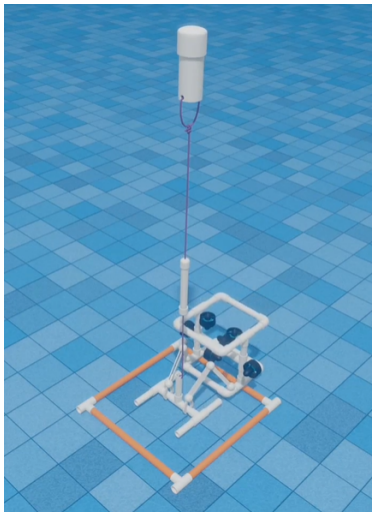


Figur 1.7: Fjerning av skadet oppdriftsmodul. Den nye oppdriftsmodulen er lik den som fjernes, men har borrelås som lar den festes på røret. Hentet fra [26].

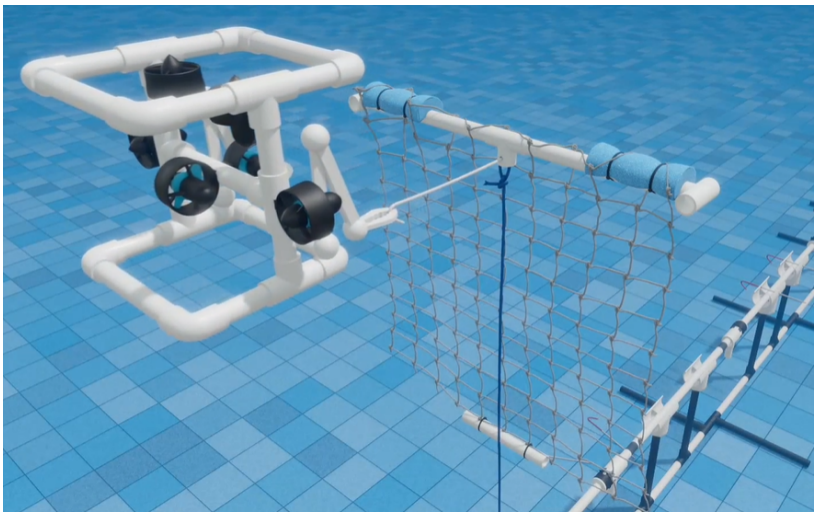
- Fjerne den skadede oppdriftsmodulen ved å rotere festet 180 grader: **5 poeng**.
- Frakte den skadede oppdriftsmodulen til kanten av bassenget, slik at den kan plukkes opp fra land: **5 poeng**.
- Plassere en ny oppdriftsmodul på kabelen: **5 poeng**.
- Feste oppdriftsmodulen ved hjelp av borrelås: **5 poeng**.

1.3 Overvåke miljøet

Figur 1.8a viser hvordan første del av oppgaven kan løses. Figur 1.8b viser hvordan andre del av oppgaven kan løses.



(a) Utplassering av hydrofon. Området på 40 cm × 40 cm er representert av de oransje PVC-rørene. Hentet fra [26].



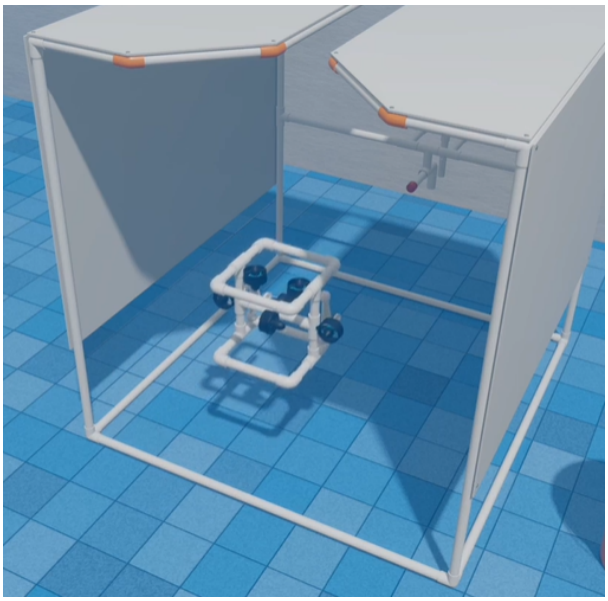
(b) Fjerning av fiskegarn. Garnet vil flyte opp til vannoverflaten når låsespinnen dras ut. Hentet fra [26].

Figur 1.8

- Utplassere en hydrofon, simulert av et PVC-rør, for å oppdage og registrere tilstedeværelsen av sjøpattedyr.
  - Plassere hydrofonen i et angitt område på 40 $cm \times 40$ cm: **5 poeng**.
  - Plukke opp hydrofonen etter 5 minutter i vannet og frakte den til bassengkanten: **5 poeng**.
- Fjerne et fiskegarn som er fanget på vindturbinens understell.
  - Dra ut en låsepinne som holder en flytende ramme med fiskegarn under vann: **10 poeng**.
  - Fjerne fiskegarnet fra vannet ved å frakte det til bassengkanten, og plukke det opp fra land: **10 poeng**.

1.4 Styre ROV-en inn i en dokking-stasjon

Figur 1.9 viser hvordan oppgaven kan løses.



Figur 1.9: Parkering av ROV i dokking-stasjon. Hentet fra [26].

- Styre ROV-en autonomt inn i dokking-stasjonen: **15 poeng**.
- Styre ROV-en manuelt inn i dokking-stasjonen: **5 poeng**.

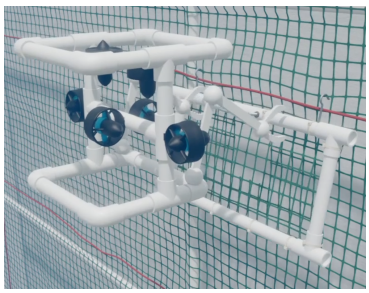
Oppgave 2: Offshore akvakultur og blå karbon
FNs bærekraftsmål:
#2 Utrydde sult
#13 Stoppe klimaendringene
#14 Liv under vann

Oppgaven er utformet for å representere arbeidsoppgaver som er nødvendige for å utføre vedlikehold på en fiskemerd, og å bidra til en sunn havhelse. Man skal blant annet reparere en skadet del av en fiskemerd, og fjerne uønsket algevekst. Oppgaven innebærer også å autonomt skille døde

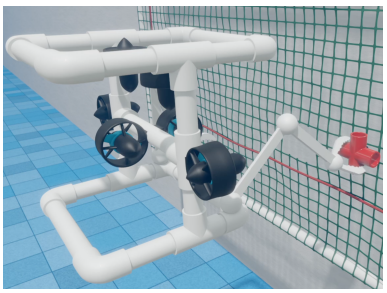
fisk fra levende, og å kunne måle størrelsen på fisken. Utover dette skal dødt sjøgress fjernes, og det skal videre plantes nytt. Alle arbeidsoppgavene er simulert ved hjelp av konstruksjoner laget av PVC-rør. Under er oppgavebeskrivelsene gjengitt, med tilhørende poengfordeling.

2.1 Inspisere en offshore akvakultur fiskemerd

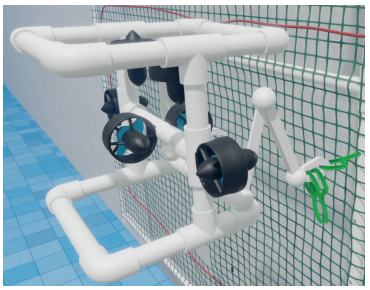
Figur 1.10 viser hvordan noen av deloppgavene kan løses.



(a) Reparasjon av skadet del av nettet.



(b) Fjerne innkapslende marin vekst.



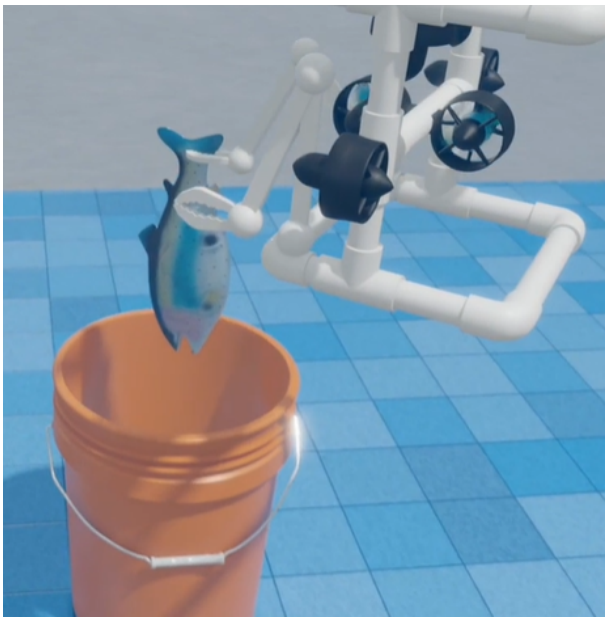
(c) Fjerne algevekst.

Figur 1.10: Alle hentet fra [26].

- Inspisere nettet for å identifisere skadede områder.
  - Kjøre en transektlinje for å identifisere skadede områder: Autonomt: **25 poeng**. Manuelt: **10 poeng**.
  - Identifisere og telle skadede områder av nettet: **5 poeng**.
- Reparere en skadet del av nettet: **10 poeng**.
- Fjerne marin vekst.
  - Fjerne innkapslende marin vekst, simulert med et PVC-kryss: **5 poeng**.
  - Fjerne algevekst, simulert med 3 piperensere: **5 poeng**.

2.2 Opprettholde et sunt miljø

Oppgaven som innebærer oppsamling av død fisk er vist i figur 1.11.



Figur 1.11: Plassering av død fisk i oppsamlingsrør. Hentet fra [26].

- Håndtere fiskedød ved å fjerne død fisk, simulert av en gummifisk, fra bunnen av bassenget.
  - Lage et program som kan skille døde fra levende fisk. Man får utdelt en video, og programmet skal merke døde fisk med røde rammer: **10 poeng**.
  - Plukke opp en død fisk: **5 poeng**.
  - Legge fisken i et oppsamlingsrør, simulert av en bøtte: **5 poeng**.

2.3 Måle fiskestørrelser

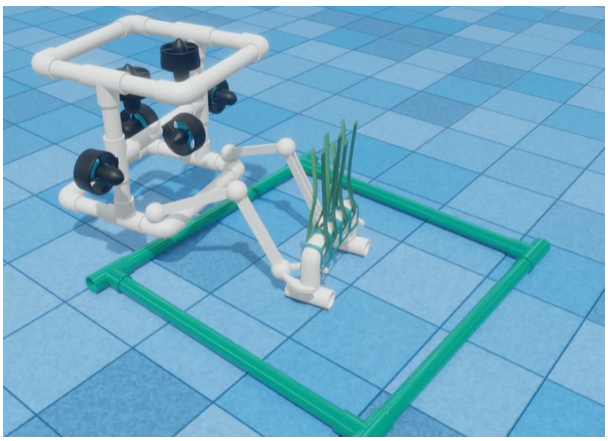
Et eksempel på hvordan man kan løse denne oppgaven er vist i figur 1.12.

Figur 1.12: Måling av fiskelengde. Hentet fra [26].

- Beregne gjennomsnittsstørrelsen på en fiskekohort (tre gummifisker av ulik lengde) med maksimalt 2 cm feilmargin: **15 poeng**.
- Beregne biomassen av fiskekohorten: **5 poeng**.

2.4 Dyrke sjøgress

Planting av nytt sjøgress er vist i figur 1.13.



Figur 1.13: Planting av nytt sjøgress. Sjøgresset som skal lukes er simulert av en lignende konstruksjon, men denne skal fjernes fra det markerte området. Hentet fra [26].

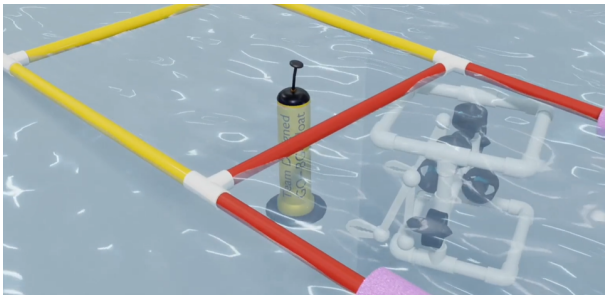
- Luke et eksisterende sjøgressbed: **5 poeng**.
- Plante et nytt sjøgressbed: **5 poeng**.

Oppgave 3: Da og nå – *Endurance22* og *MATE Floats*!
FNs bærekraftsmål:
#13 Stoppe klimaendringene

Første del av oppgaven representerer det å hente inn en «GO-BGC»-flyter, for så å plassere ut vår egenproduserte flyter i et angitt område. Flyteren vår skal deretter gjennomføre 2 vertikale profiler, altså traversere dybden av bassenget to ganger. Andre del av oppgaven handler om å kartlegge posisjonen til vraket av skipet *Endurance*, som sank i Antarktis. Deretter skal det autonomt lages en fotomosaikk av vraket, og videre skal lengden av det måles. Både «GO-BGC»-flyteren og vraket av Endurance er simulert ved hjelp av PVC-rør. Under er oppgavebeskrivelsene gjengitt, med tilhørende poengfordeling.

3.1 MATE Floats!

Utplassering av flyter i et angitt område er vist i figur 1.14.



Figur 1.14: Plassering av selvlagd flyter i angitt område. Hentet fra [26].

- Hente inn en *GO-BGC* flyter.
  - Beregne hvor flyteren dukker opp fra vannet: **5 poeng**.
  - Plukke opp flyteren, simulert av et PVC-rør: **10 poeng**.
- Design og konstruksjon av en operativ flyter for vertikal profilering.
  - Bygge en flyter før konkurransen starter: **5 poeng**.
  - Plassere flyteren i et angitt område ved hjelp av ROV-en: **5 poeng**.
  - Flyteren fullfører vertikal profilering.
    - To ganger: **25 poeng**.
    - Én gang: **15 poeng**.

3.2 Endurance22

Kartlegging av området der vraket av *Endurance* ble funnet, se figur 1.15.

Figur 1.15: Området der vraket av Endurance er plassert. Vraket er simulert med den brune PVC-strukturen. Hentet fra [25].

- Finne og kartlegge posisjonen til Endurance.
  - Kjøre en transektlinje over området der vraket er. Vraket er simulert av PVC-rør: **10 poeng**.
  - Indikere på et kart hvor vraket er: **5 poeng**.
- Lage en fotomosaikk av vraket.
  - Ta bilder av alle delene av vraket: **5 poeng**.
  - Autonomt sette bildene sammen til en fotomosaikk: **20 poeng**.
  - Manuelt sette bildene sammen til en fotomosaikk: **10 poeng**.
- Måle lengden av vraket fra baug til akterende.
  - Innen 10 cm av den sanne lengden: **10 poeng**.
  - Innen 10.1 til 20 cm av den sanne lengden: **5 poeng**.

Videre vil ROV-en veies, og belønnes med 10 ekstrapoeng dersom egenvekt i luft er mindre enn 20 kg, og 5 ekstrapoeng med egenvekt under 28 kg. Det deles ikke ut poeng dersom ROV-en er

tyngre enn 28 kg, og veies den til over 35 kg resulterer det i diskvalifikasjon. Nytt for året er at det ikke tas størrelsesmål av farkosten. Likevel begrenses størrelsen på ROV-en av dokking-stasjonen på én kubikkmeter.

Oversikt over poengfordeling

Det gås ikke inn på detaljer rundt poenggivning i forhold til dokumentasjon og sikkerhet, da det ikke har betydning for den tekniske delen av prosjektet. Tabellen under viser en oppsummering av den totale poengfordelingen i konkurransen.

Produktdemonstrasjoner	
Oppgave 1-3	300 + tidsbonus
Vektrestriksjoner	10
Organisatorisk effektivitet	10
Prosjektering og kommunikasjon	
Teknisk dokumentasjon	100
Produktpresentasjon	100
Markedsføring	50
Bedriftens spesifikasjonsark	20
Bedriftsansvar	20
Bruk av VR-utstyr	25
Sikkerhet	
Gjennomgang av sikkerhet og dokumentasjon	20
Sikkerhetsinspeksjon	30
Sikker jobbanalyse	10
Totalt	695 + tidsbonus

Begrensninger og restriksjoner

Fysiske begrensninger for ROV-en omfatter ikke flyteren, da denne regnes som en egen enhet.

I utforskerklassen spesifiseres det at deltakere skal designe en ROV, uten eksterne komponenter, for å løse de praktiske oppgavene. Alle deler og komponenter skal være festet i ROV-ens ramme. Thrusterene skal ha beskyttelse som ikke tillater fingre å berøre propellene. Farkosten skal operere i kloret ferskvann med temperaturer i området 15 ^{o}C - 30 ^{o}C. Det nevnes også at uforutsette strømninger i vannet kan oppstå på grunn av bassengets filtreringssystem. Bassenget oppgis til å være maksimalt 6 meter dypt, og alle oppgaver utføres innenfor 10 meter fra bassengkanten. Stasjonen for skjermer og styresystem vil maksimalt være plassert 3 meter fra bassenget. Lengde på navlestreng må altså beregnes etter disse målene. Vinsj eller kran for sjøsetting er ikke tillatt. Dette utføres manuelt.

Arrangøren stiller med kraftforsyning, og dette er den eneste forsyningen man har tilgang til. Ekstra batterier og lignende tillates ikke. Kraftforsyningen leverer en nominell spenning på 48 VDC. I år er også deltakere pålagt å bruke en av MATE sine elektriske sikringer på 20 A, 25 A eller 30 A. Dette begrenser altså effektforbruket til 1440 W. All spenningsregulering skal foregå i elektronikkhuset til ROV-en, og kan ikke overgå 48 VDC. I tillegg er det oppgitt at koblinger og motorer som er eksponert i vann skal vanntettes, og ha en minimum resistans på 10 MΩ til vannet.

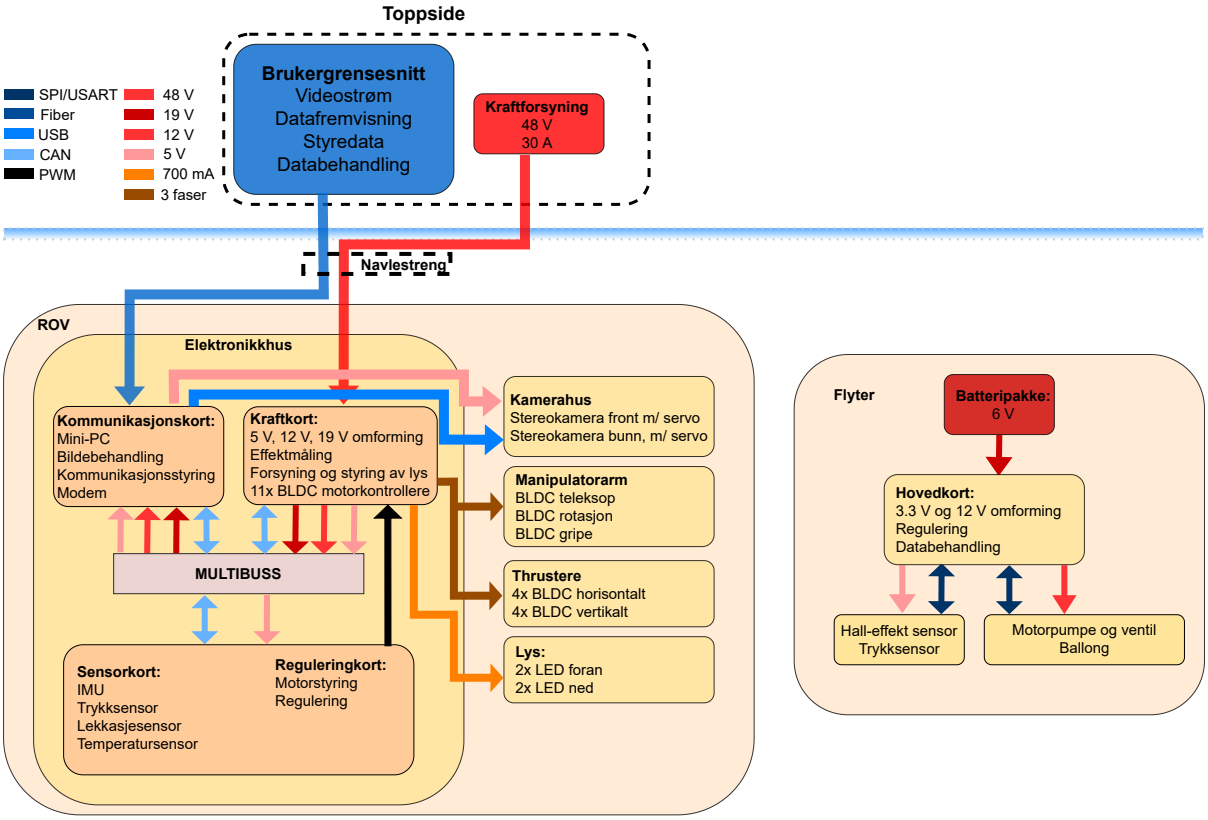
Flyteren skal være bygget fullstendig adskilt fra ROV-en, og skal fungere som en egen enhet.

Størrelsen på flyteren er begrenset til 1 m i høyde, og 18 cm i diameter. Batterier ombord brukes som kraftforsyning. Her angir arrangøren en maksimal spenning på 12 VDC, og maksimalt strømtrekk på 6 A. Det kreves i tillegg en sikring på 7.5 A, som maksimalt kan plasseres 5 cm fra positiv pol på batteripakken.

1.4 Overordnet system

Figur 1.16 viser et blokkskjema av det overordnede systemet. ROV-en styres fra et brukergrensesnitt på toppsiden. Herfra sendes det styredata, i tillegg til at det vises videostrøm og status fra ROV-en. Kraftforsyning og fiberkommunikasjon fra toppsiden går gjennom en navlestreng. Kraftforsyningen går direkte inn på kraftkortet. Her foregår omforming til nødvendige spenningsnivåer, og distribuering av disse. Lys, motorer og motorkontrollere forsynes direkte fra kraftkortet. Fiberkommunikasjonen går inn på en mini-PC, som behandler data og organiserer kommunikasjonen i systemet. Kommunikasjonssignalene går inn på et kommunikasjonskort, via USB, og sendes videre over den lokale CAN-bussen. Mini-PC-en er også koblet direkte til kameraene, og driver kontinuerlig bildebehandling. Både kommunikasjon og kraftforsyning sendes ut til det interne systemet i elektronikkhuset, via en felles multibuss. Det siste kortet er et kombinert sensor- og reguleringskort. Sensordelen av kortet leser og behandler data fra de ulike sensorene i systemet. Reguleringsdelen sørger for styring av thrustere og manipulatormotorer basert på styredata fra toppsiden. I tillegg utføres regulering av thrustere fra sensordataen. Pådragsdata sendes videre via motorkontrollerne på kraftkortet, og ut til aktuelle motorer.

Flyteren drives av en batteripakke på 6 V, som går direkte inn på et hovedkort. Her omformes spenningen fra 6 V til både 3.3 og 12 V. I tillegg gjennomføres databehandling og regulering i mikrokontrolleren på kortet. Sensordataen hentes fra to eksterne sensorkort, som inkluderer én hall-effekt-sensor, og én trykksensor. Mikrokontrolleren sender også styredata til motorpumpen og luftventilen, som sammen med en ballong utgjør oppdriftsmotoren.



Figur 1.16: Overodnet blokkskjema av ROV-en.

1.5 Prosjektinndeling

Prosjektet er delt opp i flere oppgaver, hvor hver bachelorgruppe er tildelt et eget ansvarsområde. Totalt utgjør dette 9 bacheloroppgaver, hvor 5 av disse er innen elektro, 2 innen maskin, 1 innen data og 1 innen økonomi. Nedenfor presenteres hver oppgave i korte trekk, med ansvarsområde og gruppemedlemmer.

1.5.1 Elektro

Kraftforsyningsmodul – Carl Henrik Preber Ettesvoll, Nicolai Jensen Narvesen og Jon Arve Andersen:

Oppgaven handler om å utvikle en kraftforsyningsmodul til ROV-en. I konkurransen skal ROV-en få elektrisk forsyning fra land via en navlestreng. Konkurransemanualen oppgir at det blir tilført 48 V spenning, og at strømbegrensningen er på 30 A. Kortet skal sørge for at all elektronikk blir tilført riktig spenningsnivå, og at effektforbruk overvåkes der det er nødvendig. Det skal også kunne prioritere kritiske prosesser, ved hjelp av digitalt styrte sikringskretser. Kraftkortet må ha rask responstid, kombinert med et robust design.

Maskinsyn og kommunikasjon – Christoffer Næss, Mats Røste og Tage Mellemstrand:

Oppgaven er todelt. Maskinsyn-delen går ut på å hente ut informasjon fra stereokameraene. Dette

innebærer blant annet avstands- og størrelsesmåling, gjenkjenning av objekter og nødvendig informasjon for å løse de autonome oppgavene. Kommunikasjonsdelen går ut på å lage et robust kommunikasjonssystem i samarbeid med brukergrensesnittgruppen. Systemet skal kunne motta kommandoer fra toppsiden, behandle disse og sende dem videre til mikrokontrollere, som styrer de lokale prosessene i ROV-en. I tillegg skal systemet hente bilder fra kameraer, og sender dem tilbake til toppsiden, med så lite forsinkelse som mulig.

Sensorkort – Jørgen Hemnes Johannessen:

Oppgaven handler om å utvikle et sensorsystem til ROV-en. Dette innebærer å hente inn rådata fra sensorer, behandle dem og distribuere dem videre over kommunikasjonssystemet. Sensordata som behøves til reguleringen, er vinkelhastighet og -posisjon, samt dybden ROV-en befinnner seg på. Det må i tillegg være mulig å detektere kritiske feil, som for eksempel vannlekkasje eller overoppheting. For å realisere systemet, må det utvikles et kretskort med en mikrokontroller som sørger for tilfredsstillende databehandling. Vinkeldata leses fra en *Inertial Measurement Unit* (IMU) som settes på kortet. Resterende data leses fra eksterne sensorer. Dette innebærer trykk- og temperaturmåling, i tillegg til lekkasjedeteksjon.

Styrings- og reguleringssystem – Tomas Royal Choat, Kristian Birkeland og Otto Nessa Ljosdal:

Oppgaven går ut på å utvikle et motorsystem for ROV-en. Dette består av thrustere for manøvrering, samt elektriske motorer for bevegelse av manipulatorarmen. Det skal utvikles et robust styringssystem for alle motorer. Dette må kunne motta manuell styredata fra toppsiden, eller autonom styredata fra mini-PC-en, og generere motorpådrag etter ønsket oppførsel. For å forenkle manøvreringen, skal det i tillegg utvikles et reguleringssystem, bestående av tre PID-regulatorer. Disse skal sørge for at ROV-en automatisk kan holdes ved en ønsket dybde, og at den ligger stabilt i vannet. Sistnevnte oppnås ved å motvirke rotasjon fremover, bakover og sideveis.

Utvikling av smart flyter – Malin Harr Overland og Hanne Lovise Berger:

Oppgaven går ut på å utvikle en trådløs flyter, som automatisk kan manøvreres i vertikal retning. Manøvreringen realiseres ved hjelp av et styresystem, og en oppdriftsmotor. Motoren består av en pumpe og en solenoidventil, som endrer oppdriften ved å forflytte luft fra innsiden av flyteren, og ut i en ballong. For at styresystemet skal vite når motoren må aktiveres, kreves et sensorsystem. Dette inkluderer en trykkmåler for avlesing av dybdeendring, og en hall-effektsensor for å detektere om flyteren holdes av ROV-en. Flyteren skal i tillegg drives på batterier, og systemet må derfor tilpasse spenningsnivå til de ulike formålene. Dette gjøres på et hovedkort, som også sørger for tilfredsstillende styring og databehandling.

1.5.2 Maskin

Design og montering av ROV og flyter - Christine Nordal og Sandra Nygård:

Oppgaven går ut på å designe og bygge rammen til ROV-en, samt den indre og ytre konstruksjonen til flyteren, ved å følge produktutviklingsprosessen. Den indre konstruksjonen til flyteren inkluderer oppdriftssystemet. Fokuset ved designet er å sikre at farkostene er godt egnet til å utføre oppgavene i konkurransen. I år rettes det også et økt fokus mot bærekraft og miljø. Det vil

derfor bli lagt vekt på konseptet *Design for environment* (DFE) i utviklingsprosessen. Målet er å redusere miljøpåvirkningen i utviklingen av produktene. Effektiv bruk av DfE kan også bidra til reduserte kostnader og produksjonstid, i tillegg til å øke kvaliteten på produktet. Oppgaven inneholder også materialvalg, dimensjonering, strukturell analyse, flytanalyse og beregning av oppdrift og stabilitet. Dette for å sikre at produktene er stabile, har riktig oppdrift og er lette å manøvrere.

Design og produksjon av manipulator – Henrik Welde og Sindre Rød Torsteinsen:

Oppgaven går ut på å designe en manipulator som er i stand til å gripe tak i objekter. Fokuset for designet er å gjøre den så enkel som mulig, uten at dette går på bekostning av funksjonaliteten. Dette vil forenkle demontering, i tilfelle vedlikehold må utføres. Samtidig skal det tilstrebes å produsere armen så lett som mulig, da redusert vekt vil bidra til både bedre manøvreringsegenskaper, og poeng i konkurransen. Klypen må være i stand til å gripe objekter av forskjellige størrelser og former. I tillegg skal den ha mulighet for rotasjon og teleskopbevegelse.

1.5.3 Data

Operatørgrensesnitt og kommunikasjon – Vebjørn Lia Riiser og Åse Jortveit Sagebakken:

Oppgaven som skal løses, er å utvikle et system for å styre og overvåke ROV-en. Det må derfor implementeres et system for å sende kommandoer og styringsdata fra toppsiden til mini-PC-en i ROV-en, som gjøres i samarbeid med maskinsyn-gruppen. I tillegg må det designes et brukergrensesnitt som presenterer videostrøm i sanntid, og sensordata. Hovedfokuset for grensesnittet er brukervennlighet, som innebærer fremvisning av nødvendig informasjon på en oversiktlig måte.

1.5.4 Økonomi

Endringsprosess i UiS Subsea – Maren Lovise Jåsund, Sina Brunes og Sanna Sørskår:

UiS Subsea er i år utvidet ved å involvere en gruppe fra økonomi og administrasjon. Målene fremover er blant annet å utvide organisasjonen til å involvere flere fagfelt, og ha flere pågående prosjekter samtidig. Ved hjelp av John P. Kotters 8-stegs modell, skal gruppen se på hvordan UiS Subsea på best mulig måte kan fortsette endringsprosessen av organisasjonen. Hovedsakelig innebærer dette organisering av de økonomiske og administrative oppgavene. UiS Subsea er hovedkilden. For å få et større perspektiv på økonomi og administrasjon, intervjues i tillegg tre andre bedrifter som organiserer disse oppgavene på forskjellige måter. Innhentet informasjon legger grunnlaget for videre analyser av hvordan UiS Subsea på best mulig måte kan organiseres med flere medlemmer og prosjekter. De strategiske analysene som skal gjennomføres, er *SWOT*-analyse og strategilerret. Analysemetodene vil gi god innsikt i bedriftene, og et godt grunnlag for å sammenligne dem.

1.6 Regulering og styring av motorsystem

Bacheloroppgaven er å utvikle et system for regulering og styring av ROV-en Fenris. Den sentrale oppgaven er å lage et kretskort som fungerer som et grensesnitt mellom motorkontrollerne og de andre kretskortene i elektronikkhuset, se figur 1.16. Som figuren også viser, deles PCB-kortet med sensorkretsen, og motorkontrollerne er montert på kraftforsyningskortet. I tillegg til grensesnittkortet, faller det også under vår oppgave å vurdere og velge ut passende thrustere, manpulatormotorer og motorkontrollere for disse.

Thrusterne og deres konfigurasjon, samt manipulatormotorene, er avgjørende å velge ut først for å danne et grunnlag for hvordan alt skal styres. Valget må baseres på begrensninger fra oppdragsgiveren, som i vårt tilfelle vil være *MATE ROV Competition* sine retningslinjer. Det er også avgjørende å ta beslutningene sammen med de prosjektgruppene som er direkte påvirket av valgene. Thrustervalg må utføres i samarbeid med prosjektgruppen med ansvaret for ROV-design, manipulatormotorene velges i samarbeid med manipulatorgruppen, og gruppen med ansvar for kraftforsyningen er med i avgjørelsen for alle motorene og tilhørende kontrollere.

Styringssystemet skal behandle instruksjoner fra toppsiden, eller fra en autonom pilot basert på maskinsyn. Reguleringen skal tilby aktiv stabilisering, samt en konstant dybdeposisjon. Den skal utføres med sensormålinger for de aktuelle frihetsgradene fra sensorkretsen. Styring, regulering og kommunikasjon med de andre enhetene i ROV-en skal utføres digitalt ved hjelp av en mikrokontroller. Kommunikasjon med de andre kortene inni ROV-en skal gå over CAN-buss på multibussen, bortsett fra styringssignalene til motorene, som sendes direkte fra GPIO-pinner på mikrokontrolleren til motorkontrollerne.

For å realisere regulering av de aktuelle frihetsgradene er det nødvendig med gode matematiske modeller. Disse modellene danner grunnlaget for å utvikle gode regulatorparametere og det er derfor nødvendig at disse er gode representasjoner for det faktiske systemet. Det er et mål at eventuelle forenklinger i modellene er virkelighetsnære og ikke bidrar til mer usikkerhet i modellene enn nødvendig. Basert på modellene kan systemet simuleres og regulatorparameterene testes før ROV-en er ferdig konstruert.

Målet er å utvikle et robust, effektivt og velfungerende system i bunn, for deretter å videreutvikle programvaren til å inneholde smarte løsninger og tilby *QOL*-forbedringer.

Kapittel 2

Motorer

Kapitteloversikt

En ROV er avhengig av gode manøvreringsegenskaper i vann for å løse de spesifikke oppgavene den er bygget for. Ved økt vanskelighetsgrad på oppgaver kreves også en optimalisert manipulator. Riktig valg av motorer står sentralt for å få til kontrollerte og ønskede bevegelser av farkosten. Dette kapitlet omhandler teorien bak virkemåten til ulike motorer, og bakgrunnen for de valgene vi har gjort underveis. Både motortype og motoroppstilling vil ha stor betydning for at vi skal kunne sette opp et vellykket styre- og reguleringssystem. Konkurransemanualen fra *MATE* [24] må også vurderes nøye under motorvalg, da denne angir vilkårene for at ROV-en får lov til å delta i konkurransen.

2.1 Motorer – virkemåte

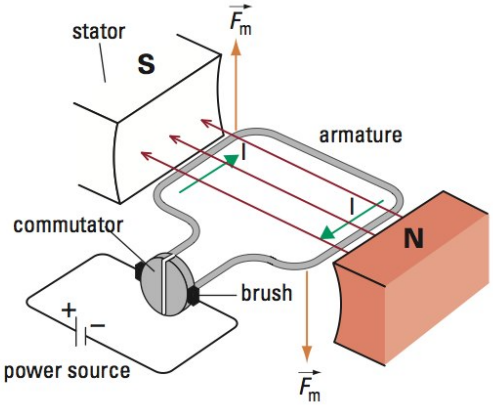
Vi har valgt å prioritere et helelektrisk system, og avgrenser derfor motorene til å ha elektrisk drivkraft. Man kunne alternativt vurdert hydrauliske motorer for å løse visse utfordringer. Manipulatoren skal kunne klype fast i objekter, og bruk av hydraulikk kunne skapt mer kraft i klypen enn rent elektriske systemer er kapable til [3]. Likevel er vi underlagt et reglement som setter grenser for hvor mye trykk et hydraulisk system får produsere, og som forbyr utslipp av de fleste former for hydraulikkvæske. Ved å bruke elektriske motorer unngår vi å bruke tid på å gjennomgå et omfattende sett av retningslinjer. Erfaringer fra tidligere år viser at elektriske motorer egner seg godt for å løse de oppgavene vi bygger ROV-en for.

En elektrisk drevet motor omformer elektrisk energi til mekanisk bevegelse [41]. Det finnes flere typer elektriske motorer, men felles for de ulike typene er at bevegelsen oppstår som følge av elektromagnetisme ved strømføring. Motorene består i all hovedsak av en faststående del, *stator*, og en fritt roterende del, *rotor*, som er montert sammen på en aksling. Begge komponentene må ha hvert sitt magnetfelt, enten det kommer fra permanentmagneter eller induksjon i spoler. Magnetisk tiltrekningskraft kraft vil da få rotoren til å rotere relativt til statoren [7]. Nøyaktig hvordan dette foregår avhenger av hvilken type elektrisk motor som blir brukt. Videre i kapitlet vil vi derfor gå nærmere inn på de typene vi har vurdert til ROV-en: *børstet DC-motor*, *børsteløs DC-motor* og *stegmotor*.

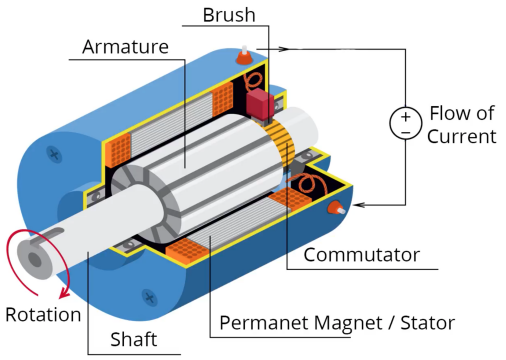
2.1.1 Børstede DC-motorer

Teori i dette delkapittelet er hentet fra artikkelen *Brushed DC electric motor* [55].

Børstede DC-motorer (BDC-motorer) kan være basert på forskjellige prinsipper, men et typisk eksempel på hvordan børstemotoren kan fungere vises i figur 2.1a. Statoren består av permanentmagneter, eller elektromagneter, som setter opp et konstant magnetfelt. Rotoren, som består av et armatur av viklinger (i figuren illustrert med kun én vikling), er plassert mellom statorens to poler. Rotoren spenningssettes ved hjelp av fjærbelastede børster som presses inn mot en kommutator. Kommutatoren er delt opp i elektrisk ledende sektorer, der hver sektor er koblet til motstående endepunkter av en vikling i rotoren. I figuren er det kun to slike sektorer, da vi bare har én vikling i armaturet. Når det går strøm i viklingen vil statorens magnetfelt påføre magnetisk kraft på hver side av viklingen for å få den til å rotere. Kommutatorens sektorer vil endre polaritet ettersom viklingen roterer, og strømretningen endres da for hver halve rotasjon. Dette sørger for at den magnetiske kraften påføres rotoren kontinuerlig i rotasjonsretning.



(a) Prinsippskisse for virkemåten til børstet dc-motor. Ved å spenningssette kommutatoren vil det gå strøm i armaturviklingen. Strømmen vil sammen med magnetfeltet fra stator påføre krefter på viklingen for å få den til å rotere. Kraften er maksimal når viklingen ligger parallelt med magnetfeltlinjene. Kraften vil i dette tilfellet variere mye, da den kun når maksimum to ganger per rotasjon. Hentet fra [45].



(b) Realisering av prinsippet i en mer virkelighetsnær fremstilling av motoren. Armaturet består som oftest av flere viklinger, og kommutatoren er derfor delt opp i flere deler. Hver vikling er ikke lenger alltid strømførende. Kun når den står mest mulig parallelt med magnetfeltlinjene vil den få strømforsyning. Dette sørger for et mye mer stabilt og effektivt kraftbidrag til rotoren. Hentet fra [14].

Figur 2.1

Man ønsker å utnytte mest mulig av den elektriske energien som blir tilført motoren. Dette resulterer i en mer effektiv motor. For å få til dette må dreiemomentet på motoren maksimeres. Dreiemomentet, τ, er gitt av følgende formel [57]:

$$\tau = | \vec{r} | \cdot |\overrightarrow{F_m}| \cdot sin(\theta) \tag{2.1}$$

hvor

- $\vec{r}$ er posisjonsvektoren fra rotasjonspunktet til punktet på viklingen der kraften virker.
- $\overrightarrow{F_m}$ er det magnetiske kraftbidraget.
- θ er vinkelen mellom posisjonsvektor og kraftvektor.

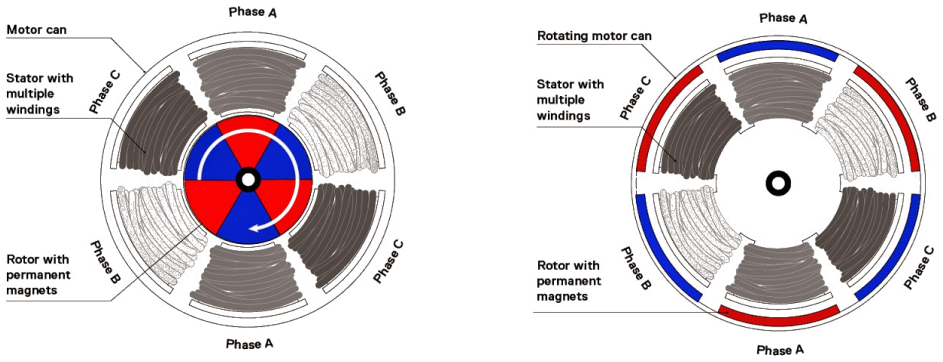
Ifølge ligning 2.1 vil altså dreiemomentet maksimeres når vinkelen θ er 90^o. I figur 2.1b ser man hvordan motoren kan være realisert. Vanligvis legger man til flere viklinger og deler kommutatoren opp i tilsvarende flere sektorer, to motstående sektorer for hver vikling. Dette sørger for at hver vikling kun er spenningssatt et lite øyeblikk hvor de står mest mulig parallelt med magnetfeltet fra stator. Den magnetiske kraften vi da virke tilnærmet 90^o på rotoren hele tiden, og skape en jevn rotasjonshastighet.

En av de store fordelene med børstemotoren er at den ikke krever en avansert motorkontroller for å kunne kjøres. Likevel kan motoren by på utfordringer ved bruk i en ROV. Kontakt mellom børster og kommutator sørger for kontinuerlig slitasje, og vil kreve vedlikehold med jevne mellomrom. Fysisk kontakt mellom faststående børster og en roterende kommutator vil også skape problemer knyttet til vanntetting av motoren. I konkurransemanualen er det fastslått at vanntetting må utføres, men dette ser vi nærmere på i delkapittel 2.4. I tillegg er det behov for sensorer for å kunne lese av posisjonen til motoren. Utfordringene rundt dette gjennomgår vi i kapittel 3.

2.1.2 Børsteløse DC-motorer

Teori i dette delkapittelet er hentet fra artikkelen *Brushless DC electric motor* [56].

Børsteløse DC-motorer (BLDC-motorer) er, som navnet indikerer, ikke basert på prinsippet med børster. I BLDC-motoren går det ingen strøm i rotor. Denne er satt opp med permanentmagneter. Statoren består av motstående spolepar som blir spenningssatt og virker som elektromagneter. Et eksempel på hvordan motoren kan se ut, er vist i figur 2.2. Rotasjonen skapes av den magnetiske tiltrekningskraften mellom elektromagnetene i stator og permanentmagnetene i rotor. Prinsippet er det samme som for børstemotoren, men kommuteringen skjer på en annen måte. Kommutasjon skjer ved å å spenningssette statorspolene i bestemte mønster. Det er da mulig å skape et roterende magnetfelt som rotoren vil følge. For å sette opp et jevnt roterende magnetfeltet som er mulig for rotoren å følge, behøver motoren en motorkontroller. Denne består av en mikrokontroller med tilhørende elektronikk, og vil bli nærmere forklart i kapittel 3.



Figur 2.2: Figuren viser en BLDC-motor med seks statorspoler og seks permanentmagneter i rotor. Motstående spoler er koblet til hver sin fase, der kun to faser fører strøm om gangen. Da motoren er koblet opp med tre faser, og kun to skal være tilkoblet om gangen, er det seks mulige kombinasjoner av fasetilkobling. Disse seks kombinasjonene utgjør stegene i en kommutasjonssekvens. Motoren til venstre er av typen *inrunner*, og motoren til høyre er av typen *outrunner*. Hentet fra [34]

Figuren over viser at det er forskjeller i konfigurasjon på BLDC-motorer. En motor der rotoren er plassert innenfor statoren, kalles en *inrunner*. Rotoren kan også være plassert på utsiden av stator, og man kaller den da en *outrunner*. For å forstå forskjellen på de to typene trenger vi disse sammenhengene som gjelder for både BDC- og BLDC-motorer [20]:

$$E_b = K_e \cdot \omega \tag{2.2}$$

$$\tau = K_T \cdot I \tag{2.3}$$

hvor

- E_b er indusert spenning som følge av bevegelse mellom viklinger og permanentmagneter. Denne kalles *back EMF* (BEMF), og blir nærmere forklart i kapittel 3.
- K_e er motorens BEMF-konstant.
- ω er motorens rotasjonshastighet.
- τ er dreiemomentet.
- K_t er motorens moment-konstant.
- I er strøm.

I tillegg oppgir man ofte en motorkonstant, K_v, som har sammenheng med begge ligningene over.

$$K_v = \frac{1}{K_e} \tag{2.4}$$

$$K_v = \frac{1}{K_t} \tag{2.5}$$

Det viser seg altså at K_e og K_t er ekvivalente konstanter. De noteres med forskjellige enheter, men disse er også ekvivalente.

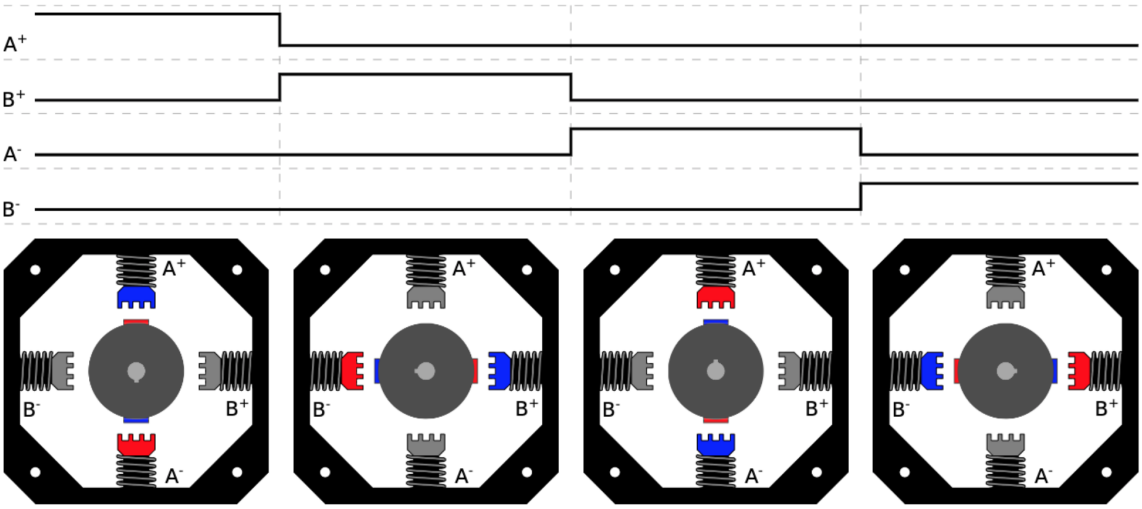
Med sammenhengene over i bakhånd, kan man enklere forstå forskjellen mellom en inrunner og en outrunner. Hovedforskjellen mellom de to typene, er hvordan utformingen på dem påvirker motorkonstanten K_v [33]. For en inrunner og outrunner i samme størrelsesorden, vil K_v for inrunneren være større. Da E_b er begrenset til å ikke være større enn forsyningsspenningen, ser man fra ligning 2.2 og 2.4 at større K_v tillater større rotasjonshastighet. Ser man videre på ligning 2.3 og 2.5, vil en større K_v gi et mindre dreiemoment. Forskjellene i K_v kommer av at outrunneren har en lenger momentarm. Avstanden fra rotasjonssenter til ytterkanten av rotoren er større. Dette gjør at det er plass til flere permanentmagneter på rotoren til outrunneren. Flere magneter betyr også flere kommutasjonssteg per rotasjon. Dette resulterer altså i bedre dreiemoment, men lavere rotasjonshastighet.

Den største fordelen med BLDC-motoren i forbindelse med konstruksjon av ROV-en er at den ikke er basert på børster. Levetiden på motoren vil øke betraktelig ettersom det ikke er fysisk kontakt mellom bevegelige deler. Lite slitasje gjør at behovet for jevnlig vedlikehold reduseres. Da statoren er den eneste strømførende delen, vil også vanntetting av motoren bli lettere. Å støpe statoren inn i et elektrisk isolerende materiale er mer ideelt, da den er adskilt fra rotoren. Utfordringen med BLDC-motorer er at også disse har vanskeligheter med å vite hvilken posisjon de står i. Dersom man vil vite rotorposisjonen, er man avhengig av en sensor som kan lese av denne. I tillegg er det for BLDC-motoren behov for en relativt avansert motorkontroller. Denne sørger for at rotoren kan klare å følge det roterende magnetfeltet.

2.1.3 Stegmotorer

Teori i dette delkapittelet er hentet fra artikkelen *Creating Precision Robots* [13].

Stegmotoren har flere statorspoler, gjerne fire eller åtte. Disse fungerer som elektromagneter når de blir spenningssatt. Motstående statorspoler i stegmotoren er elektrisk koblet til hverandre, og definerer en fase. Spolene i samme fase er viklet slik at når man spenningssetter fasen, vil de to spolene få motsatt magnetisk polaritet. Stegmotoren i figur 2.3 har fire statorspoler, og man kjører den derfor med to faser. Ved å veksle mellom hvilken fase som er spenningssatt, og hvilken retning den er spenningssatt, får man rotoren til å bevege seg steg for steg.



Figur 2.3: Stegmotor med fire statorspoler og to poler på rotoren. Figuren viser en rotasjonssekvens med tilhørende inngangssignaler på fire steg. Fase A er tilkoblet statorspole A^+ og A^-. Fase B til B^+ og B^-. Magnetisk nord er symbolisert med rød farge, og magnetisk sør med blå. Eksekveringssekvensen over kalles *wave drive*, og kjennetegnes ved at kun én fase er spenningssatt om gangen. I steg 1 er høyt spenningsnivå tilkoblet statorspole A^+. Strømmen går da inn på A^+, og til jord via A^-. Denne kjøremetoden gjør at stator til en hver tid har to motsatte poler. Da rotoren også har to poler, blir hvert rotasjonssteg 90°. Hentet fra [10].

Rotoren, bestående av to eller flere magnetiske poler, vil rotere i steg som følge av tiltrekningskrefter fra statorens magnetfelt. Størrelsen på stegene avhenger av antall statorspoler, antall poler på rotoren og signalsekvensen som fasene aktiveres i. Eksempelvis vil en vanlig stegmotor med åtte statorspoler og 50 poler (tenner) på rotoren gjerne kjøre 200 steg per omdreining. Dette tilsvarer $360° \div 200 \text{ steg} = 1.8°/\text{steg}$.

En fordel med stegmotoren er at den har et godt holdemoment. Det kreves mer kraft fra utsiden for å rotere en stillestående stegmotor, enn det kreves for en BDC/BLDC-motor. Et annet stort pluss med stegmotoren er muligheten til å kunne vite nøyaktig posisjon til enhver tid uten bruk av sensorer. Dette krever dog en oppstartsfunksjon for å vite hvilken stabile posisjon motoren inntar ved oppstart. Da man vet stegstørrelsen til motoren kan man også beregne hvor mye den har rotert i hver retning ettersom hver endring i signalsekvensen fører til ett steg. En ulempe med stegmotoren kan være at den vanligvis kjøres med en relativt lav rotasjonshastighet. I tillegg har den, som BLDC-motoren, også behov for en motorkontroller. Dette er ikke nødvendigvis et stort problem, men det gjør at vi må bruke tid på enda en komponent.

2.2 Motorer for manøvrering

For at en ROV skal være lett å styre, er valget av thrustere og konfigurasjonen av disse essensielt. En stor del av grunnlaget for valgene vi har gjort i forbindelse med thrustere, er basert på erfaring fra tidligere år [21] [5] [9].

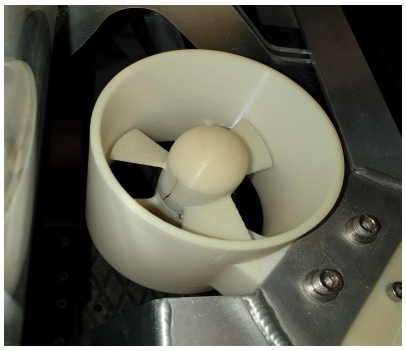
2.2.1 Motorvalg

Thrusternes oppgave er å bevege hele ROV-en i vann, og egenskapene til de ulike motortypene gjennomgått i forrige delkapittel vil ha innvirkning på hvilke thrustere som blir valgt. Et mål for årets prosjekt, er at vekten til ROV-en skal reduseres vesentlig i forhold til fjorårets ROV. Thrustere står for en betydelig andel av den totale vekten, og dette må tas hensyn til. I tillegg er prosjektet underlagt en studentorganisasjon, og økonomiske midler vil være begrenset.

Å finne alternativer av ulike thrustere er ingen stor utfordring, snarere tvert imot. Man finner utrolig mange måter å realisere en thruster på kun med et enkelt nettsøk. For å avgrense mulighetene bestemte vi oss tidlig for å bruke BLDC-motorer. Det er mindre vanlig å bruke stegmotoren som thruster. Den er hovedsaklig laget for presisjonskjøring, og ikke høy rotasjonshastighet. Børstemotoren krever mer vedlikehold, og er i tillegg verre å vanntette dersom det skulle være aktuelt. BLDC-motoren er effektiv, og har vært vellykket som thruster tidligere år. Den krever en motorkontroller, men disse har med årene både blitt bedre og mindre i størrelse. Valget av thruster måtte tas tidlig i prosessen, da det påvirker flere andre aspekter av hele prosjektet. Derfor kokte vi valget fort ned til tre alternativer:

Egenprodusert thruster basert på en BLDC-motor

Flere av de tidligere ROV-ene fra UiS Subsea har brukt egenproduserte thrustere. En fordel for oss er at dette konseptet har blitt testet før, og har sett ut til å være vellykket. Egenproduserte thrustere vil også være et billig alternativ fremfor å kjøpe nye på markedet. BLDC-motorer er relativt billige, og propeller kan man 3D-printe på universitetet. Likevel kan dette være en tidkrevende prosess. Som nevnt tidligere må selve motoren vanntettes. Propeller og motorhus må også designes og testes. Det har tidligere blitt kjørt tester av propellkarakteristikk, som avslørte at innkjøpte propeller hadde jevnere karakteristikk og var mer effektive enn de som ble 3D-printet på universitetet [9].



Figur 2.4: Bildet er tatt av en tidligere 3D-printet thruster fra UiS Subsea.

P1000 – ThrustMe

Vurderingene av ThrustMe sin thruster er gjort basert på erfaringene fra fjorårets ROV, som brukte nettopp denne [21]. Den klart største fordelen med denne er at i år, som i fjor, ville ThrustMe bidra som sponsor med gratis thrustere og tilhørende motorkontrollere. Kontrollerene er også vanntette, som betyr at de kan plasseres på utsiden av elektronikkhuset. Dessverre har ThrustMe opplyst om at thrusterene ikke er tilgjengelige før mars måned. Dette vil ha betydelig innvirkning på fremdriften i prosjektet, da vi er avhengige av å få testet både selve motorkarakteristikken og programkode vi lager for styring. I tillegg er P1000 vesentlig tyngre enn andre alternativer, med oppgitt vekt på 800 gram per thruster [49]. Ved 12 VDC tilførsel er thrusten oppgitt til å være 6,5 kg. Ut fra erfaringene til fjorårets team, har vi blitt fortalt at dette er i overkant av hvor mye kraftbidrag vi trenger.

Figur 2.5: P1000 Thruster. Bildet er hentet fra [49]

T200 – Blue Robotics

T200 fra Blue Robotics er en thruster designet for ROV-er i samme størrelsesorden som den vi utvikler. Thrusteren har en oppgitt vekt på 344 g [36], som er under halve vekten til P1000. Ved 12 VDC er thrusten maksimalt 3,7 kg. Dette vil være fordelaktig, da vi ønsker mindre kraftbidrag i år enn i fjor. Vi har snakket med den norske leverandøren JM Robotics, og leveringstid vil være på omtrent en uke. For å komme fort i gang med testing av kraftbidrag og modellering av systemet, vil dette være optimalt. Det er også mulig å kjøpe tilhørende motorkontrollere fra Blue Robotics designet for denne thrusteren. En ulempe med disse kontrollerne er at de ikke er vanntette, og vi kan bli nødt til å plassere dem inne i elektronikkhuset. JM Robotics har sagt seg villige til å gi oss rabatt på produktene, men dette vil fortsatt være det dyreste alternativet.

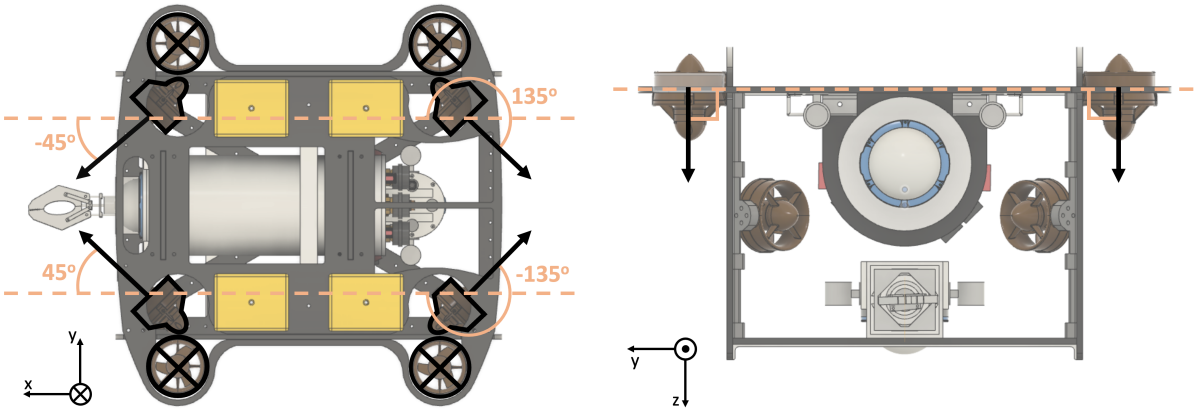
Figur 2.6: T200 Thruster. Bildet er hentet fra [36]

Konklusjon

Ett av hovedmålene for årets prosjektgruppe er å få testet ROV-en i vann så tidlig som mulig. Ved å velge en ferdig vanntettet thruster sparer vi mye tid på produksjon og testing. Den totale vekten på ROV-en skal bli mindre enn i fjor, og da thrustere vil være et hovedbidrag på totalvekten har vi valgt å prioritere dette over pris. Valget falt altså på T200 fra Blue Robotics. Med leveringstid på en uke vil vi kunne komme fortere i gang med å sette opp systemet, og forhåpentligvis få testet og optimalisert det før innlevering av bacheloroppgaven.

2.2.2 Konfigurasjon

Konfigurasjonen av thrusterene på ROV-en vil ha stor innvirkning på hvordan styresignaler gis ut til hver enkelt thruster. Alle tidligere ROV-er fra UiS Subsea har brukt åtte thrustere, fire for horisontal bevegelse, og fire for vertikal bevegelse. Dette har vært et vellykket oppsett, og vi har derfor valgt å ikke gjøre store endringer i år. Likevel er det nyanseforskjeller som er verdt å presentere. Figur 2.7 viser hvordan thrusterene er orientert.



(a) Motorkonfigurasjonen sett ovenfra. Motorene er positivt rettet den veien pilene peker. For de vertikale motorene viser krysset at pilen peker nedover.

(b) De vertikale thrusterene ligger alle i samme plan. De horisontale ligger i et eget plan lenger nede.

Figur 2.7

De horisontale thrusterene er satt opp med en vinkel i forhold til foroverretning. Man kan vinkle thrusterene for å kunne gi mer bidrag i enten x- eller y-retning, se figur 2.7a. Årets MATE-konkurranse krever like mye sideveis bevegelse som forover og bakover. Vi har derfor valgt å orientere thrusterene med 45° fra x- og y-aksen. Dette sørger for at thrusterne kan bidra med like mye kraft langs begge akser. De fremre thrusterene er satt til å være positive forover, og bakre thrustere positive bakover. Dette gjør det enklere å bevege ROV-en langs aksene uten at den roterer som følge av ulikheter i kraftbidrag. Thrusterene bidrar nemlig med ulik kraft i positiv og negativ retning. Dette ble verifisert gjennom en test som er gjengitt i vedlegg B.1. De horisontale thrusterene er i tillegg plassert i et plan tett opp mot sentrum av elektronikkhuset. Her vil det være mest motstand i vannet under kjøring. Med denne plasseringen av thrusterne, begrenser vi tipping av ROV-en når vi kun ønsker horisontal bevegelse.

De vertikale thrusterene er plassert på hvert sitt hjørne. Dette bør være en god oppstilling for å kunne holde ROV-en stabil i vannet uten at den tipper forover, bakover, eller til siden. Det forenkler prosessen med å regne ut pådrag under regulering. Thrusterne er også orientert med positiv retning nedover. Dette har vi valgt å gjøre i forbindelse med at nedoverretning blir i kapittel 6 definert som den positive dybderetningen.

I utgangspunktet var det tenkt å montere de to parene med diagonalt motstående thrustere motsatt retning av hverandre. Siden thrusterne er mer effektive i den ene retningen, bruker motorene mindre strøm for å gi samme kraftbidrag som i den andre retningen. Ved å bruke den tenkte konfigurasjonen, kunne det blitt forhindret at alle fire thrusterene blir drevet på 100% pådrag samtidig. Ved å gi maks pådrag i en retning vil alltid to av thrusterne skaleres ned med rundt 20%[1]. Da vi har et begrenset effektbudsjett, ville dette hjulpet til med å holde effekten nede når ROV-en beveges i z-retning. Gjennomsnittet av effektbruk ville ikke bitt endret, men maks effektbruk ville blitt senket betraktelig. Ved å dempe pådraget til 80% på to thrustere, kunne maksimalt effektbruk blitt dempet med opp mot 150 W. Denne konfigurasjonen ble likevel ikke valgt. Gruppen med ansvar for design av rammen til ROV-en monterte alle de fire thrusterne i samme retning, da festeanordningen allerede var designet på denne måten.

2.3 Motorer for manipulator

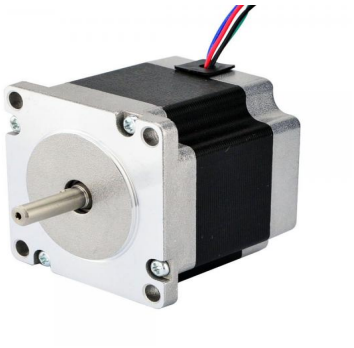
For at ROV-en skal kunne gjennomføre alle oppgavene i MATE-konkurransen, trengs det en manipulatorarm. Basert på oppgavenes vanskelighetsgrad, har det av årets prosjektteam blitt bestemt å ha tre frihetsgrader på manipulatoren. Styring av de tre frihetsgradene skjer ved hjelp av én aktuator for hver av dem. Manipulatoren må designes robust, og være i stand til å holde objekter på opptil 10 N i vann [23]. Aktuatorenes egenskaper må derfor velges med dette tatt i betraktning.

Egenskapene til de ulike motortypene ble gjennomgått i delkapittel 2.1. Av samme årsak som for thrusterne, valgte vi å utelukke børstede DC-motorer. Da vi ikke lenger velger spesifikt blant thrustere, men en generell elektrisk motor, blir alternativene mange. Vurderingene våre går derfor kun på hvilken type motor vi velger.

[1] Se testrapport B.1

2.3.1 Motorvalg

Stegmotor

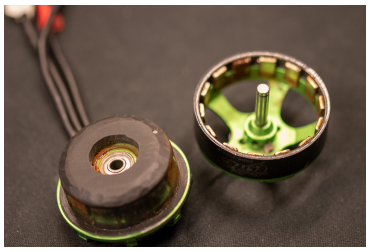


Figur 2.8: Bildet viser Nema 23 stegmotor. Hentet fra [46]

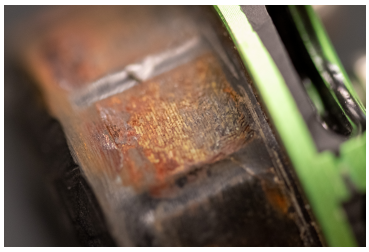
Stegmotoren var naturlig å vurdere. Denne ble brukt i fjorårets ROV [21], og det ligger også flere av disse igjen på universitetet etter tidligere år. Ved å ta i bruk en motor vi allerede har, sparer vi både tid og penger. Motorene er av typen Nema 17 og Nema 23. Det at stegmotoren kan kjøres med faste steg, gjør at vi kan styre manipulatorarmen med høy presisjon. Dette vil komme oss til nytte dersom vi ønsker å stille den inn i gitte posisjoner. Holdemomentet i stegmotoren er også en stor fordel. Manipulatorarmen vil kunne bli utsatt for en del ytre påvirkninger, og et godt holdemoment vil derfor kunne motvirke uønsket bevegelse. Den største ulempen med å bruke en stegmotor, er vekten. Aktuatorene må, i likhet med thrusterene, være vanntette. Vekten på en Nema 23 med vanntett kapsel ble i fjor så mye som 1.271 kg [21]. Motoren kjøres på 2,5 V og 2,8 A. Ingen andre komponenter i ROV-en kjører på dette lave spenningsnivået, og kraftforsyningsgruppen blir derfor nødt til å legge opp til enda en nedregulering. I tillegg vil det å bruke en stegmotor kreve en annen type motorkontroller enn den thrusterne styres med.

BLDC

BLDC-motor har også blitt brukt av tidligere team fra UiS Subsea, med vellykket resultat. Den største fordelen med BLDC-motoren er vektbesparelse. Motoren vi vurderer er en Eaglepower 3508 [32], og har en oppgitt vekt på 88 g. Dersom vi inkluderer vekten fra epoksy til vanntetting, vil denne motoren fortsatt være over 1 kg lettere enn stegmotoren. Dette summerer seg til en betydelig vektforskjell dersom vi skal ha tre motorer av samme type. Om valget faller på en BLDC-motor, vil det også kunne brukes samme motorkontrollere som for thrusterne. Dette vil gjøre det lettere å styre motorene. Eaglepower-motoren kan også kjøre på både 12 V og 19 V, som er to nivåer det er lagt opp til i elektronikkhuset. En av ulempene med BLDC-motorer, er at man går glipp av mulighetene for å kjøre etter nøyaktig posisjon. Man mister i tillegg det gode holdemomentet stegmotoren kan bidra med. Dette kan likevel kompenseres for mekanisk med bruk av ulike giringsprinsipper. Vi hadde liggende flere Multistar Elite 3508 BLDC-motorer etter tidligere, men disse var nedslitt, og vil kunne lekke strøm i bassenget. Derfor vil det også medføre ekstra kostnad ved å bruke Eaglepower-motoren. Figur 2.9 viser begge de to BLDC-motorene vi har nevnt over.



(a) Multistar Elite 3508. En motor som tidligere år har fungert godt.



(b) Epoksyen er nedslitt, og statoren har begynt å ruste.

(c) Eaglepower 3508. En motor med flere av de samme egenskapene som Multistar-motoren.

Figur 2.9

Konklusjon

Vurderingene av manipulatormotorer ble gjort sammen med manipulatorgruppen **REFERER TIL DERES RAPPORT**. I likhet med valget av thrustere, vil vekten av motorene være utslagsgivende. Vi vil kunne redusere vekten med omtrent 3 kg dersom vi velger BLDC-motoren. Det å kunne bruke samme motorkontrollere er også en stor fordel som trekker oss i den retningen. Manipulatorgruppen har bekreftet at holdemomentet kan løses mekanisk, og derfor vil det ikke medføre et stort tap å velge bort stegmotoren. I tillegg krever ikke årets oppgaver i MATE-konkurransen stor nøyaktighet i forhold til å posisjonere de ulike leddene av armen. BLDC-motoren kombinert med et godt girsystem vil derfor kunne gi piloten god nok presisjon på øyemål via kameraet i ROV-en. Konklusjonen blir altså at vi bruker en Eaglepower 3508 for alle tre frihetsgrader på manipulatorarmen.

2.4 Konkurransevilkår – vanntette motorer

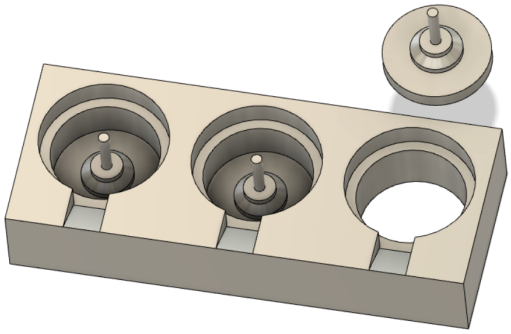
I kapittel 1 ble det nevnt at konkurransemanualen fra MATE stiller krav til motorer eksponert for vannet. Disse må være elektrisk isolerte, med en resistans på minimum 10 MΩ.

De børsteløse motorene brukt på manipulatoren har koblinger som vil være eksponerte for vannet. Kobbertrådene i statorviklingene er isolert fra hverandre med lakk, og skal derfor ikke lede strøm over i vannet. Likevel er koblingene fra faseledninger til statorspoler utsatt. Det samme gjelder koblingene fra statorspolene til nøytralpunktet i stjernekoblingen på motoren. Her er det kun brukt krympestrømper, og vi blir derfor nødt til å isolere ytterligere.

MATE foreslår i konkurransemanualen å støpe motoren inn i epoksy, som er et velegnet materiale for innstøping. Det finnes forskjellige typer epoksy, men det er ikke spesifisert hvilken type det er anbefalt å bruke til dette formålet. For å forsikre oss mot overoppheting av motoren, har vi brukt en epoksy med gode termiske ledeegenskaper, kalt *832TC*. I følge databladet [4] har denne en termisk ledeevne på 0.7 W/mK, og en resistivitet på $8.2 \cdot 10^{12}\ \Omega \cdot \text{cm}$.

Måten man isolerer motoren, er å støpe inn statoren. Her er det viktig å sørge for at epoksyen ikke ødelegger for fri bevegelse mellom stator og rotor. Epoksyen må dekke statoren, uten å være i fysisk kontakt med rotor. Da luftgapet mellom rotor og stator er 0.25 mm, kreves det

en støpeform med veldig presise mål. Vi tegnet derfor opp en CAD-modell av en støpeform, og fikk den produsert i CNC-maskinen på verkstedet. Både CAD-modellen, og den ferdigproduserte formen er vist i figur 2.10. Formen er produsert med løse bunnplater til hver motor. Dette er gjort for at det skal være mulig å få motoren ut av formen etter innstøping. Hver bunnplate har også en aksling som hindrer epoksyen fra å renne ned i kjernen av motoren.



(a) CAD-modell av støpeformen tegnet i Fusion 360.

(b) Ferdig produsert støpeform.

Figur 2.10

Epoksyen har en høy viskositet ved romtemperatur, og det ble derfor brukt en varmeovn til å varme opp epoksyen til 70 °C. Ved denne temperaturen er epoksyen mer flytende, men for å sikre at den flyter godt inn i de trange delene av motoren, samt at mengden luftbobler minimeres, ble det i tillegg brukt et vakuumkammer. Ved å periodevis trekke vakuum for så å øke trykket, ble epoksyen trukket inn i de trange delene av motoren, mens luftboblene ble tvunget til overflaten. For å sikre at epoksyen ikke heftet seg til selve støpeformen, ble støpeformen i forkant smurt med formfett. Dessverre kom noe av dette fettet på ytterkanten av statoren når denne ble plassert i formen, og epoksyen løsnet fra den ytterste delen av statorkjernen.

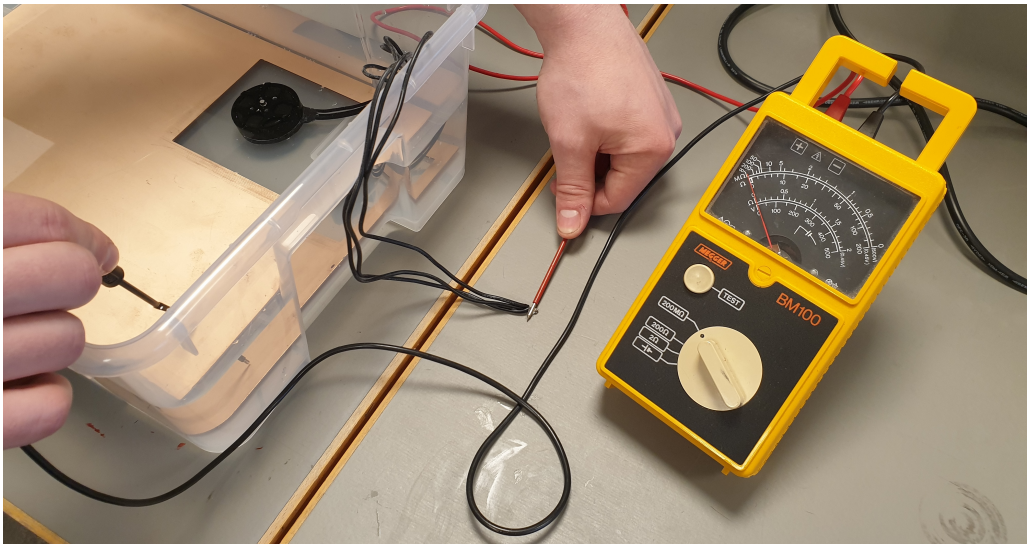
Epoksyen ble herdet ved å plassere formen i et varmeskap på 70 °C i 2 timer. Deretter kunne epoksyen slipes ned for å ikke komme i konflikt med rotoren. Sidene på statoren ble slipt ned slik at epoksyen som ikke heftet skikkelig ble fjernet. Selv om dette førte til at metallet fra statorkjernen ble eksponert, påvirkes ikke isolasjonsevnen. Statoren ble allikevel penslet med et tynt lag epoksy for å bremse korrosjonen av statorkjernen. I figur 2.11a vises staoren før den ble støpt i epoksy, og i figur 2.11b vises statoren etter den var ferdig behandlet.

(a) Motoren før innstøping.

(b) Motoren etter innstøping.

Figur 2.11

For å verifisere at innstøpingen var vellykket, brukte vi en isolasjonsmåler, eller *megger*. Dette er et instrument som måler motstand i megaohm. Meggeren vi benyttet oss av, måler motstand ved 500 VDC spenning. Vi brukte et lite kar med vann for å måle isolasjonsevnen til motoren i riktige omgivelser. Den ene proben på meggeren koblet vi til en kobberplate som lå i vannet. Den andre koblet vi til fasene på motoren. Figuren under viser hvordan vi utførte målingene.



Figur 2.12: Bildet viser oppsettet for å måle isolasjonsevnen til motoren med en megger.

Vi målte en motstand på over 200 MΩ, og kan derfor konkludere for at vi er godt innenfor retningslinjene fra konkurransemanualen.

2.4.1 Konklusjon

For å legge et grunnlag for valget av motorer, ble det innledningsvis i kapittelet presentert virkemåten til forskjellige motortyper. Basert på dette ble det bestemt å bruke ferdigproduserte

thrustere med børsteløse motorer. Disse er av typen T200 produsert av *Blue Robotics*. Fordelen med disse thrusterne, er at de er utviklet spesifikt for den bruken som behøves i dette prosjektet. I tillegg er de lett tilgjengelige, med rask leveringstid. Effektbruk og kraftnivå er også innenfor rammene i den skalaen ROV-en er utviklet. Bakdelen med T200 er at de var det dyreste alternativet vi vurderte. Til manipulatorarmen ble det også valgt å bruke børsteløse motorer. Valget ble tatt hovedsaklig for vektreduksjonen ved å velge disse over andre alternativer. Motorene er av typen *Eaglepower 3508*. Ulempen med BLDC-motorer for manipulatoren er mangel på et godt holdemoment, men dette er vurdert til å kunne løses med mekaniske prinsipper. På grunn av krav om vanntette motorer, ble disse motorene i tillegg støpt i termisk epoksy.

Kapittel 3

Motorkontrollere

Kapitteloversikt

Som nevnt i kapittel 2 er alle motorene i systemet avhengig av en motorkontroller hver for å kunne kjøres. Dette kapittelet vil ta for seg den overordnede virkemåten til en generell motorkontroller, og de vurderingene vi har gjort for å velge en kontroller som passer for vår ROV. Videre vil vi gå mer spesifikt inn på virkemåte og oppsettet av elektriske kretser på den vi har valgt. Dette gjøres for å få et innblikk i hvordan motorkontrolleren genererer signaler for å drive motoren.

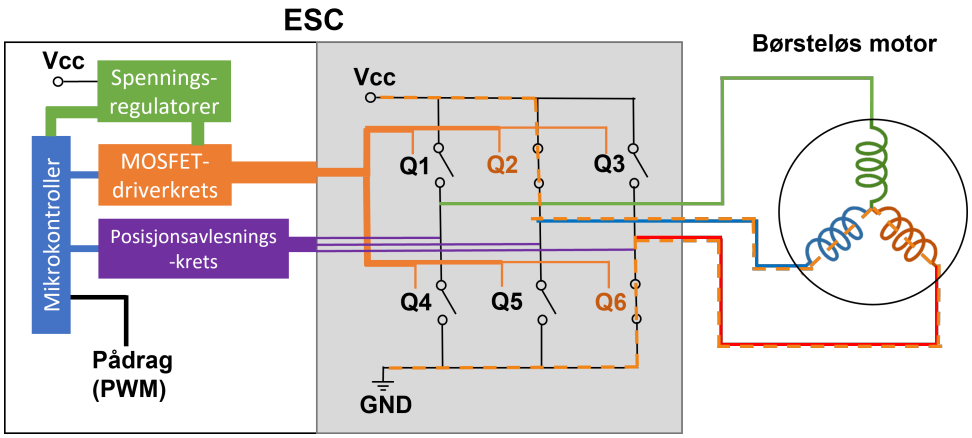
3.1 Virkemåte

En motorkontroller har som oppgave å drive en elektrisk motor etter en ønsket oppførsel. Det finnes utallige typer motorkontrollere, og ønsket virkemåte varierer fortrinnsvis ut fra hvilken type elektrisk motor som skal styres. Likevel kan det også være mindre forskjeller i hvordan bestemte motortyper styres. For å ikke bevege oss for langt bort fra det som er relevant for vårt system, har vi valgt å fokusere på en motorkontroller av typen *ESC*[1] for børsteløse motorer.

[1] **E**lectronic **s**peed **c**ontroller

Teori videre i dette delkapittelet er hentet fra artikkelen *Electronic speed control* [53].

ESC-en er en smart elektrisk krets, som betyr at kretsen er bygget opp rundt en mikrokontroller. Dette gjør det mulig å styre de børsteløse motorene som brukes på ROV-en med ønsket hastighet, og jevnt driv. I figur 3.1 ser vi et eksempel på en prinsippskisse av virkemåten til en standard ESC, og hvilke moduler den er bygd opp av.



Figur 3.1: I skissen er MOSFET Q2 og Q6 aktivert. Dette gjør at strømmen går fra blå fase, via rød fase og ned til jord. Skissen er hentet fra [6]

For å skape det roterende magnetfeltet som rotoren skal følge, er de tre fasene inn til motoren koblet til et sett av MOSFET-er[2]. Strømmen vil gå gjennom to av fasene i motoren, alt etter hvilke MOSFET-er som er aktivert. Det roterende magnetfeltet skapes følgelig ved å aktivere MOSFET-ene i et bestemt mønster bestående av seks steg. Dess fortere sekvensen går, dess fortere roterer motoren.

For å styre MOSFET-ene er *gate*-terminalen på hver av dem koblet til en MOSFET-driverkrets. Denne driverkretsen mottar signaler fra mikrokontrolleren på ESC-en om hvor i sekvensen vi befinner oss, og om vi skal videre til neste steg. Den sørger også for at hver gate-terminal påtrykkes det spenningsnivået som kreves for å aktivere tilhørende MOSFET.

Mikrokontrolleren er avhengig av informasjon om posisjonen til rotoren for å vite når den skal sende ut signal til driverkretsen om å gå over i neste steg. Uten posisjonsavlesning ville ikke det roterende magnetfeltet være mulig for rotoren å følge, og motoren ville endt opp med ingen, eller i beste tilfelle en hakkete rotasjon. Som man ser i figuren, er det derfor koblet en avleserkrets for posisjon til hver fase. Denne vil forklares nærmere senere i kapittelet.

ESC-en er også utstyrt med en eller flere spenningsregulatorer. Disse brukes til å nedskalere inngangsspenningen for å kunne forsyne mikrokontrolleren og MOSFET-driverkretsen. Dette gjør at inngangsspenningen ikke trenger å være en spesifikk spenning, så lenge den ligger innenfor et gitt spenningsområde.

Rotasjonshastigheten på motoren styres med pulsbreddemodulasjon, eller ofte omtalt som et PWM[3]-signal. Dette er et lavspenningssignal med varierende pulsbredde. Signalet tolkes i mikrokontrolleren på ESC-en, og brukes til å styre hastighet og retning på kommuteringen av MOSFET-

[2] **M**etal–**o**xide–**s**emiconductor **f**ield-**e**ffect **t**ransistor
[3] **P**ulse **w**idth **m**odulation

ene gjennom de seks stegene i kommuteringssekvensen. For systemet vårt er det fordelaktig at det brukes et PWM-signal for å styre motorene, siden dette genereres relativt enkelt i mikrokontrolleren vi skal bruke, og sørger for at vi kan gi ønskede pådrag til alle motorene simultant.

3.2 Valg av motorkontroller

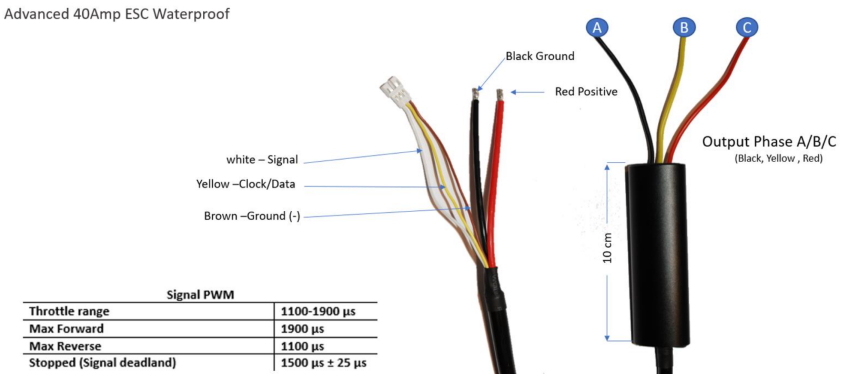
En av fordelene ved å bruke BLDC-motorer til både manøvrering og til manipulator, er at vi kun har behov for én type motorkontroller. For motorkontrollere finnes det, som for motorer, utallige alternativer å velge mellom. Ut fra tilgjengelighet og relevans, ble det derfor bestemt å vurdere tre ulike typer.

3.2.1 Benjamin Vedders ESC (VESC)

I tiligere år har motorkontrolleren fra Benjamin Vedder blitt brukt. Dette er en kraftig motorkontroller som har mange justerbare parametere, og er originalt laget for elektriske skateboard. VESC kan kalibreres og tunes ved hjelp av programmet *BLDC-tool*, også laget av Benjamin Vedder. VESC har blant annet mulighet for å kommunisere over CAN-buss, som er lett å implementere ettersom resten av ROV-en kommuniserer med samme protokoll. Noen eksemplarer av VESC var tilgjengelig etter tidligere år, men det ville vært nødvendig å kjøpe inn flere. Dette vil være relativt kostbart ettersom én av disse motorkontrollerne koster opp mot 1000 kr. Denne motorkontrolleren er ikke vantett og må derfor plasseres inne i elektronikkhuset. På grunn av størrelsen ville elleve av disse også tatt opp mye plass. Det har tidligere år i tillegg vært et problem at motorkontrolleren henger seg opp, og trenger en restart for å fungere.

3.2.2 ThrustMe

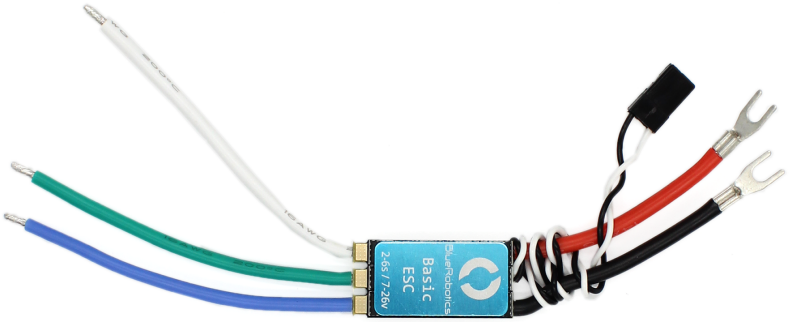
Da fjorårets prosjektgruppe fikk sponsoravtale med ThrustMe om thrustere, fikk de også med tilhørende motorkontrollere, vist i figur 3.2. Dette gjorde det aktuelt for oss å vurdere disse. Motorkontrollerene drives på 9-25 VDC og har et maksimalt strømtrekk på 40 A. De er 100×32 mm store, og veier 200 g [49]. Elleve motorkontrollere vil altså øke totalvekten på ROV-en betydelig. Den klart største fordelen med disse er at de er produsert vanntette. Dette åpner opp for å plassere dem utenfor elektronikkhuset, og kan begrense oppvarming. Å plassere dem utenfor elektronikkhuset betyr dog at det kreves flere kabelgjennomføringer, og kontakter for dette er kostbart. I tillegg er leveringstiden på disse like lang som for thrusterne fra ThrustMe, som er uheldig da prosjektet avhenger av rask fremdrift.



Figur 3.2: ESC fra ThrustMe. Bildet er hentet fra [49].

3.2.3 Blue Robotics – Basic ESC

Blue Robotics har også egen motorkontroller, og anbefaler å bruke denne sammen med thrusterne sine. ESC-en er vist i figur 3.3. Den drives på 7-26 VDC og har et maksimalt strømtrekk på 30 A. I størrelse er den 32 × 17.1 mm, og veier bare 16.3 g. Disse vil altså ikke øke totalvekten til ROV-en nevneverdig. Ulempen med denne motorkontrolleren er at den ikke er vanntett, og må derfor plasseres inne i elektronikkhuset. Dette vil gi et økt varmebidrag som må tas hensyn til. Da motorkontrolleren er liten i størrelse, vil det være mulig med god luftgjennomstrømning i elektronikkhuset. Dette vil kunne kompensere for noe av varmebidraget.



Figur 3.3: ESC fra Blue Robotics. Bildet er hentet fra [35].

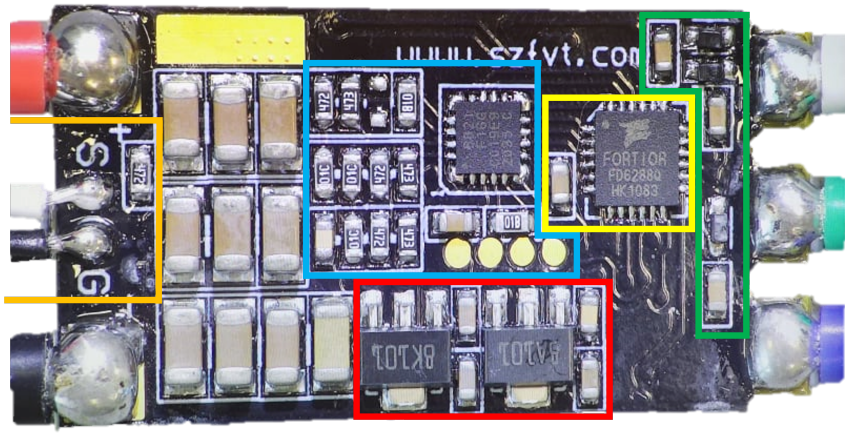
3.2.4 Konklusjon

Prosjektgruppen var samstemte om å ikke ta opp for stor plass av elektronikkhuset. Derfor tok vi tidlig avgjørelsen om å utelukke motorkontrolleren fra Benjamin Vedder. En vanntett motorkontroller er i utgangspunktet veldig godt egnet for en ROV, men på bakgrunn av vekt og leveringstid valgte vi også bort ThrustMe sin motorkontroller. Det er fordelaktig å bruke en motorkontroller som er laget for thrusterne vi har valgt. Derfor bestemte vi oss for motorkontrolleren fra Blue Robotics, selv om denne ikke er vanntett. Størrelsen på den gjør at den kan plasseres i elektronikkhuset uten å ta opp alt for mye plass. Motorkontrolleren er også kompatibel med motorene som skal brukes i manipulatoren.

3.3 Kretsanalyse av valgt motorkontroller

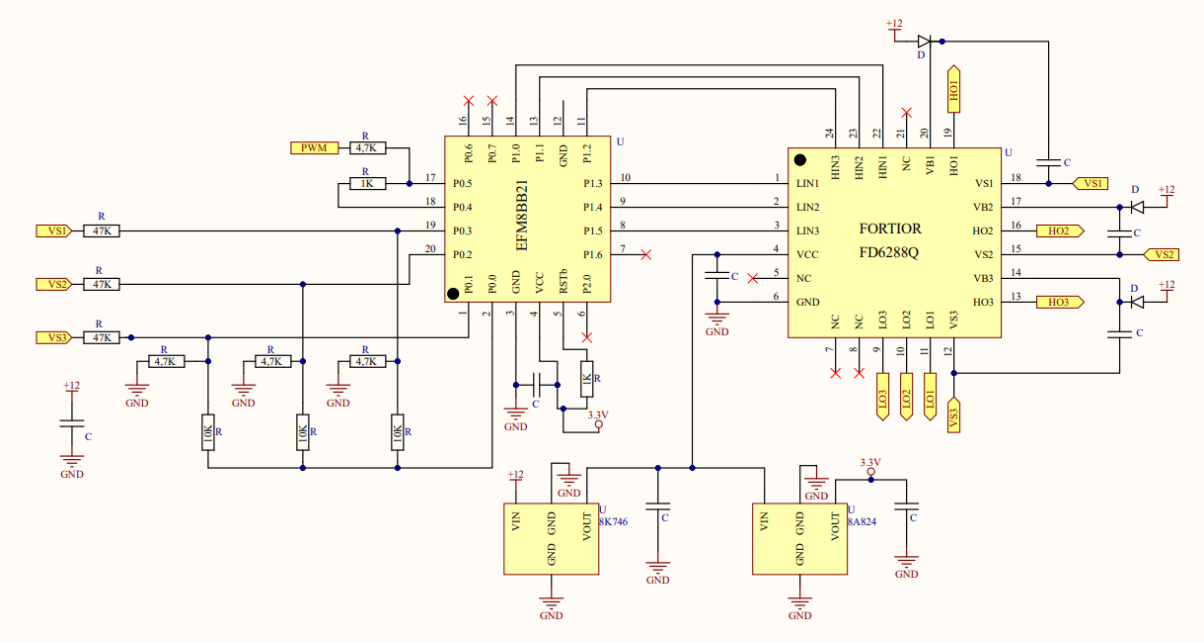
For å beskrive virkemåten til motorkontrolleren vi har valgt å bruke, ville relevant data og informasjon kommet godt med. Dessverre viste det seg vanskelig å få tilgang på dette. Vi etterspurte et datablad via e-post til Blue Robotics, men fikk i svar at produsenten av motorkontrolleren ikke oppgir spesifikk informasjon om oppbygning og virkemåte.

Motorkontrollerne fikk vi levert tidlig i prosjektet. Derfor var det mulig å legge ned en del arbeid i å kartlegge elektriske forbindelser for å finne ut hvordan signalene inn til motorene genereres. Ved bruk av multimeter kan man punktprøve hver enkelt pinne på hele ESC-en, og finne ut om det er elektrisk forbindelse uten motstand. Figur 3.4 viser et bilde av motorkontrolleren tatt gjennom mikroskopet vi brukte under punktprøving. De ulike delkretsene er rammet inn ulike farger, og vil bli nærmere forklart utover i delkapittelet.



Figur 3.4: Bildet er tatt med mikroskop. Oransje felt viser inngangen til PWM-signalet. Blått felt viser avleserkrets for posisjon, samt mikrokontroller med tilhørende testpunkt. Rødt felt viser to spenningsregulatorer med ladningsbøtter. Gult felt er MOSFET-driveren, og grønt felt viser dioder og kondensatorer for gate-spenningene til MOSFET-ene.

Fra de forbindelsene som ble funnet under punktprøving, tegnet vi også opp et kretsskjema som viser alle koblinger på kretsen. Det fullstendige kretsskjemaet er vist i figur 3.5. Dette vil komme godt med for å kunne forstå hvor de ulike signalene går når motorkontrolleren mottar et PWM-signal den skal behandle og gjøre om til et motorpådrag.



Figur 3.5: Kretsskjema som viser forbindelsen vi fant på motorkontrolleren. Komponenten *EFM8BB21* er mikrokontrolleren, og *FORTIOR FD6288Q* er MOSFET-driverkretsen. De to komponentene nederst i skjemaet markert med *8K746* og *8A824* er spenningsregulatorene. Koblingspunktene *VS1-3* går til fasene inn på motoren. *HO1-3* går til gate-terminalene på MOSFET-ene tilsvarende Q1-3 i figur 3.1, og *LO1-3* til gate på MOSFET-ene Q4-6

Videre i delkapittelet vil de innrammede delkretsene i figur 3.4 forklares nærmere basert på det vi tegnet opp i kretsskjemaet.

3.3.1 Spenningsregulatorer

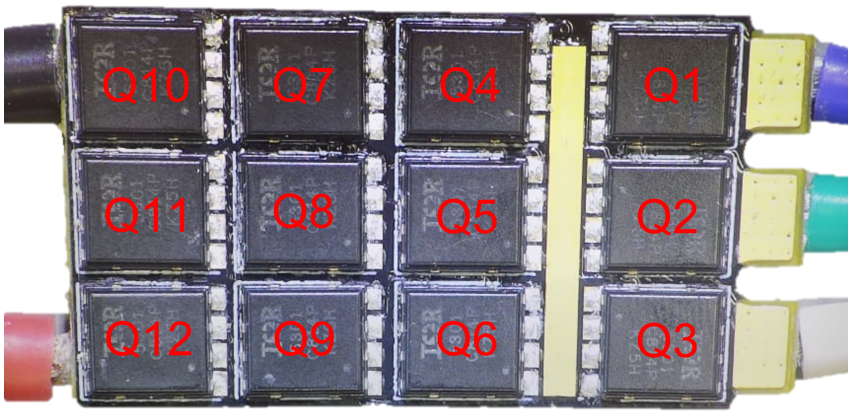
På motorkontrolleren er det to spenningsregulatorer. Disse er markert med ulike nummer, som antakeligvis kun har å gjøre med at de gir ut ulike spenningsnivå. Regulatorene i seg selv ser ut til å være enkle, og produsenten av motorkontrolleren har sannsynligvis brukt en type som er rimelig i pris. Dette er vanskelig derimot vanskelig å bekrefte, da vi ikke har funnet noe form for datablad på disse.

Den ene spenningsregulatoren er koblet til spenningstilførselen, som kan variere mellom 7 V og 26 V [35]. Regulatoren ble testet for ulike spenningsnivåer på inngangen, og ble målt til å gi ut 12 V når inngangen ble forsynt med ca. 15 V eller mer. Ettersom motorkontrolleren forsynes med 12 V, testet vi også regulatoren med inngangsspenning på dette nivået. På utgangen klarte den da bare å gi ut ca. 10.5 V. Denne regulatoren forsyner både den andre regulatoren og driverkretsen til MOSFET-ene. Den andre regulatoren målte vi til å gi ut 3.3 V, og denne forsyner mikrokontrolleren.

3.3.2 Transistornettverk

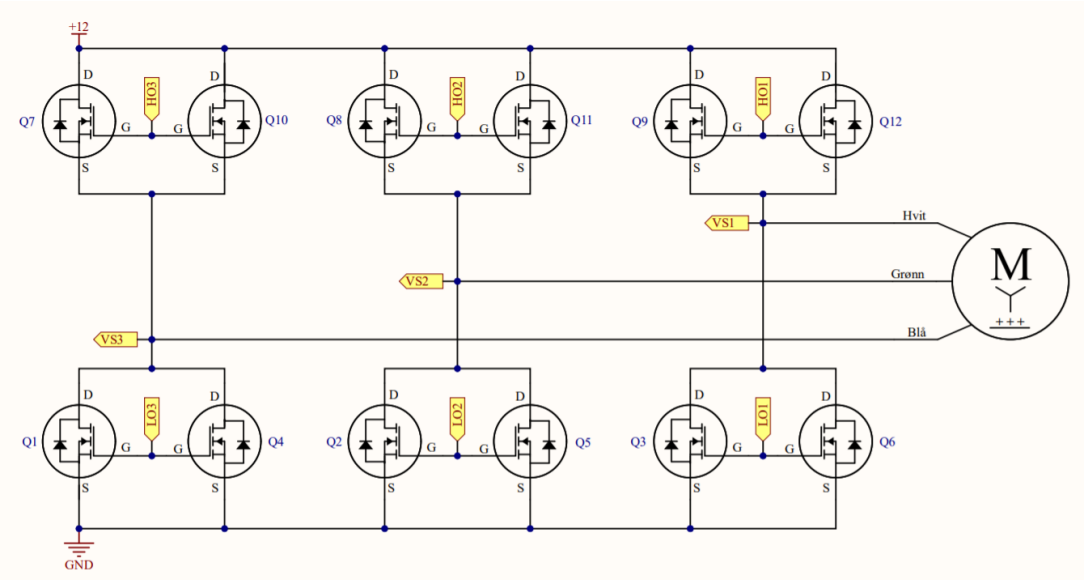
Som nevnt tidligere brukes MOSFET-er til å styre strømmen gjennom fasene koblet til motoren. Dette gjøres normalt ved å koble 6 MOSFET-er i en H-bro, slik som ble vist i figur 3.1. Dette

er en konfigurasjon som gjør at strømmen kan styres til å gå begge veier gjennom alle fasene, og dermed drive en 3-fase motor. På ESC-en fra Blue Robotics brukes det totalt 12 MOSFET-er, hvor to og to er koblet i parallell. Dette gjør at motstanden gjennom MOSFET-ene, og dermed effekttapet, halveres. I figur 3.6 vises MOSFET-ene på motorkontrolleren. Disse ligger på andre siden av motorkontrolleren i forhold til resten av delkretsene. MOSFET-ene er av typen IRFH5301 Power MOSFET fra IOR [19]. De tre fasene, som har utgang på høyre side av figuren, har distinkte farger, og disse fargene vil videre bli brukt til å skille de tre fasene fra hverandre.



Figur 3.6: Bilde av MOSFET-er på motorkontrolleren. Disse er vanligvis dekket av en blå aluminiumsplate. Faseutgangene til høyre har hver sin ledningsfarge. Derfor defineres de tre fasene heretter som blå, grønn og hvit fase.

Transistornettverket ble også punktprøvd, og figur 3.7 viser hvordan de 12 MOSFET-ene er koblet sammen. MOSFET-ene her er nummerert tilsvarende som i figur 3.6. De gule tilkoblingspunktene viser hvordan denne delen av kretsen er koblet til resten av motorkontrolleren. Den komplette skjemategningen ble vist i figur 3.5. Koblingspunktene HO1-HO3 er koblet til de høye utgangene på MOSFET-driverkretsen, mens portene LO1-LO3 er koblet til de lave utgangene på driverkretsen. Grunnen til at de 6 øverste MOSFET-ene har egne dedikerte høye utganger på driverkretsen blir forklart senere i kapittelet.



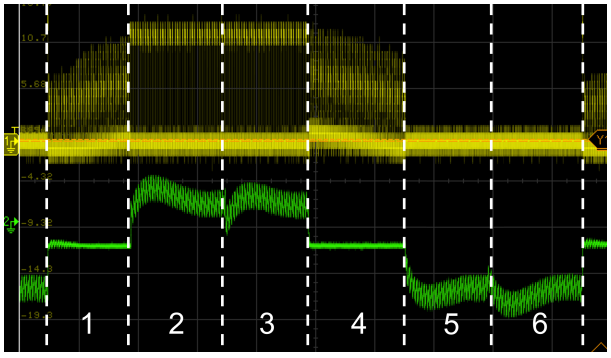
Figur 3.7: Kretsskjema som viser koblingen mellom alle MOSFET-ene på motorkontrolleren.

Motorkontrolleren styrer spenningen på motorens tre faser ved å aktivere to MOSFET-par om gangen. Dette gir seks forskjellige steg hvor hver fase er koblet til 12 V i to av stegene, koblet til jord i to og er «flytende» i de to siste. Ved å kommutere gjennom disse seks stegene vil rotoren rotere. Retningen på motorens rotasjon avhenger av hvilken retning kommuteringssekvensen eksekveres i, men rekkefølgen på stegene er alltid den samme. De seks stegene i kommuteringssekvensen, med tilhørende aktive MOSFET-er, er vist i tabell 3.1.

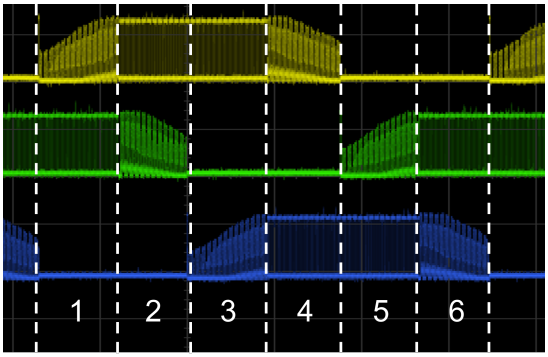
Tabell 3.1: Eksempel på en kommuteringssekvens av fasene.

Steg	Aktive MOSFET, Høy side	Aktive MOSFET, Lav side
1	Q2 og Q5	Q7 og Q10
2	Q2 og Q5	Q9 og Q12
3	Q1 og Q4	Q9 og Q12
4	Q1 og Q4	Q8 og Q11
5	Q3 og Q6	Q8 og Q11
6	Q3 og Q6	Q7 og Q10

Videre ble fasespenningene og -strømmene målt med oscilloskop når motorkontrolleren ble kjørt med en T200 thruster. I figur 3.8a vises spenningen mellom hvit fase og jord, og strømmen gjennom den samme fasen, under en fullstendig kommuteringssekvens. I figur 3.8b vises et skopbilde når alle tre faser ble målt samtidig. Den gule grafen er tilsvarende signal i begge de to figurene.



(a) Kommuteringssekvensen målt på én fase. Den gule grafen viser spenningen, og den grønne viser strømmen. Strømmen er målt med en probe som gir ut spenning basert på målt magetisme som følge av strømføring i fasen.



(b) Kommuteringssekvensen målt på alle tre fasene. Den gule grafen viser hvit fase, grønn graf viser blå fase og blå graf viser grønn fase.

Figur 3.8

I steg 1 er den hvite fasen flytende, som betyr at den ikke er direkte koblet til hverken inngangsspenningen eller jord. Spenningsmålingen man ser i dette steget kommer av at det induseres en strøm i spolen inne i motoren som tilhører denne fasen. Dette skjer når magnetene i rotoren beveges relativt til spolen. Spenningen som induseres kalles *BEMF*[4], og blir forklart mer detaljert når avleserkretsen for posisjon skal analyseres. I steg 2 er hvit fase koblet til 12 V, og grønn[5] fase til jord. Strømmen går da inn på den hvite fasen, og ut gjennom den grønne. I steg 3 er hvit fase fortsatt koblet til 12 V, men nå er blå fase koblet til jord. Strømmen går da samme vei gjennom den hvite fasen, men ut gjennom blå. I steg 4 er hvit fase igjen flytende. I steg 5 er hvit fase koblet til jord, og grønn fase til 12 V. Vi ser da at strømmen går i motsatt retning

[4] **B**ack **e**lectro**m**otive **f**orce

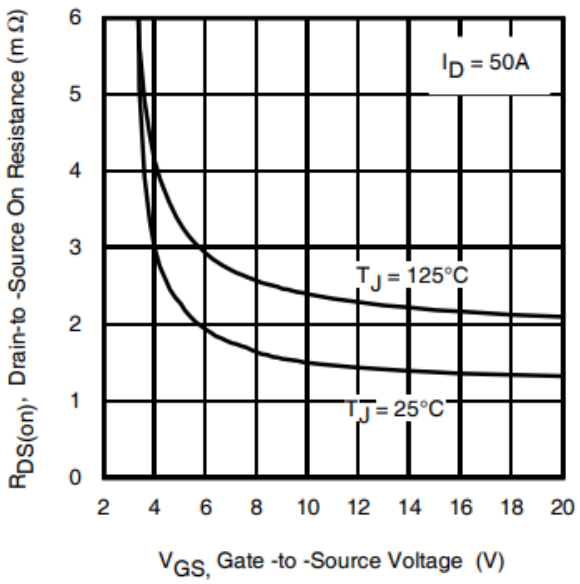
[5] Merk at grønn graf viser blå fase, og blå graf viser grønn fase.

gjennom den hvite fasen. I steg 6 er hvit fortsatt koblet til jord, og nå er blå koblet til 12 V. De seks stegene viser at hver fase kan føre strøm i ulike retninger, som videre får motoren til å rotere.

3.3.3 Gate-driver

Mikrokontrolleren på motorkontrolleren er i seg selv ikke i stand til å drive MOSFET-ene, og disse er derfor koblet til en driverkrets som styres av mikrokontrolleren.

MOSFET-ene brukt på motorkontrolleren er av N-type, som betyr at de krever en positiv spenning mellom gate og *source*, V_{GS}. For mange MOSFET-er kreves en V_{GS} på 10 V eller høyere for at de skal kunne aktiveres. Ved høyere V_{GS} vil motstanden i MOSFET-en bli lavere. I databladet til MOSFET-en vises forholdet mellom V_{GS} og motstand i figur 12, side 5 [19]. Denne er gjengitt i figur 3.9 under. Spenningene som kreves for å aktivere MOSFET-en er langt over 3.3 V som mikrokontrolleren kan forsyne. Derfor brukes den ekstra driverkretsen.



Figur 3.9: Forholdet mellom V_{GS} og motstand. Hentet fra [19].

På grunn av intern kapasitans mellom gate, source og *drain* kreves det en vesentlig ladning for å drive gate til riktig spenning. Ifølge figur 6, side 3, i databladet til MOSFET-en, kreves det rundt 80 nC for å drive gate-terminalen til 10 V [19]. Ettersom to MOSFET-er er koblet i parallell, kreves det derfor 160 nC for hvert par. For å minimere effekttapet må denne oppladningen skje fortest mulig, slik at motstanden gjennom MOSFET-en ikke forblir høy unødvendig lenge. Det å forsyne 160 nC veldig raskt, fører til store strømmer som ofte er større en hva mikrokontrolleren kan drive uten å ta skade.

Mikrokontrolleren har en grense på maksimalt 100 mA som kan bli tilført eller trukket fra én av I/O-pinnene [40]. Det er også oppgitt en grense på 200 mA for hele mikrokontrolleren. Mikrokontrolleren vil derfor bare kunne drive to I/O pinner dersom begge trekker 100 mA. Ettersom strøm tilsvarer endring i ladning over tid, kan man finne den raskeste oppladningstiden mikrokontrolleren kan gjennomføre fra en I/O-pinne:

$$t = \frac{Q}{I} = \frac{160 \cdot 10^{-9}}{100 \cdot 10^{-3}} = 1.6\ \mu s$$

Selv om dette er relativt raskt, vil uansett ikke mikrokontrolleren kunne tilføre mer enn 3.3 V. Det står også i databladet at en slik strøm over lengre tid vil kunne påvirke påliteligheten til mikrokontrolleren [40].

Bruken av en driverkrets gjør at disse problemene løses. Det brukes derfor en krets som er laget spesifikt for å drive gate-terminalene til MOSFET-er. Driveren på motorkontrolleren er en FD6288Q produsert av Fortior Tech [11]. Denne driveren blir forsynt med 12 V fra den ene spenningsregulatoren på motorkontrolleren, og sørger dermed for at gate-spenningene blir høye nok til å aktivere tilhørende MOSFET.

3.3.4 Bootstrapping

Som nevnt tidligere er de seks øverste MOSFET-ene koblet til egne utganger på driverkretsen, og dette kan forklares. Disse er også av N-type, og krever positiv V_{GS}. Samtidig er source koblet til en fase, og ikke til jord. Dette betyr at spenningen på source er den samme som spenningen over lasten, som i dette tilfellet er motoren. Da denne spenningen kan være minst like høy som spenningen som tilføres gate-terminalen, kreves det ekstra elektronikk for å sørge for at V_{GS} er stor nok til å aktivere MOSFET-en.

Det å sørge for en stor nok V_{GS} løses ved å koble en kondensator mellom source og inngangen VB på driverkretsen. Inngangen VB er videre koblet til 12 V via en diode. Dette betyr at når den tilhørende fasen, og dermed source-terminalen, er koblet til jord, vil kondensatoren lades opp til 12 V. Når de høye MOSFET-ene skal skrus på, vil intern logikk i driveren koble spenningen på VB, som nå er 12 V, sammen med forsyningsspenningen til driveren inn på gate-terminalen. På denne måten sikrer man at V_{GS} er positiv og stor nok til å skru MOSFET-en på, uavhengig av hva spenningen over lasten er. Dioden sørger for at dette høye spenningsnivået ikke påvirker spenningstilførselen på 12 V.

I kretsen, vist i figur 3.5, ser man på grunn av dette 3 par med kondensatorer og dioder koblet til driverkretsen. Denne oppkoblingen kalles vanligvis for *bootstrapping*. Pinnen VB på gate-driveren er altså en *V-bootstrap*.

Prinsippet fungerer kun ettersom MOSFET-ene skrus på periodisk. Kondensatorene blir dermed hele tiden ladet opp, og kan forsterke spenningsnivået på gate-terminalen til de høyre MOSFET-ene. Bootstrapping brukes ikke på de lave MOSFET-ene da source på disse er koblet til jord. Ved å sende 12 V inn på gate-terminalen her, vil V_{GS} uansett være stor nok.

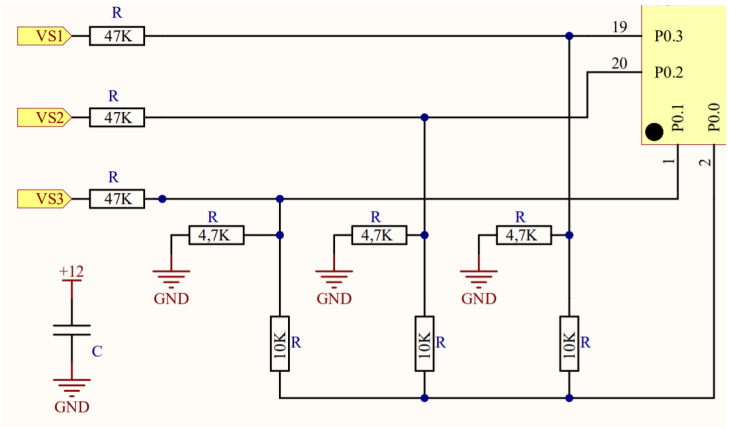
3.3.5 Posisjonsavlesning

Teori om posisjonsavlesning er hentet fra artikkelen *3-Phase Sensorless BLDC Motor Control Kit with S32K144* [39].

Som nevnt i delkapittel 3.1, er motorkontrolleren avhengig av å vite posisjonen til rotoren for å kunne vite når den skal kommutere til neste steg i sekvensen. Flyten mellom stegene er essensiell

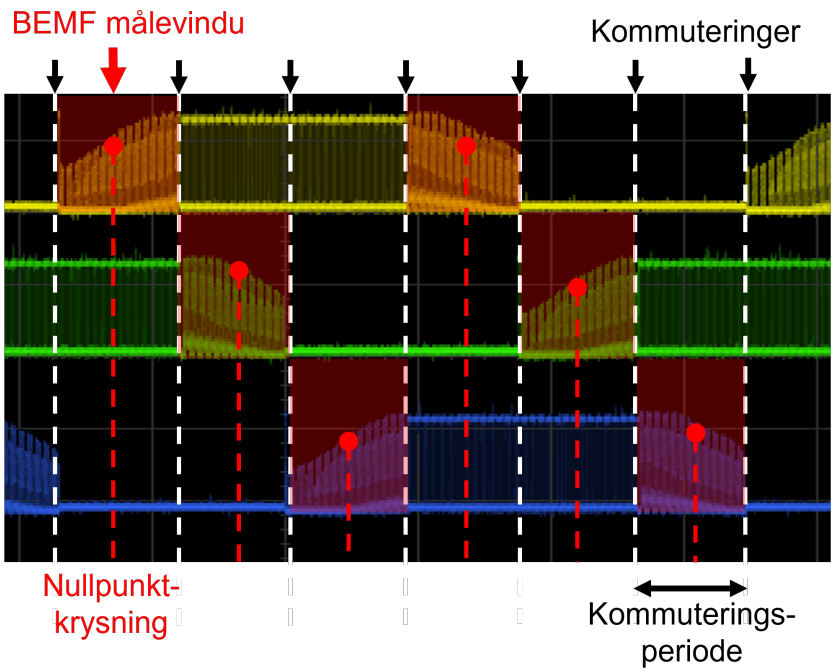
for at motoren skal kunne rotere jevnt. Det finnes flere ulike metoder for posisjonsavlesing av en motor. Et vanlig prinsipp er å måle posisjonen på rotoren ved å bruke sensorer. Dette kan eksempelvis være Hall-sensorer, eller en enkoder. Disse vil kunne detektere et gitt antall punkter per rotasjon, avhengig av hvor mange sensorer som er brukt, og bruke punktene til å beregne rotorens omtrentlige posisjon. Vi vil ikke gå mer i detalj på prinsippet med bruk av sensorer, da vår motorkontroller baserer seg på sensorløs posisjonsavlesning.

I figur 3.8b kan man se at i hvert steg av kommuteringssekvensen er én av fasene frakoblet den sluttede kretsen fra inngangsspenning til jord, og omtales da som flytende. Likevel viser oscilloskopet spenningsavlesning. Dette kommer av den tidligere nevnte BEMF-en. Ifølge Faradays induksjonslov vil et varierende magnetfelt indusere en elektromotorisk spenning på en elektrisk ledende krets [54]. I motoren vår vil permanentmagnetene på rotoren forårsake et roterende magnetfelt, som passerer statorspolene i den frakoblede fasen. Det induseres en BEMF-spenning, som måles av motorkontrolleren via en avleserkrets. Figur 3.10 viser et utklipp fra det fullstendige kretsskjemaet, som kun viser avleserkretsen.



Figur 3.10: Utklipp fra kretsskjema vi har tegnet. Utklippet viser avleserkretsen, som forøvrig er markert med blått omriss i figur 3.4. Koblingspunktene VS1, VS2 og VS3 kommer henholdsvis fra hvit, grønn og blå fase. Kretsen er koblet til de fire pinnene P0.0-3 på mikrokontrolleren.

BEMF-en leses av i mikrokontrolleren ved at hver fase overvåkes, og sammenlignes med et felles referansepunkt. På pinne P0.1-3 leses spenningen fra hver fase av mikrokontrolleren. Spenningen er nedskalert av en spenningsdeler bestående av en 47 kΩ, og en 4.7 kΩ motstand. Hver fase er også tilkoblet pinne P0.0 via en 10 kΩ motstand. Ved å lese av spenningen på P0.0, får mikrokontrolleren et felles referansepunkt å sammenligne fasespenningene med. Prinsippet som brukes for å vite når det skal kommuteres, er deteksjon av nullpunktkrysning. Dette er illustrert i figuren 3.11 under.



Figur 3.11: Figuren viser hvor nullpunktkrysningen skjer når fasene er i flytende tilstand.

Nullpunktkrysning refererer til at BEMF-spenningen passerer «nullpunktet» enten stigende eller synkende. Grunnen til at fasene er samlet til et felles referansepunkt, eller nullpunkt, er at dette ikke nødvendigvis er ved 0 V. Mikrokontrolleren sammenligner fasen i flytende tilstand med referansepunktet ved hjelp av en analog komparator. Når spenningsnivået på den flytende fasen passerer spenningsnivået til referansepunktet, vil komparatoren gi et signal til kontrolleren. Når motorkontrolleren kommuterer, startes en *timer*. Signalet fra den analoge komparatoren forteller mikrokontrolleren at timeren har nådd halvveis til neste steg i kommuteringen. Når timeren teller ferdig andre halvdel av steget ferdig, kommuterer fasene, og hele prosessen gjentas for neste steg i sekvensen.

3.3.6 Styring av motorkontroller

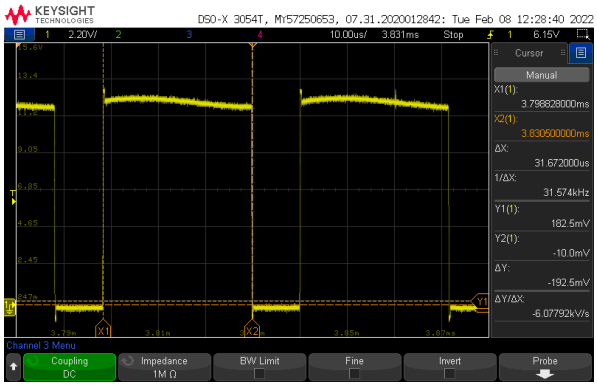
Det finnes flere protokoller for kommunikasjon med en motorkontroller. De fleste av disse baserer seg på et PWM-signal med varierende pulsbredde. Utviklingen av nyere protokoller har vært drevet av behovet for raskere oppdatering av pådraget til motoren. For eksempel har behovet for raske oppdateringsfrekvenser i droner vokst, da bruken av PID-regulatorer for stabilitetskontroll har økt. Den nyeste utviklingen av kommunikasjonsprotokoller for motorkontrollere er en digital protokoll kalt *Dshot* [37]. Denne vil ikke bli videre forklart, da den ikke brukes på ESC-en fra Blue Robotics.

Basic ESC bruker en vanlig, men modifisert, versjon av den eldste blant kommunikasjonsprotokollene. Den maksimale oppdateringsfrekvensen, frekvensen på PWM-signalet inn til motorkontrolleren, er oppgitt til å kunne være 400 Hz. I *Blue Robotics* sine eksempler brukes allikevel et PWM-signal på 50 Hz, da dette er mer enn raskt nok. For å kontrollere pådraget til motoren varierer pulsbredden på signalet mellom 1100 μs og 1900 μs. 1100 μs pulsbredde tilsvarer maksimalt pådrag i revers, 1900 μs tilsvarer maksimalt pådrag fremover og 1500 μs tilsvarer null pådrag.

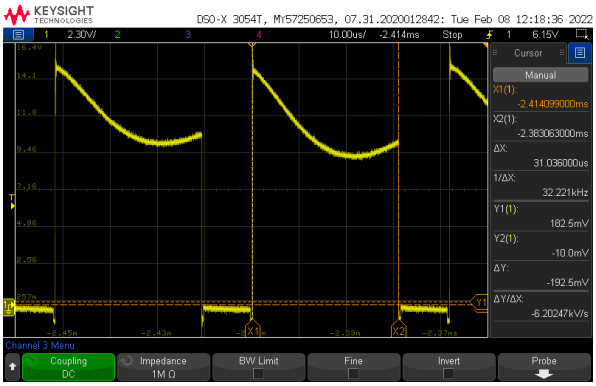
For å forstå hvordan ESC-en genererer motorpådrag, må man observere pådragssignalene når motoren kjører med ulik last. Det finnes flere måter en motorkontroller kan regulere motorpådraget på. Eksempelvis kan pådraget reguleres gjennom avlesing av strømtrekk, som er proporsjonalt med dreiemomentet til motoren, vist i ligning 2.3 fra kapittel 2. Motorkontrolleren regulerer da pådraget for å sørge for et konstant dreiemoment ved en gitt pulsbredde på PWM-signalet. En annen vanlig måte å regulere pådrag på, er direkte hastighetsregulering. Dette gjør motorkontrolleren basert på posisjonsavlesningen av rotoren. Videre vil vi se på hvordan pådragsreguleringen skjer i motorkontrolleren som brukes i ROV-en.

For å regulere pådraget til den tilkoblede motoren, bruker motorkontrolleren et eget maskinvaregenerert PWM-signal som svitsjer MOSFET-ene av og på. Dette kan observeres i figur 3.8b. Når fasene er koblet til 12 V, ser man at spenningssignalet ikke er konstant høyt, men svitsjer opp og ned med en frekvens som er mye raskere enn selve kommuteringsfrekvensen. Videre er det viktig å skille mellom PWM-signalet som går inn på motorkontrolleren, heretter omtalt som styresignalet, og PWM-signalet som blir maskinvaregenerert og sendt til MOSFET-ene. PWM-signalet som svitsjer MOSFET-ene er målt med oscilloskop til å har en fast periode på omtrent 41.7 μs, uavhengig av styresignalet som sendes inn til motorkontrolleren. Pulsbredden til styresignalet styrer bare driftsyklusen til PWM-signalet inn på MOSFET-ene, og ikke perioden på signalet. Denne svitsjingen skjer uavhengig av kommuteringssekvensen, og endrer kun den gjennomsnittlige påtiden til hver MOSFET. Ved lav driftssyklus krever derfor motoren flere svitsjeperioder fra MOSFET-ene for å rotere langt nok til fullføre et steg i kommuteringssekvensen.

For å analysere oppførselen til pådragssignalene, ble en av thrusterene kjørt i luft, som har minimal motstand, og videre i vann, som har betraktelig høyere motstand. Driftsyklusen til det maskinvaregenererte PWM-signalet forblir den samme i begge tilfeller. Signalet ble også målt med ulik pulsbredde på styresignalet inn på motorkontrolleren. Ved å justere pulsbredden på styresignalet mellom 1100 μs og 1900 μs, ble det verifisert at driftsyklusen på det maskinvaregenererte PWM-signalet varierer mellom 0 og 100 %. Figur 3.12 viser spenningen på den hvite fasen i ett av kommuteringsstegene hvor den er koblet til 12 V. I skopet er signalet forstørret, slik at man kun ser én periode av det maskinvaregenererte PWM-signalet. Pulsbredden på inngangssignalet var i denne gjennomføringen 1800 μs, og de to delfigurene viser forskjellene mellom kjøring i luft og vann.



(a) Påtid til maskinvaregenerert PWM-signal målt på hvit fase. Her har styresignalet en pulsbredde på 1800 μs, og thrusteren kjøres i luft. Påtiden er her 31.67 μs.



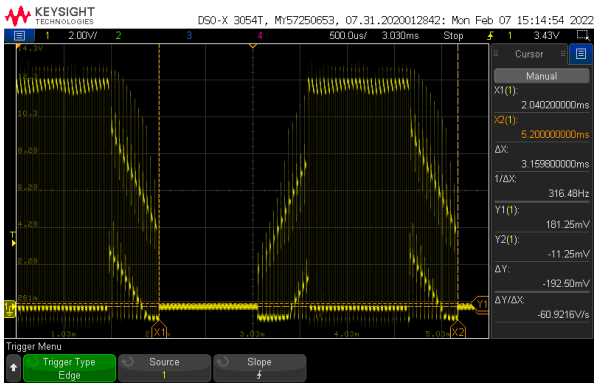
(b) Påtid til maskinvaregenerert PWM-signal målt på hvit fase. Her har styresignalet en pulsbredde på 1800 μs, og thrusteren kjøres i vann. Påtiden er her 31.04 μs.

Figur 3.12

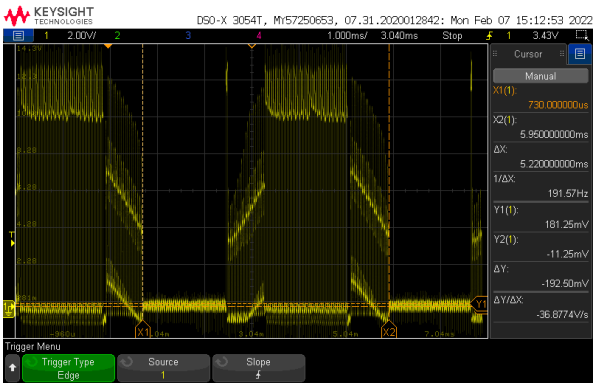
Selv om pulsbredden er lik i begge situasjonene, ser man større induktiv effekt når thrusteren

kjøres i vann. Dette kan observeres ved å se på formen av signalpulsene i de to tilfellene. Grunnen til forskjellene er antakeligvis at vann oppleves av propellene som en høyere last, og systemet trekker mer strøm. Ved høyere strømtrekk vil induktansen i motorspolene motsette seg den store strømendringen, og dette vises igjen på skopet som et større spenningsfall i starten av hver puls. Ettersom strømmen etterhvert slipper gjennom, faller spenningen mot samme nivå som pulsene ligger på ved kjøring i luft.

Måler man derimot hvor lang tid det tar å gå gjennom de seks stegene i kommuteringssekvensen, kan man observere at dette tar mye lenger tid i vann. Figur 3.13 viser spenningsmålingene fra oscilloskopet av en hel kommuteringssekvens på den hvite fasen ved kjøring i både luft og vann.



(a) Kommutering gjennom en hel sekvens målt på hvit fase med en pulsbredde på 1700 μs i styresignalet. Her kjøres motoren i luft.



(b) Kommutering gjennom en hel sekvens målt på hvit fase med en pulsbredde på 1700 μs i styresignalet. Her kjøres motoren i vann.

Figur 3.13

Av figur 3.13a kan man se at de seks stegene tar 3.16 ms når thrusteren roterer fritt i luft. De samme stegene tar, som vist i figur 3.13b, 5.22 ms når thrusteren er under vann. Ettersom kommuteringen skjer tregere i vann, er også rotasjonshastigheten til motoren lavere, selv med samme pulsbredde på styresignalet. Det ble også verifisert at motoren trakk mer strøm under kjøring i vann.

Det at driftsyklusen på PWM-signalet til MOSFET-ene ikke endres når lasten på motoren økes, og strømtrekket går opp, betyr at motorkontrolleren ikke regulerer på dreiemoment. Videre verifiserte vi at kommuteringssekvensen tar lenger tid ved kjøring i vann enn i luft, noe som indikerer at motorkontrolleren heller ikke regulerer på rotasjonshastighet. Motorkontrolleren endrer kun driftsyklusen til det maskinvaregenererte PWM-signalet når styresignalet endres, og det viser seg at hastigheten til motoren er avhengig av lasten den utsettes for. Sendes et styresignal med pulsbredde på 1500 μs, tilsvarer dette 0% driftsyklus på PWM-signalet til MOSFET-ene. Styresignal med pulsbredde på 1900 eller 1100 μs tilsvarer 100% driftsyklus, men hvor sekvensen til kommuteringen utføres i forskjellig retning. Motorkontrolleren er altså kun en styringsenhet som justerer det prosentvise pådraget til thrusteren, uten å drive regulering basert på motorens respons.

3.4 Konklusjon

I dette kapittelet har vi presentert den generelle virkemåten til motorkontrollere av typen ESC for børsteløse motorer. Det ble valgt å bruke motorkontrolleren Basic ESC fra *Blue Robotics* for alle thrustere og motorer til manipulator. Virkemåten til motorkontrolleren ble analysert for å få et innblikk i hvordan styresignalet vi sender til motorkontrolleren blir behandlet, og omformet til et pådragssignal motoren kan drives på. Gjennom analysene ble de ulike delkretsene kartlagt og beskrevet. Videre fant vi ut at motorkontrolleren generer et pådragssignal kun basert på pulsbredden til styresignalet på inngangen, og den driver ingen intern regulering. På grunn av fremdriften i prosjektet, ble videre analyser av motorkontrolleren nedprioritert. Med mer tid ville det vært nyttig å sette seg inn i programvaren på mikrokontrolleren for å få en grundigere forståelse av hvordan signalbehandlingen på motorkontrolleren foregår. Likevel har analysen gitt oss en god forståelse av hvordan styresignalet omformes til et motorpådrag, og dette vil gi oss gode inngangsverdier før vi skal starte utviklingen av styresystemet.

Kapittel 4

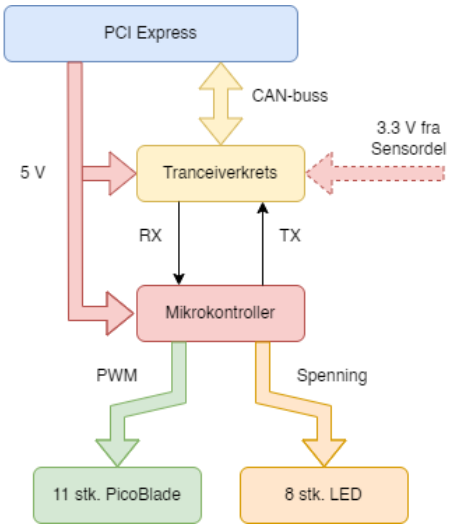
Maskinvare

Kapitteloversikt

I dette kapittelet presenteres den resterende maskinvaren, utover motorer og motorkontrollere. For at motorer og motorkontrollere skal kunne fungere med resten av systemet, skal det derfor utvikles et kretskort som bindeledd. Kretskortet vil da fungere som et grensesnitt mellom motorkontrollere, mikrokontroller og CAN-bussen for kommunikasjon med de andre delsystemene av ROV-en. I tillegg er det ønskelig å ha et ryddig og oversiktlig elektronikkhus. Derfor vil kapittelet også omhandle motorkontrollerenes plassering inne i elektronikkhuset. Fokuset ligger på å gjøre vedlikehold enkelt, og la luft bevege seg rundt slik at varme komponenter kjøles ned. God planlegging av maskinvaren vil gjøre arbeidet med systemet vesentlig enklere utover i prosessen.

4.1 Kretskort

Som et sentralt bindeledd i systemet har det blitt designet et kretskort for å gjøre kommunikasjonen mellom de forskjellige komponentene mulig. Kretskortet realiserer koblingen mellom mikrokontroller og motorkontrollere. Det er viktig å få en pålitelig og god kobling, som er enkel å koble fra i tilfelle vedlikehold skulle bli nødvendig. Kretskortet realiserer også kommunikasjonen med resten av systemet slik at mikrokontrolleren kan motta styresignaler og sensordata. Et flytskjema over kretskortet og hovedkomponentene er vist i figur 4.1.

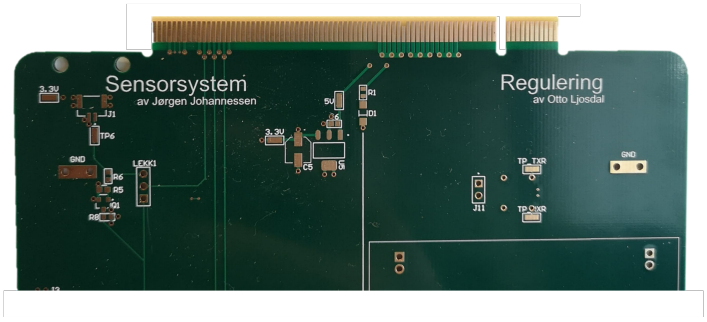


Figur 4.1: Flyskjema som viser de ulike delene av kretskortet.

Formen og størrelsen på kretskortet er bestemt av bildebehandlingsgruppen slik at alle kortene lett kan monteres sammen i elektronikkhuset. Størrelsen på de enkelte kortene er derfor bestemt ut fra det kortet som krever størst areal. Det ble derfor tidlig bestemt at sensorgruppen og reguleringsgruppen skulle lage et felles kretskort, ettersom ingen av gruppene krevde over halvparten av det tilgjengelige arealet som var tildelt. Dette gir bedre plass i elektronikkhuset, bedre luftgjennomstrømming og dermed bedre kjøling. Delkapittelet vil videre forklare de ulike delene av kretskortet presentert i flytskjemaet.

PCI Express

For å koble de ulike kretskortene i ROV-en sammen, vil det i fronten av elektronikkhuset monteres et vertikalt kretskort med kontakter av typen *PCI Express*. For å koble kretskortet til disse kontaktene må det designes med såkalte gullfingre, vist i figur 4.2. Dette betyr at kretskortet har eksponerte padder på enden, som kan være gull-platede. Kretskortet utgjør dermed en av kontaktene i seg selv.



Figur 4.2: Gullfingre på kretskortet.

Kretskortet med PCI Express-kontaktene er laget av bachelorgruppen med ansvar for bildebehandling, og utgjør en multibuss for kommunikasjon og strømforsyning. Over denne går kommunikasjonen på CAN-buss, 5 V forsyning til mikrokontrolleren vår, andre spenningsnivå og signaler til andre grupper. Denne løsningen med en felles buss gjør at mengden løse kabler i elektronikkhuset reduseres betydelig, og luftflyten blir dermed mye bedre. I tillegg blir elektro-

nikken ryddig, oversiktlig og lett å vedlikeholde.

Mikrokontroller

For å kunne regulere basert på sensordata, motta styresignaler og sende pådrag til motorkontrollerene, er det nødvendig med en mikrokontroller. Det var et ønske å ikke designe en egen krets for en mikrokontroller, men bruke et ferdig utviklingskort med mikrokontroller og tilhørende elektronikk. Ettersom det er en pågående global mangel på integrerte kretser, ble det bestemt å ta i bruk et av utviklingskortene som var tilgjengelige på universitetet.

Valget falt på kortet STM32F4DISCOVERY [43]. Dette har en STM32F407 mikrokontroller med en 32-bit Arm Cortex-M4 mikroprosessor. Denne mikroprosessoren har en flyttallsprosessor [44] som er essensielt ettersom mikrokontrolleren skal utføre reguleringen, og dermed mange matematiske utregningner som involverer flyttall.

Kortet STM32F4DISCOVERY er laget for prototyping, og er designet for å enkelt kunne kobles til eksterne kretser. Ettersom kortet er laget for å kunne håndteres uten stor forsiktighet, er det ikke nødvendig med ekstern ESD[1]-beskyttelse. Ifølge tabell 45, side 112 i databladet, har mikrokontrolleren STM32F407 en ESD-beskyttelse på 2000 V.

CAN-busstranceiver

For å kommunisere med resten av ROV-en brukes kommunikasjonsprotokollen CAN-buss. Transceiveren sender og mottar seriellkommunikasjon over et differensialpar. Sensorkortet vil kunne sende sensordata, mini-PC-en vil kunne sende styresignal, og fra kraftforsyningsmodulen vil det kunne sendes informasjon om strømtrekk. Det blir også mulig å sende informasjon om hvilket pådrag som er gitt hver thruster tilbake til operatørgrensesnittet. For å realisere kommunikasjonen over CAN-buss brukes en tranceiver-krets, TJA1044GT/3Z. Denne kretsen har en en ESD-beskyttelse på 8000 V ifølge side 2 i databladet[31].

Transceiveren kobles til mikrokontrolleren med to serielle linjer. RX, for å motta, er koblet til pinne PD0 og TX, for å sende, er koblet til pinne PD1 på mikrokontrolleren. Tranceiveren er koblet til andre enheter via et differensialpar, CANH og CANL, som begge går over PCI-kontakten. Fordi kommunikasjonen går over et differensialpar, er signalene bedre beskyttet mot støy, ettersom ekstern påvirkning vil gjøre utslag på begge differensiallinjene i tilnærmet like stor grad. Tranceiveren er forsynt med 5 V, og har en egen inngang V_{IO} som bestemmer spenningsnivået på differensiallinjene. Denne er koblet til 3.3 V fra regulatoren på sensordelen av kretskortet. På denne måten sikrer man at alle enhetene opererer differensiallinjene på samme spenningsnivå.

For å begrense effekten av at signalet reflekteres når det når enden av det fysiske nettverket, er det anbefalt at hver ende termineres med en 120 Ω motstand [51]. Målet er at termineringen har samme impedans som overføringslinjene [42]. Ettersom det er ønskelig å kunne teste kommunikasjonen mellom kortene uten at alle kortene er tilstede, er det lagt til rette for at det med en *jumper* kan kobles inn en 120 Ω motstand mellom CANH og CANL ved tranceiveren. Dette tillater de forskjellige kortene å kobles etter behov, og termineringen kan kobles til og fra på kortet.

[1] **E**lectro **S**tatic **D**ischarge

PWM-signal

Et viktig formål med kretskortet er å koble mikrokontrolleren til motorkontrollerene. Mikrokontrolleren skal produsere et PWM-signal for hver motorkontroller for å styre hastigheten til motorene. Det må derfor settes opp koblingspunkter for disse PWM-signalene. Motorkontrollerene vil ikke være montert på kretskortet, noe som er grunngitt i kapittel 4.2, og det er derfor viktig at mototorkontrollerene kan kobles fra kretskortet slik at elektronikkhuset kan demonteres.

For å realisere tilkoblingen er det brukt *PicoBlade* plugger fra MOLEX [30]. Hanpluggen er overflatemontert, og loddes direkte på kretskortet. På ledningen fra motorkontrolleren festes det krympesko. Krympeskoene monteres i et hus som passer i hanpluggen.

Pluggene er oppgitt til å tåle en strøm på 1 A. Spenningsnivået fra mikrokontrolleren er 3.3 V, og strømmen er begrenset av en 4.7 kΩ motstand på motorkontrolleren. Disse pluggene har derfor mer en god nok kapasitet.

LED

For å kunne vise hvilke thrustere som er aktive under testing, er det brukt 8 rav-fargede WL-SMCW LED [8]. Disse LED-ene er plassert slik som thusterene er plassert på ROV-en, og gir et oversiktlig bilde av hvilke thrustere som er aktivert. Hver av LED-ene er koblet til en egen utgang på mikrokontrolleren, og kan dermed styres individuelt. For å begrense hvor mye strøm som går gjennom LED-en er det plassert en 330 Ω motstand i serie med hver LED. Ettersom foroverspenningen er typisk 2 V, og spenningen mikrokontrolleren gir er 3.3 V, blir strømmen gjennom LED-en tilnærmet:

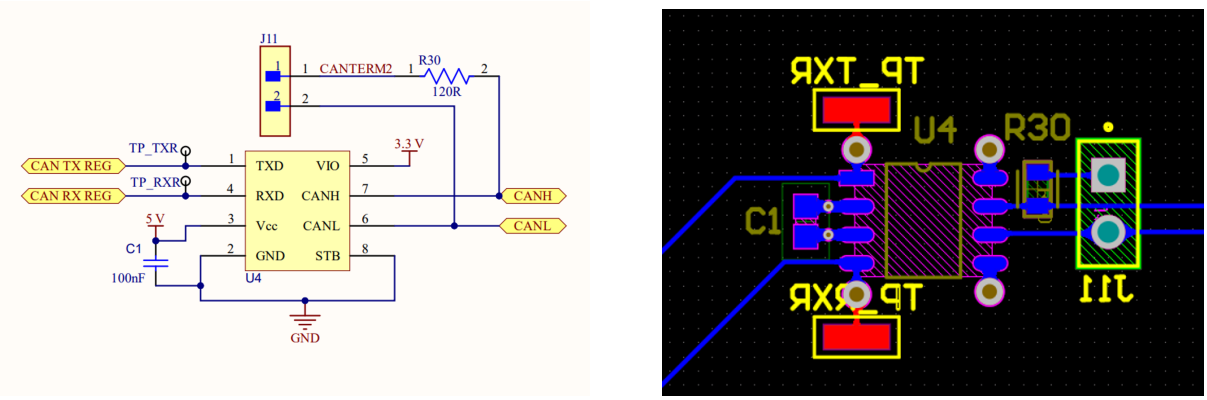
$$I = \frac{3.3\ V - 2\ V}{330\ \Omega} = 3.9\ mA \tag{4.1}$$

Dette er en tilnærming ettersom foroverspenningen avhenger av strømmen, men det gir likevel en god indikasjon på strømtrekket. Den tilnærmede strømmen er langt under den oppgitte maksimale kontinuerlige foroverstrømmen på 30 mA for LED-en. Dette er også under grensen på 25 mA for I/O pinnene på mikrokontrolleren, gitt på side 79 i databladet [44].

4.1.1 Skjemategning

For å produsere et kretskort, må det først lages en digital versjon av kortet slik at det kan genereres filer som kan sendes til en produsent. Vi har valgt å bruke *Altium designer* for å lage vårt kretskort.

Første steg i utviklingen av kretskortet er å lage en skjemategning. I denne skjemategningen defineres alle tilkoblingene som skal gjøres på kretskortet. Som et eksempel, er det vist et utsnitt av skjemategnignen for CAN-buss kretsen i figur 4.3a. I tillegg skal det lages et utlegg av kretsen. Utlegget er en digital versjon av kretskortet hvor man bestemmer plasseringen av komponentene og banene som forbinder disse. Et utsnitt av utlegget for CAN-buss kretsen er vist i figur 4.3b.



(a) Utsnitt av skjemtegningen for CAN-buss kretsen.

(b) Utsnitt av utlegget for CAN-buss kretsen.

Figur 4.3

Skjemategningen gir et oversiktlig bilde av hvordan kretsen er ment å fungere, og gjør prosessen med å lage et utlegg enklere. Når utlegget skal lages, bruker Altium informasjonen gitt i skjemategningen til å vise grafisk hvordan de ulike koblingene skal realiseres. Dette gjør at det er lettere å plassere komponenter på en måte der baner ikke kommer i konflikt med hverandre. Altium har flere verktøy for å automatisk generere utlegget fra skjemategningen, men for vårt kretskort er det mer hensiktsmessig å sette opp utlegget manuelt. Dette er fordi vi ønsker å holde sensordelen og reguleringsdelen av kretskortet adskilt, og unngå at baner med svitsjende signaler blir plassert i nærheten av sensorelektronikken.

Komponent-bibliotek

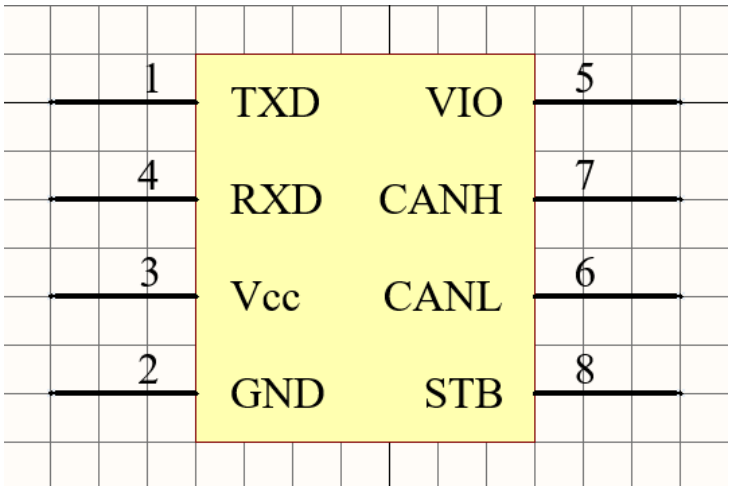
For å lage en digital skjemategning i Altium, er det behov for et komponent-bibliotek som inneholder alle komponentene som skal brukes på kretskortet. Komponentbiblioteket består av to egne bibliotek, hvor det ene inneholder symbolet brukt i skjemategningen, og det andre innholder fotavtrykket til komponentene. Disse to bibliotekene er knyttet sammen slik at Altium vet hvilket fotavtrykk som tilhører hvilket symbol.

I mange tilfeller kan man laste ned ferdiglagede komponenter som innholder symbol og fotavtrykk. Altium har en egen søkemotor hvor komponenter fra mange vanlige produsenter er tilgjengelige. Dette gjør det enkelt å komme raskt i gang med skjemategningen. Noen av komponentene, slik som mikrokontrolleren, er tilgjengelige på nettsteder hvor andre som har laget fotavtrykket før, har lastet dem opp. Komponenter som ikke er tilgjengelige ferdiglaget må lages selv. Altium har et eget verktøy som gjør dette relativt enkelt å gjennomføre.

For å kunne se hvordan plasseringen av komponenter passer i forhold til andre komponenter i elektronikkhuset, er det ønskelig med en 3D-modell av kretskortet. Dette krever at fotavtrykkene har en 3D-modell for hver av komponentene som skal plasseres på det endelige fotavtrykket.

Skjemategnings-symbol

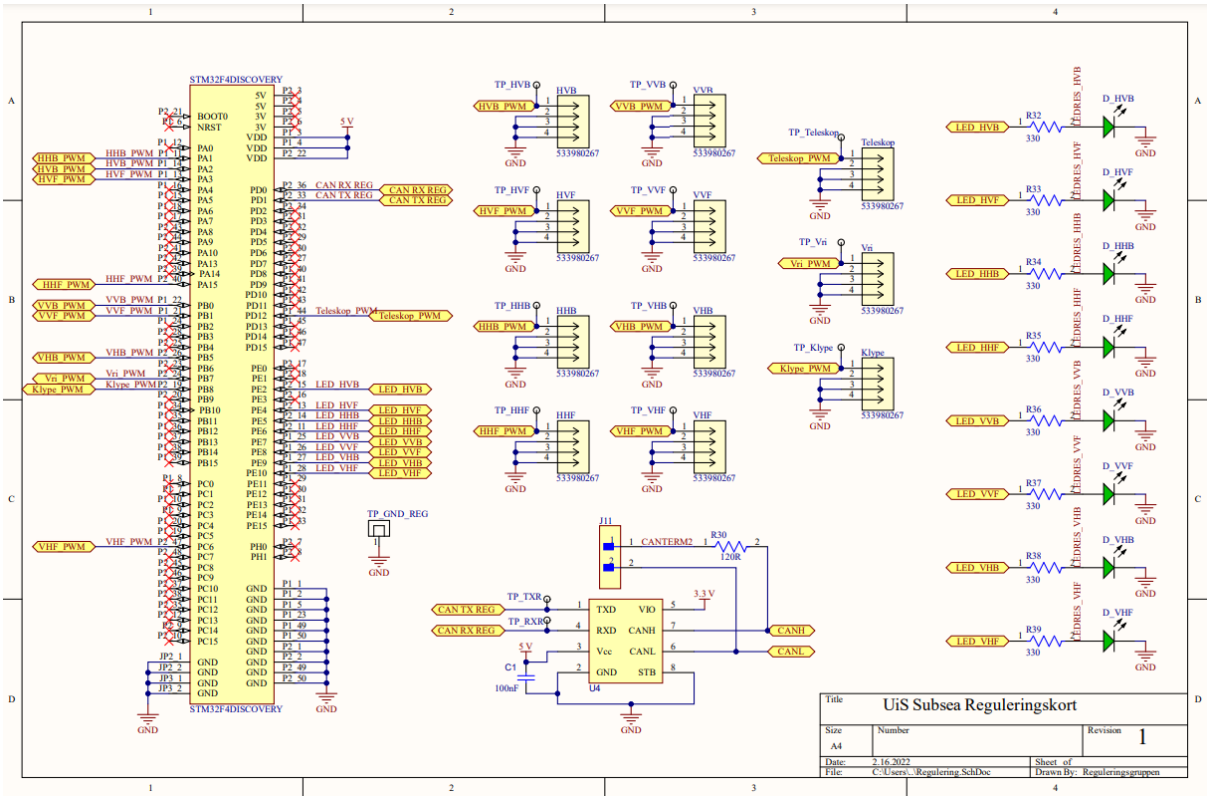
For å lage skjemategningen er det viktig at alle komponentene har et symbol. I figur 4.4 vises symbolet for CAN-buss transceiveren TJA1044GT/3Z. Dette symbolet har pinner hvor andre komponenter kan kobles til. Disse pinnene har en *designator*, et navn som lar Altium knytte pinnene opp mot paddene i fotavtrykket.



Figur 4.4: Skjemategning for CAN-buss transceiveren TJA1044GT/3Z.

Navnene på pinnene gjør det enkelt å identifisere hva som skal kobles til, og gjør det enklere å forstå virkmåten til komponenten.

Symbolet importeres til skjemategningen og knyttes opp mot andre komponenter med ledninger. I figur 4.5 vises den ferdige skjemategningen for hele reguleringsdelen av kretskortet.



Figur 4.5: Ferdig skjemategning for reguleringsdelen av kretskortet.

Utviklingskortet er plassert til venstre i skjemategningen. Øverst i midten er *PicoBlade* pluggene plassert med testpunkt for hvert av PWM-signalene. Nederst i midten av skjemategningen er CAN-busstransceiveren plassert med testpunkt for TX og RX. Det er ikke plassert testpunkt for

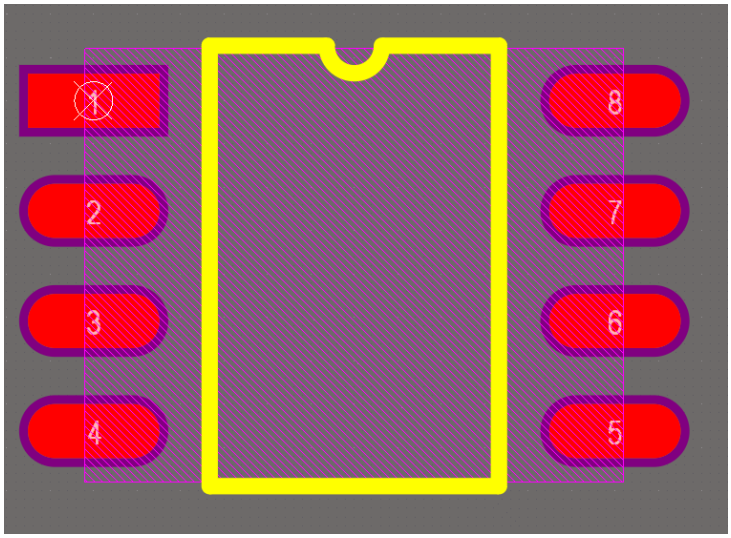
CANH og CANL ettersom dette er gjort på sensordelen av kortet. Til høyre i skjemategningen er LED-ene plassert. For å gjøre skjemategningen mer oversiktlig, er det brukt porter til å koble de forskjellige kretsene sammen. Dette gjøres for å unngå at det går ledninger på kryss og tvers av skjemategningen.

Fotavtrykk

Hvert symbol er, som nevnt tidligere, knyttet opp mot et fotavtrykk. Dette fotavtrykket representerer det fysiske avtrykket som er nødvendig for at komponenten skal kunne loddes på kretskortet. For mange komponenter er det standard størrelser på fotavtrykkene. LED-ene og motstandene vi har valgt å bruke er av størrelsen 0603, etter den imperiske[2] koden. Ettersom fotavtrykkene ligger i et eget bibliotek, kan flere typer komponenter med samme fysiske størrelse kobles opp mot samme fotavtrykk.

Fotavtrykket inneholder den informasjonen som er nødvendig for å kunne produsere kretskortet. Størrelsen og formen på kobberpaddene komponentene skal loddes til er viktig informasjon. Dersom komponenten ikke er overflatemotert, men krever et hull, definerer man størrelsen på hullet, og hvor stor kobberpadde som skal være rundt hullet.

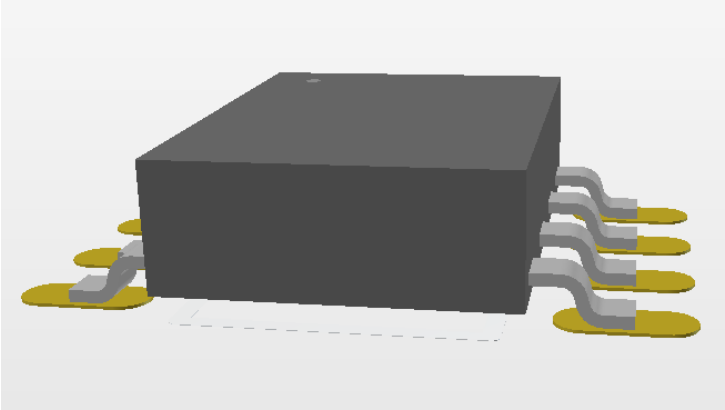
I figur 4.6 vises fotavtrykket til CAN-buss transceiveren, TJA1044GT/3Z, som skal monteres på kretskortet.



Figur 4.6: Fotavtrykket for CAN-buss transceiveren TJA1044GT/3Z.

Dette er en overflatemontert komponent, og de røde områdene viser kobberpaddene som skal være eksponert. Nummeret på paddene er *designatoren* som er knyttet opp til pinnene med samme nummer i symbolet. I figuren er det også et rosa rektangel som representerer plasseringen av en 3D-modell tilknyttet fotavtrykket. I figur 4.7 vises fotavtrykket i 3D med modellen av CAN-busstransceiveren plassert på toppen.

[2]Etter det imperiske målesystemet. Størrelsen er gitt som 0.06 × 0.03 tommer.



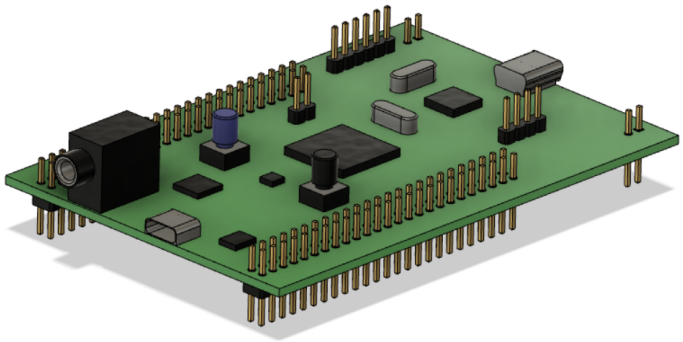
Figur 4.7: 3D modell av CAN-buss transceiveren TJA1044GT/3Z.

Tilsvarende fotavtrykk må altså enten lastes ned, eller lages selv, for alle komponentene på kretskortet.

3D-modeller

I dette prosjektet er det hensiktsmessig at det genereres en 3D-modell av kretskortet, slik at det kan verifiseres at komponenter ikke kolliderer inne i elektronikkhuset. 3D-modeller av komponenter er også en god måte å verifisere at fotavtrykket har riktig størrelse.

For noen av komponentene, slik som PicoBlade-pluggene fra MOLEX, har MOLEX ferdiglagede fotavtrykk med 3D-modeller som gjør det enkelt å implementere disse i skjemategningen. For mikrokontrolleren STM32F4DISCOVERY var kun fotavtrykket tilgjengelig, og det var derfor nødvendig å lage en egen 3D-modell av utviklingskortet. Dette ble gjort i *Autodesk Fusion 360*, og den ferdige modellen er vist i figur 4.8.



Figur 4.8: 3D modell av mikrokontrolleren STM32F4DISCOVERY

Modellen inneholder ikke alle små komponenter på utviklingskortet, men er laget detaljert nok til å kunne verifisere at ingenting kolliderer i elektronikkhuset.

4.1.2 Utlegg

Ettersom kretskortet skal produseres av produsenten *JLCPCB*, åpner dette for at kortet kan ha flere enn to lag. Det ble derfor bestemt i fellesskap med sensorgruppen at kortet skulle ha 4 lag. Det er da mulig å plassere komponenter både i topp- og bunnlaget, samt trekke signaler i disse lagene. Det nest øverste laget blir da et jordlag, og det nest nederste et spenningslag. Bruken av et jordlag og et spenningslag gjør at man ikke avhenger av at tykkelsen på banene er tilstrekkelig til å levere og returnere nok strøm. I Altium blir jordlaget laget ved å plassere et *Polygon Pour*, som fyller hele det markerte området med kobber.

Det er et krav at den totale tykkelsen på kortet er 1.6 mm, slik at gullfingrene passer inn i PCI-kontakten. JLCPCB har et eget oppsett for en stakk på tykkelse 1.6 mm. Denne er vist i figur 4.9.

#	Name	Material	Type	Weight	Thickness	Dk
	Top Overlay		Overlay			
	Top Solder	Solder Resist	Solder Mask		0.4mil	3.5
1	Top Layer		Signal	1oz	1.378mil	
	Dielectric 1	PP-006	Prepreg		7.874mil	4.1
2	Layer 1	CF-004	Signal	1oz	0.689mil	
	Dielectric 2	Core-029	Core		41.929mil	4.8
3	Layer 2	CF-004	Signal	1oz	0.689mil	
	Dielectric 3	PP-006	Prepreg		7.874mil	4.1
4	Bottom Layer		Signal	1oz	1.378mil	
	Bottom Solder	Solder Resist	Solder Mask		0.4mil	3.5
	Bottom Overlay		Overlay			

Figur 4.9: Oversikt over stakken hentet fra Altium.

De ytterste lagene har en tykkelse på 0.04 mm, eller 1.378 mils, og de to innerste lagene en tykkelse på 0.0175 mm, eller 0.689 mils. Hvis man summerer alle tykkelsene oppgitt i stakken får man en total tykkelse på 62.6 mils som tilsvarer en tykkelse på 1.59 mm.

Flere hensyn må tas når utlegget designes. Selv om det ikke brukes analoge signaler på kretskortet, er det hensiktsmessig å minimere smitte av signal mellom banene. Banene plasseres derfor med relativt god avstand, og dersom de må krysses, skjer dette slik at banene treffer hverandre med 90 grader. Det er også viktig at lengden på CANH- og CANL-banene er like slik at det ikke blir forskjeller i forsinkelsen gjennom disse. Linjene går også parallelt slik at eventuell støy påvirker begge linjene, som igjen resulterer i at den differensielle overføringen holdes upåvirket.

Det er valgt å bruke en banebredde på 10 mils, som tilsvarer 0.254 mm. Det er viktig at kobberbanene som leverer strøm til LED-ene er brede nok til å tåle at det går en strøm på ca 4 mA gjennom dem kontinuerlig. Ettersom alle banene til LED-ene er i enten topp- eller bunnlaget, har disse en tykkelse på 1.378 mils. Ved å bruke en formel som gir hvor mye strøm som kan gå i banen for å oppnå en gitt temperaturendring, kan man verifisere at banen er bred nok. Formelen for baner i topp- eller bunnlag er gitt som [47]:

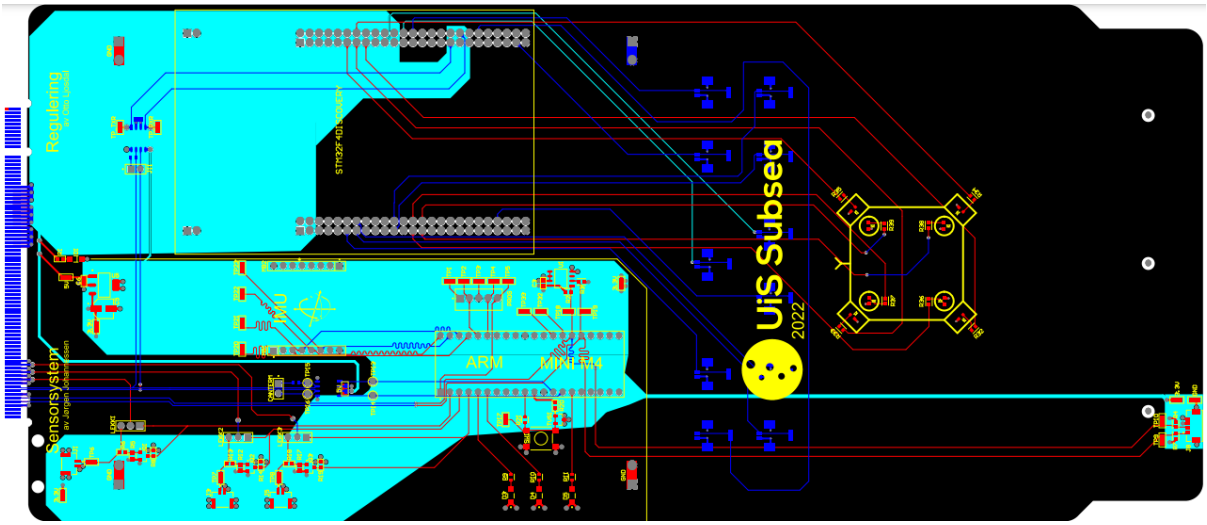
$$I = 0.048 \cdot \Delta T^{0.44} \cdot A^{0.725} \tag{4.2}$$

Hvor ΔT er den tillatte temperaturendringen i °C, og A er arealet av tversnittet i mils.

Setter man inn verdiene for banene i topp- og bunnlaget, og tillater en temperaturøkning på 1 °C, resulterer det i at banen tåler en strøm på ca. 32 mA. En banebredde på 10 mils er altså

mer en nok for å tåle strømmen LED-ene trekker.

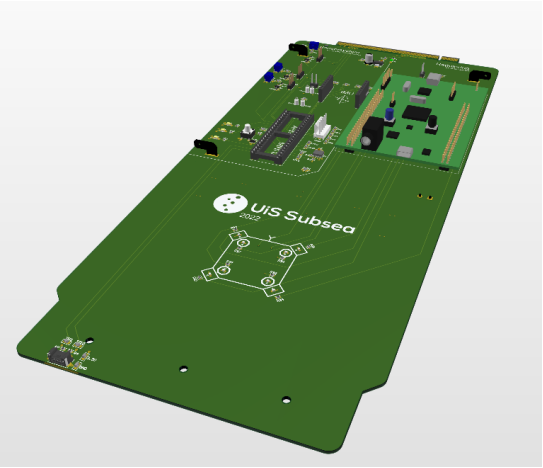
Kretskortet lages, som tidligere nevnt, felles for sensorgruppen og reguleringsgruppen. Kortet er delt inn i to hoveddeler slik at de to kretsene adskilles. Ettersom det kun er én PCI-kontakt på kortet, er det likevel noen kanaler som blir felles for begge halvdelene. Begge kretsene får strømforsyning fra de samme kanalene på PCI-bussen. Reguleringsdelen mottar 3.3 V fra sensordelen, og CAN-busslinjene går inn på de samme pinnene på PCI-kontakten. Utover dette har de to delene av kortet separate kretser. Det ferdige utlegget er vist i figur 4.10. Jordlaget vises ikke her da dette dekker hele kretskortet.



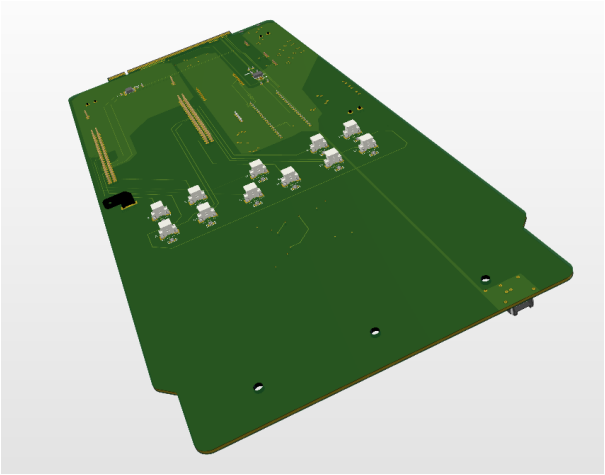
Figur 4.10: Det ferdige utlegget.

Rødt indikerer at banen går i topplaget, mørkeblått at banen går i bunnlaget og turkis at banen går i spenningslaget. Det sorte er formen på kretskortet, og det gule er et silketrykk som vil printes på toppen av kretskortet. Det turkise spenningslaget er delt i 2 deler, hvor den nederste delen tilhører sensordelen og har et spenningsnivå på 3.3 V. Den øverste delen har et spenningsnivå på 5 V.

3D-modellen av det ferdige kretskortet er vist i figur 4.11.



(a) 3D modell av det ferdige kretskortet sett fra toppen.



(b) 3D modell av det ferdige kretskortet sett fra bunnen.

Figur 4.11

Denne 3D-modellen blir brukt til å verifisere at komponentene på kretskortet får plass i elektronikkhuset, og at ingenting kolliderer med hverandre.

4.1.3 Montering

Kretskortet blir levert av JLCPCB uten påmonterte komponenter, og disse måtte derfor monteres for hånd. Før komponentene ble montert på kretskortet, måtte det brukes et multimeter for å verifisere at det ikke var kortslutning mellom de forskjellige delene av kretskortet, samt at det var kontakt der det skulle være kontakt. Gjennom hele prosessen ble det brukt et jordingsarmbånd for å beskytte komponentene mot statisk elektrisitet, ESD. Dette er spesielt viktig når komponentene håndteres direkte. For å gjøre loddingen enklere, ble det brukt flussmiddel på alle paddene. Dette gjør at loddetinnet enklere fester seg til kobberen, og sikrer elektrisk kontakt. Flussmiddel kan i enkelte tilfeller innholde etsende stoffer, og det ble derfor vasket av etter loddingen. Dette forhindrer at flussmiddelet forårsaker korrosjon av det eksponerte metallet.

Under montering ble det oppdaget en liten feil med kretskortet. Mikrokontrolleren er forsynt med 5 V fra kraftforsyningsmodulen via PCI-kontakten. På kretskortet ble dette originalt koblet til VDD på utviklingskortet, men databladet til mikrokontrolleren [44] er det oppgitt at VDD tåler en maksimal spenning på 4 V. Dette ble derfor koblet om ved hjelp av en ledning slik at tilførselen blir levert til 5V-pinnene på utviklingskortet. Pinnene som originalt koblet VDD til 5V-nettet på kretskortet ble fjernet. I følge brukermanualen , side 18, er det anbefalt å bruke 5V-pinnene som strømtilførsel [43]. I figur 4.12 vises ledningen som kobler 5 V til 5V-pinnene på utviklingskortet.

Figur 4.12: Ledning som kobler 5 V til 5V-pinnene på utviklingskortet.

Den oransje pilen indikere hvor pinnen VDD var koblet til før. Denne pinnen er fjernet, og ledningen er loddet direkte på kretskortet. Den blå pilen indikerer hvor ledningen er loddet til 5V-pinnene slik at disse forsynes med 5 V.

I figur 4.13 vises det ferdige kretskortet med alle komponenter montert.

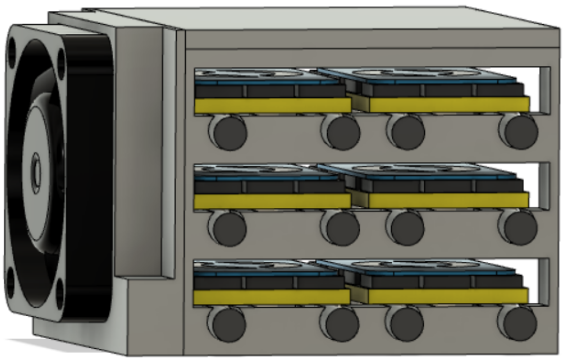
(a) Det ferdige kretskortet sett fra toppen. **(b)** Det ferdige kretskortet sett fra bunnen.

Figur 4.13

Testrapportene i vedlegg B.2 og B.3 presenterer tester som viser at kretskortet virker som ønsket.

4.2 Plassering av motorkontrollere

Motorkontrollerene er ikke vantette og må derfor plasseres inne i elektronikkhuset. Det er ikke ønskelig at disse ligger løst inne i huset, og det er derfor konstruert et stativ for disse vist i figur 4.14.



Figur 4.14: 3D modell av hus til å holde motorkontrollere, laget i Fusion 360.

Stativet har plass til 6 motorkontrollere, og det er derfor behov for to av disse. Planen er å plassere stativene på kraftforsyningskortet slik at ledningene som forsyner motorkontrollerene med 12 V blir kortest mulig.

Reguleringskortet vil være plassert rett over kraftkortet i elektronikkhuset. På reguleringskortet er pluggene plassert på undersiden, som gjør det enkelt å koble ledningene for PWM-signalet fra motorkontrollerene direkte til reguleringskortet.

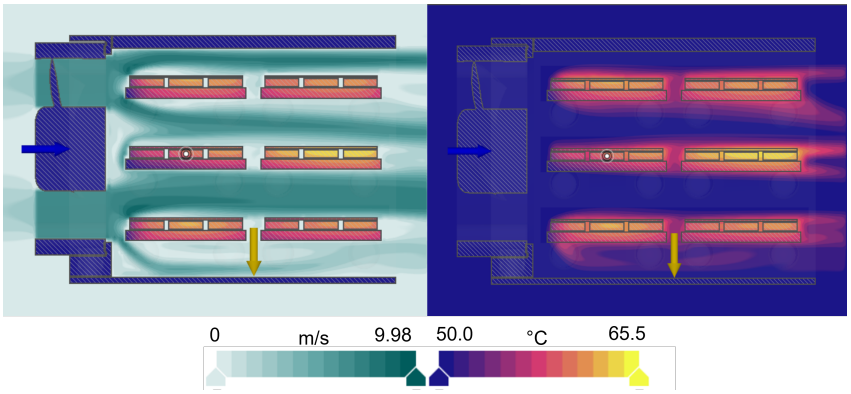
Holderene blir 3D-printet i ABS-plast slik at de kan tåle temperaturer på opp mot 100 °C. Det er også påmontert en vifte, slik at det holdes en kontinuerlig luftstrøm over motorkontrollerene. Denne viften er av typen MR3010H12B, og har en luftstrøm på 0.18 kubikkmeter per minutt [29]. Stativene er plassert slik at den varme luften blåses direkte på aluminiumskroppen til elektronikkhuset, slik at luften kjøles ned mest mulig effektivt.

Spenningen som forsyner driverkretsen på motorkontrollerene er på ca 10.5 V, som vist i kapittel 3.2.3. MOSFET-ene brukt i motorkontrolleren kan i følge figur 12, side 5, i databladet, ha en motstand, $R_{DS(on)}$, på opptil 2.5 mΩ med en V_{GS} på 10 V [19]. Ettersom to og to er koblet i parallell, vil hvert av de 6 parene med MOSFET-er ha en motstand på 1.25 mΩ. Det vil maksimalt gå ca. 10 A gjennom hver fase, og effektapet i hver motorkontroller vil på det maksimale da være:

$$\text{Effekttap} = Antall \cdot R \cdot I^2 = 6 \cdot 1.25 \cdot 10^{-3} \cdot 10^2 = 0.75W \tag{4.3}$$

Fordeles dette effekttapet mellom de 12 MOSFET-ene ender det på 0.0625 W per MOSFET. Ved å bruke et simuleringsverktøy i Autodesk Fusion 360, kan man få en indikasjon på at viften er god nok til å kjøle motorkontrollerene tilstrekkelig. Motorkontrollerene kan bli opp til 140 °C

før de begynner å miste effekt [35]. Lufttemperaturen er estimert til å være 50°C, ettersom det er rimelig å forvente at det kan bli relativt varmt i elektronikkhuset. Resultatet fra simuleringen er vist i figur 4.15.

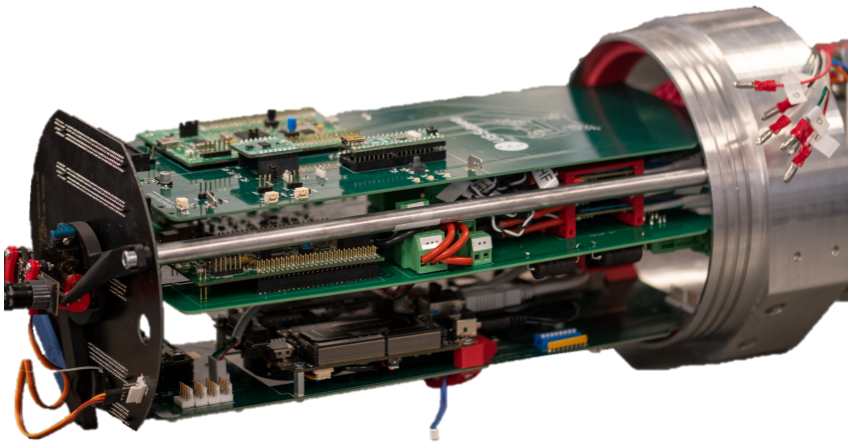


Figur 4.15: Temperatur og luftstrøm gjennom holder med motorkontrollere

Ifølge simuleringen vil noen av MOSFET-ene kunne bli opptil 65.5 °C, som er langt under grensen for hva motorkontrolleren tolererer. Simuleringen viser en lufthastighet som virker i overkant stor, men selv om denne i virkeligheten er lavere, indikerer likevel simuleringen at luftgjennomstrømningen vil være god.

4.3 Konklusjon

Utover motorer og motorkontrollere er det behov for en del komponenter for å knytte systemet sammen. Kretskortet som har blitt utviklet, er et essensielt bindeledd mellom mikrokontroller, motorkontrollere og det resterende systemet. I testkapittel B.2 og B.3 testes kretskortet. Testen viser at tranceiverkretsen og mikrokontrolleren virker. I tillegg har det også blitt designet stativer for motorkontrollerene slik at disse kan plasseres på kraftkortet i nærheten av forsyningen. Holderene sørger også for at motorkontrollerene får nok kjøling. Utviklingen av maskinvaren har gitt et godt grunnlag for å videre kunne implementere systemets programvare. Figur 4.16 viser elektronikken montert i elektronikkhuset. Kretskortet omtalt i dette kapittelet er det øverste kortet, og motorkontrollerene er plassert i de røde stativene i bakenden på kortet under.



Figur 4.16: Ferdig montert elektronikk i elektronikkhuset.

Produksjonsfilene fra Altium kan lastes ned via linken i vedlegg A.

Kapittel 5

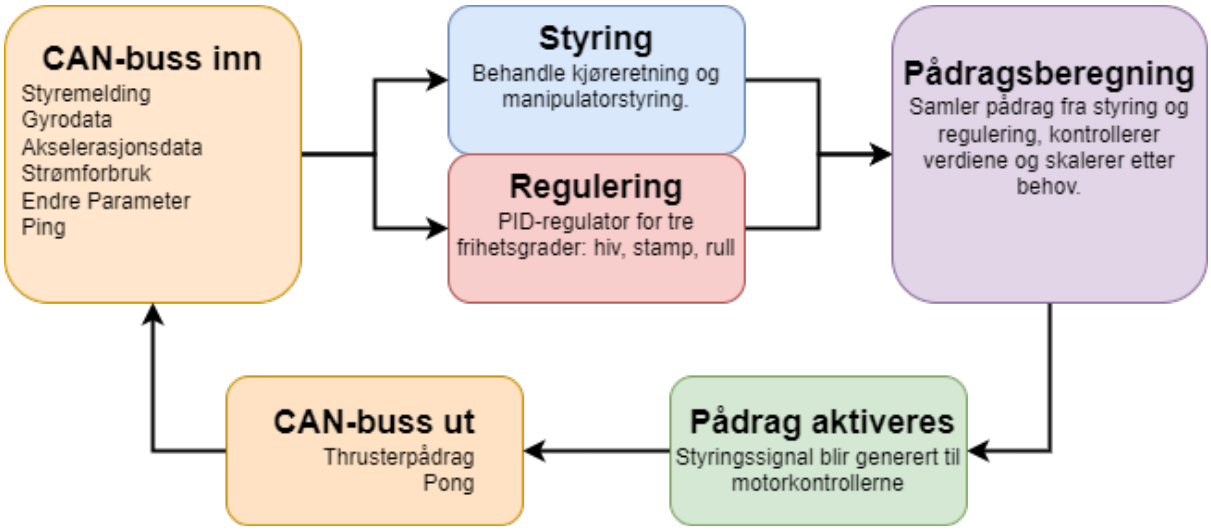
Programvare

Kapitteloversikt

Dette kapittelet tar for seg programmeringsprosessen fra utviklerverktøy til programstruktur. Hensikten med kapittelet er å gjøre rede for hvilke behov systemet har, og hvordan mikrokontrolleren blir utnyttet for å løse disse behovene. Kapittelet tar for seg initialisering og struktur, men spesifikk beskrivelse av funksjonene for styring og regulering blir gjennomgått i større detalj i kapitlene der disse er utledet.

5.1 Overordnet system

Hensikten med programvaren i mikrokontrolleren er å utføre styring og pådragsberegninger basert på kommandoer fra piloten og målinger fra sensorkortet. Enhetene i ROV-en kommuniserer over CAN-grensesnitt. Styringskortet mottar pilotinstruksjoner, målinger fra IMU for orientering og akselerasjon, og målt strømforbruk hvert 50 ms. Pilotinstruksjonene tolkes og tilpasser pådrag for hver thruster, og styrer hvordan manipulatoren skal oppføre seg. Tilsvarende behandles gyromålingene fra IMU-en ved hjelp av tre PID-regulatorer for å opprettholde stabil orientering i vannet. Mikrokontrolleren styrer pådraget til alle motorene ved å generere et pulsbreddemodulert signal til hver motorkontroller. Informasjon om pådrag for hver enkelt thruster blir også kommunisert tilbake til piloten, slik at det kan overvåkes i sanntid. Bildebehandlingsenheten sender også regelmessig en *ping* til de andre enhetene i ROV-en for å overvåke at ingen enheter har sluttet å fungere. Enhetene returnerer *pong* for å bekrefte at systemet fungerer som det skal. Figur 5.1 viser et overordnet blokkskjema av systemet på mikrokontrolleren.



Figur 5.1: Overordnet blokkskjema for programvaren.

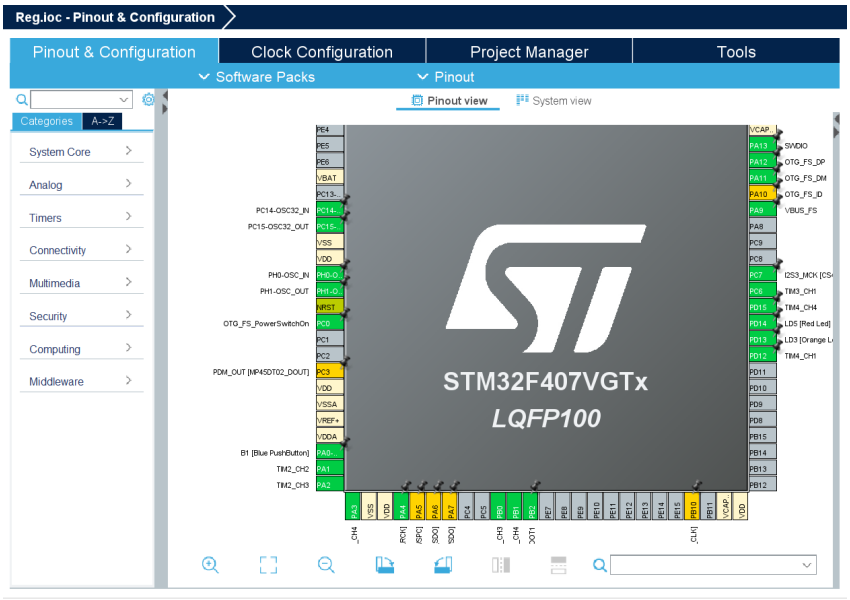
Delene av programvaren som går direkte på behandling av styringskommandoer og regulering blir forklart i større detalj i sine respektive kapitler, kapittel 6 og kapittel 8. Dette kapittelet tar i større grad for seg den generelle konfigurasjonen av mikrokontrolleren, programvarestrukturen, og hvordan verdiene fra styring og reguleringen blir behandlet og kontrollert, samt kommunikasjon inn og ut av mikrokontrolleren.

5.2 Konfigurasjon av mikrokontroller

Programmet, eller *Integrated Development Environment* (IDE), vi har benyttet til å programmere mikrokontrolleren er STM32CubeIDE. Utviklingsverktøyet er utgitt av STMicroelectronics, som også er produsenten av mikrokontrolleren. Integrert i utviklingsverktøyet, er et bibliotek med drivere og funksjoner. Ved å fortelle programmet hvilken type mikrokontroller som skal programmeres, vet programmet å finne en oversikt over alle spesifikasjonene til den enheten som skal konfigureres.

STM32CubeIDE er beregnet for å programmere i *C*, og dessuten er det driverstøtte for å programmere i to forskjellige språknivå. Dette innebærer lavnivå, hvor man adresserer registeradresser direkte, eller ved HAL, «*Hardware Abstraction Layer*», som er et noe høyere språknivå. Et høyere nivå programmeringsspråk gjør koden mer «leselig» og tildels mer oversiktlig, på bekostning av effektivitet. Tapet av optimalisering ved å bruke HAL-drivere er likevel så marginal at vi har valgt å bruke HAL-drivere for å konfigurere og initialisere mikrokontrolleren, men med innslag av lavnivå i egendefinerte funksjoner.

STM32CubeIDE tilbyr et grafisk grensesnitt, vist i figur 5.2, for å konfigurere oppsettet av mikrokontrolleren. Grensesnittet forenkler det å velge hvilke moduler som skal brukes, og det å konfigurere disse for hvilke funksjoner de skal utføre. Vårt behov av perifermoduler er i all hovedsak en CAN-modul for å realisere kommunikasjon til de andre mikrokontrollerne i ROV-en, og TIM-moduler som er timere. TIM-modulene brukes for å kunne generere pulsbreddemodulerte signaler. Det grafiske grensesnittet gjør det enkelt å konfigurere de nødvendige GPIO[1]-modulene, og rute modulene til ønskede utgangspinner.



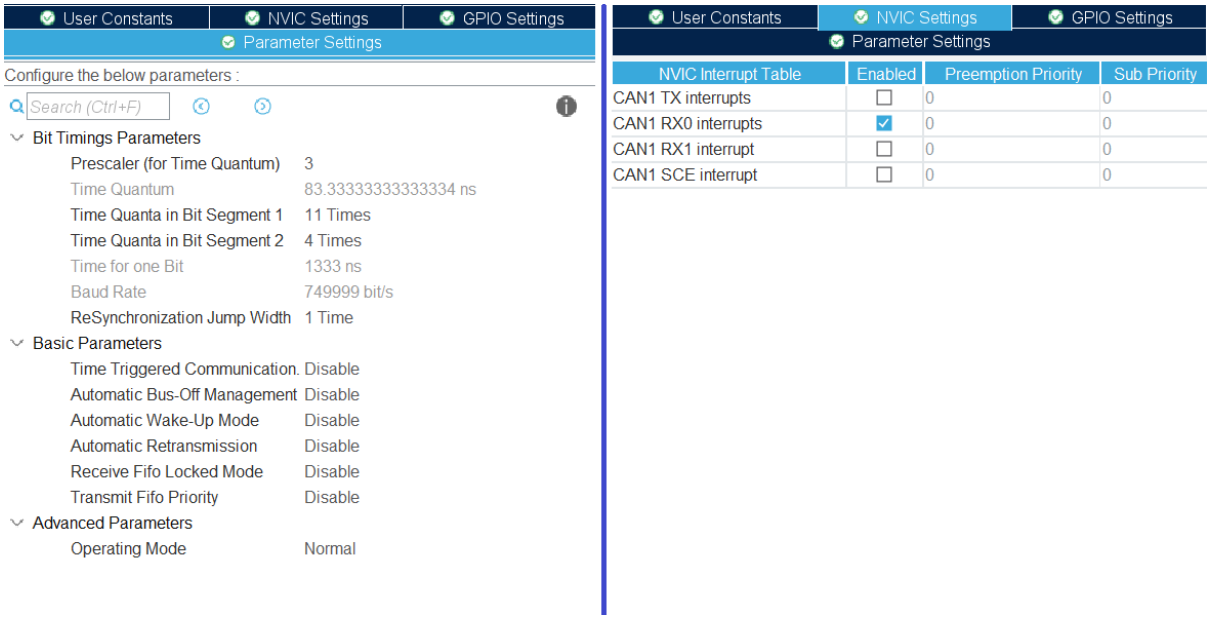
Figur 5.2: Grafisk fremvisning av mikrokontrolleren og hvilke pinner og moduler som er tilgjengelige.

Utviklerverktøyet genererer også prosjektfiler med kode som initialiserer de valgte modulene. I filene opprettes designerte steder for brukerdefinert kode og egendefinerte funksjoner. En ting som er viktig å presisere i forbindelse med det å skrive parameterverdier til registre, er at noen registre representerer et verdiområde som starter på 1, men verdiområdet i selve registeret starter på 0. Dette er tilfelle med flere av TIM-modulenes registre. For eksempel telleverdiene for *Clock Division*, *Counter Period* og *Pulse* starter verdiområdet på 1, siden det ikke gir mening å telle til «0». Dette opplever man eksempelvis når man konfigurerer TIM-modulen, som i vårt tilfelle har en systemklokke på 72 MHz. Ønsker man en *Prescaler* på 72, må man skrive 71 til gjeldende register for å ende opp med tellehastighet på 1 MHz. Dette vil være gjentredende i flere av konfigurasjonene av de følgende modulene.

[1] **G**eneral **P**urpose **I**nput **O**utput

5.2.1 CAN-modul

CAN-modulen realiserer kommunikasjon med de andre enhetene i systemet; sensorkortet, kraftforsyningskortet og bildebehandlingsenheten. Bildebehandlingsgruppen har hatt hovedansvar for kommunikasjonsgrensesnittet i ROV-en, og har derfor videre gjort arbeidet med konfigurasjonen av CAN-modulen. Alle fire mikrokontrollerne kommuniserer over den samme CAN-bussen, og derfor er alle konfigurert likt. Modulen er satt opp med en «*BaudRate*» på 750 000. RX for CAN-modulen er koblet til pinne PD0, og TX er koblet til PD1. I tillegg er det aktivert avbruddshåndtering på mottaksregisteret RX0. Altså når det mottas en melding som skrives inn i mottaksregisteret RX0, blir et avbruddsflagg aktivert. Mikrokontrolleren håndterer avbruddet umiddelbart, før den fortsetter i prosessen den jobbet med idet avbruddet inntraff. Mikrokontrolleren mottar rundt 80 meldinger per sekund over CAN-bussen, men hvert avbrudd tar knapt et par mikrosekund. *BaudRate* tilsier at overføringshastigheten er 750 000 bit per sekund, slik at et bit tar 1.33 μs. I figur 5.3 vises en oversikt over innstillingene for CAN-modulen.



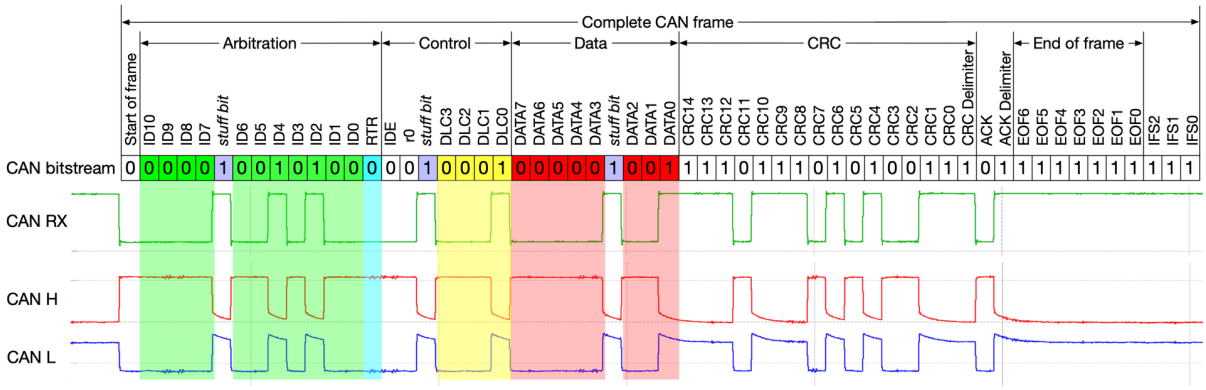
Figur 5.3: Innstillinger for CAN-modulen. Venstre side er generelle parametere og høyre side viser aktivering av avbrudd.

CAN-modulen filtrerer innkommende meldinger basert på en 8-bit ID-maske. Hver enhet koblet til bussen er designert hvert sitt unike ID-område:

- Styringskontrolleren: 64-95
- Sensorkontrolleren: 96-127
- Kraftkontrolleren: 128-159
- Bildebehandlingskontrolleren: 160-191

I tillegg til å være en adresse, fungerer ID-masken også som prioritering på bussen. Det vil si at meldinger med lav ID i meldings-«*headeren*», har prioritet på bussen hvis det oppstår kø hvor flere kontrollere prøver å sende samtidig. Meldinger med lavere prioritet venter på tur for å sende

data over bussen. I tillegg til ID og databit, sendes også en rekke kontrollbit. En oversikt over dette er vist i figur 5.4.



Figur 5.4: Oversikt over alle elementer i en CAN-melding. I dette tilfelle sendes 1 byte med data, markert med rødt. Hentet fra [52], kreditt: [50].

I tabell 5.1 vises en oversikt over innholdet i de ulike CAN-meldingene.

Tabell 5.1: Oversikt over en CAN-melding. Dominante bit er 0, og recessive bit er 1.

Navn	Størrelse (bit)	Beskrivelse
SOF	1	«*Start-of-frame*»
ID	11	Identifisering og prioritet
stuff bit	1	Fyllbit med invers polaritet, «*Run-length limited*»
RTR	1	«*Remote Transmission Request*»
IDE	1	«*Identifier extension bit*»
r0	1	Dominant kontrollbit
Data Lengde	4	Antall data-bytes 0-8
Data	8-64	Data som overføres
CRC	15	Feildetektering «*Cyclic redundancy check*»
CRC delimiter	1	Ressesivt kontrollbit
ACK slot	1	Sendes som ressesivt bit, settes dominant av mottaker
ACK delimiter	1	Ressesivt kontrollbit
EOF	7	«*End-of-frame*» ressesive bit
IFS	3	«*Inter-frame spacing*» ressesive bit

En melding kan innholde alt fra én til åtte byte data, som tilsammen utgjør en lengde på mellom 56 og 112 bit per melding. Dette tilsvarer opp til 149.33 μs for å sende en full melding over bussen.

5.2.2 Timer-modul

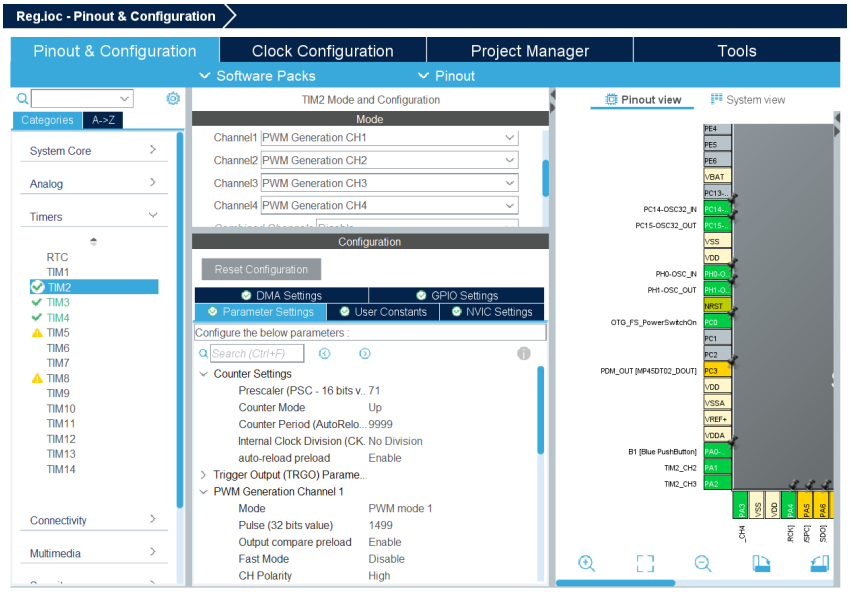
Hovedfunksjonen til styringskontrolleren er å generere signalene som går til motorkontrollerne. Motorkontrollerne styres ved hjelp av et PWM-signal, hvor bredden av pulsen bestemmer pådraget til motorene. ROV-en har tilsammen 8 thrustere og 3 manipulator-motorer som skal styres individuelt av hvert sitt PWM-signal. Altså er det behov for å generere 11 unike PWM-signal. Hver timer-modul har fire kanaler som deler periode, men har individuelle pulsbredder.

Med behov for 11 utganger har vi initialisert 3 timer-moduler, hvor TIM2 genererer signaler for horisontale thrustere, TIM3 for vertikale thrustere og TIM4 for motorene på manipulatoren.

Overordnet er alle timer-modulene konfigurert med følgende innstillinger:

- *Prescaler*: 71
- *Counter Period*: 9999
- *Pulse*: 1499
- *Output Compare*: On
- *Counter Mode*: Up

I figur 5.5 vises implementeringen av disse innstillingene i TIM2-modulen i STM32CubeIDE.



Figur 5.5: Innstillinger for TIM2 i STM32CubeIDE.

Innstillingene setter tellehastigheten til 1 MHz, telleperioden til 10 000 μs, og pulsbredden til 1500 μs. Motorkontrollerne justerer pådraget til motorene ut fra en pulsbredde mellom 1100 og 1900 μs, hvor 1500 μs er nøytral. Ved nøytral står motorene stille. Ved kortere pulsbredde øker pådrag i retning mot klokken, og lengre pulsbredde øker pådrag med klokken. Motorene kan derfor styres av funksjoner som endrer telleverdien til pulsbredden av hvert PWM-signal. I tabell 5.2 vises en oversikt over hvordan thrustere og manipulator-motorer er fordelt på de forskjellige TIM-modulene.

Tabell 5.2: *Pin-out* for *timer*-modulene

Timer-modul	Kanal	Pinne	Motorenhet
TIM2			**Horisontale thrustere**
	1	PA1	Høyre Foran
	2	PA2	Høyre Bak
	3	PA3	Venstre Bak
	4	PA15	Venstre Foran
TIM3			**Vertikale thrustere**
	1	PB0	Høyre Foran
	2	PB1	Høyre Bak
	3	PB5	Venstre Bak
	4	PC6	Venstre Foran
TIM4			**Manipulator**
	1	PB7	Teleskop
	2	PB8	Rotasjon
	3	PD12	Klype

5.3 Programstruktur

Vi har hatt fokus på at koden skal være lett leselig, oversiktlig og ikke minst rask å utføre for mikrokontrolleren. Tiltakene som er gjort for å gjøre koden lett leselig, er blant annet bruken av *masker* for faste bitmønster eller verdier, og å opprette egendefinerte *structs* for å gruppere variabler. I vårt tilfelle har vi flere prosesser som bruker identiske sett med variabler og parametere. For eksempel skal det brukes tre PID-regulatorer. Parametre og koeffisienter i regulatorene har samme navn, men forskjellige verdier. Ved å sette opp variablene i en struct vil alle regulatorene ha like parameternavn, men tilhøre forskjellige «*PID-struct*»-instanser.

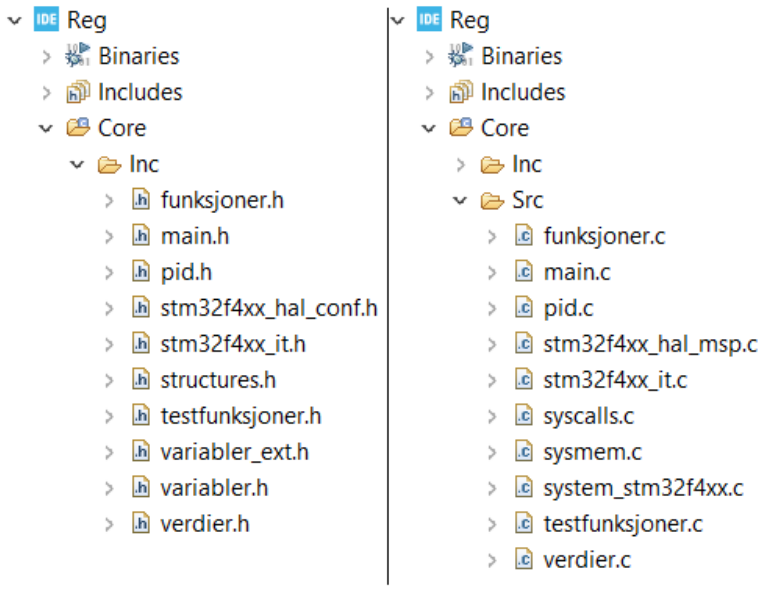
Koden i utsnitt 5.1 viser eksempler av masker og egendefinerte *structs*. Maskene LED_XXX brukes for å sette bitmønsteret i ODR-registeret som går ut til indikatordiodene på kretskortet. LED-ene brukes for å simulere utgangene til thrusterene, og for å teste at pådraget fra funksjonene blir sent til riktig motorkontroller. Manipulatormaskene blir brukt i manipulatorstyringen, blant annet for å bestemme hvilken manipulatorfunksjon som har prioritet i forhold til effektbudsjettet. En instans av struct-en «*gyrostruct*» blir opprettet som «*gyrodata*». Når styringskontrolleren mottar målinger fra sensorkortet, blir disse verdiene skrevet inn i *gyrodata*, slik at for eksempel *gyrodata.hiv* alltid representerer den nyeste dybdeposisjonen. Styrefunksjonene og PID-regulatorene regner ut thrusterpådrag hver for seg, og har derfor hver sin instans av struct-en «*thruster_sett*». Slik er det mulig å hente ut thrusterbidraget fra for eksempel *styrebidrag.hhf* eller *stampbidrag.vhb*. Standardiseringen av variabelnavnene gjør koden lettleselig og enkel å feilsøke.

Kode 5.1: Kodeutsnitt av masker i *main.h* og structs i *structures.h*

```
1 ...
2 #define LED_VHB ((uint16_t)0x0200) // LED  Pin 9
3 #define LED_VVB ((uint16_t)0x0080) // LED  Pin 7
4 #define LED_VVF ((uint16_t)0x0100) // LED  Pin 8
5 ...
6 #define MAN_KLYPE       ((uint16_t)0b00110000) // Manipulatorklypa
7 #define MAN_KLYPE_LUKKE ((uint16_t)0b00100000) // Manipulator lukke klypa
```

```
8  #define MAN_VRI          ((uint16_t)0b00001100) // Manipulator vri/rotasjon
9  ...
10 typedef struct {
11     int16_t rull;    // rullvinkel i grader
12     int16_t stamp;   // stampvinkel i grader
13     int16_t hiv;     // dybdeposisjon i cm
14     int16_t gir;     // girvinkel (ikke aktiv)
15 } gyrostruct;
16 ...
17 typedef struct {
18     int16_t hhf;     // Thruster Horisontalt Hoeyre Foran
19     int16_t hhb;     // Thruster Horisontalt Hoeyre Bak
20     int16_t hvb;     // Thruster Horisontalt Venstre Bak
21     int16_t hvf;     // Thruster Horisontalt Venstre Foran
22     int16_t vhf;     // Thruster Vertikalt Hoeyre Foran
23     int16_t vhb;     // Thruster Vertikalt Hoeyre Bak
24     int16_t vvb;     // Thruster Vertikalt Venstre Bak
25     int16_t vvf;     // Thruster Vertikalt Venstre Foran
26 } thruster_sett;
27 ...
```

For å gjøre koden mer oversiktlig er mesteparten av den egendefinerte koden kategorisk fordelt i forskjellige *source*- eller *header*-filer. En oversikt over alle disse filene er vist i figur 5.6. Dette frigjør *main.c* fra en mengde med kode, slik at den oversiktlig viser hvilke funksjoner som kjøres. Funksjonene vil kun aktiveres av flagg fra andre filer. Kodeutsnitt 5.2 på side 73 viser hvordan hele programmet blir forenklet ned til to «*if*-setninger» inni den uendelige løkken i *main.c*.



Figur 5.6: Liste av programfilene til mikrokontrolleren.

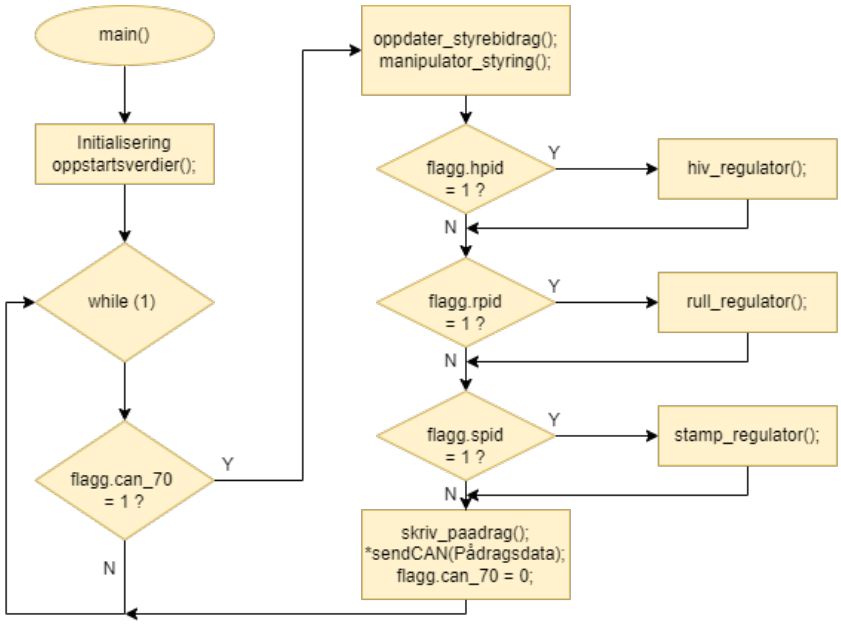
Flere av støttefilene eksisterer både som header- og source-filer. I disse tilfellene brukes header-filen til å deklarere funksjonene i source-filen. Innholdet i hver av filene er som følger:

- *variabler.h* og *variabler_ ext.h*:
  variabler.h deklarerer alle variabler og instanser av structs som skal brukes i mikrokontrolleren. *variabler_ ext.h* deklarerer at alle variablene og struct-ene skal ha egenskapen *extern*, slik at de kan brukes globalt.

- *verdier.c*:
  Inneholder funksjoner for oppstartsverdier og en funksjon som gjør det mulig å endre alle parametre og koeffisienter fra toppsiden.
- *structures.h*:
  Inneholder alle egendefinerte structs-typer
- *funksjoner.c*:
  Inneholder funksjonene for styring og kontrollering av pådragsberegning, all skalering, kontroll og konvertering. De eneste egendefinerte funksjonene som ikke er i denne filen er testfunksjoner, CAN-kommunikasjon og selve PID-funksjonene.
- *pid.c*:
  Inneholder PID-regulatorfunksjonene for hiv, stamp og rull.
- *testfunksjoner.c*
  Samling av testfunksjoner for dioder, styring og pådragsberegning.
- *stm32f4xx_it.c*:
  Tar for seg avbruddshåndtering av SysTick og CAN1_RX0.

5.3.1 Hovedprogram

Hovedprogrammet består av én løkke som forteller hva kontrolleren skal gjøre, og flytskjemaet i figur 5.7 viser hvordan løkken fungerer.



Figur 5.7: Overordnet programstruktur.
**sendCAN()* er pseudo-kode for leselighetens skyld. Faktisk kode er vist i kodeutsnitt 5.2.

Som figuren over viser, utføres bare prosessene i hovedprogrammet når *flagg.can_70* er satt. Dette flagget signaliserer at det er mottatt en ny styringspakke over CAN-buss, som har ID 70. Hovedprogrammet består av fire hovedfunksjoner; navigasjonsstyring, manipulatorstyring, regulering og pådragsprosessering. Reguleringen består av tre PID-regulatorer, én for hver av

frihetsgradene hiv, rull og stamp. Disse har hvert sitt individuelle flagg, slik at regulatorene enkelt kan slås av og på fra toppsiden. Hiv-regulatoren har en interessant funksjon hvor den blir satt på pause hvis piloten sender instruksjoner om å endre dybdenivå. Flagget for hiv-regulering blir ikke satt høyt igjen før piloten har sluttet å sende dybdepådrag. Funksjonen som setter hiv-flagget igjen er beskrevet i avbruddshånderingen av *SysTick* i delkapittel 5.3.2. Videre viser kodeutsnittet under hvordan hovedprogrammet er implementert.

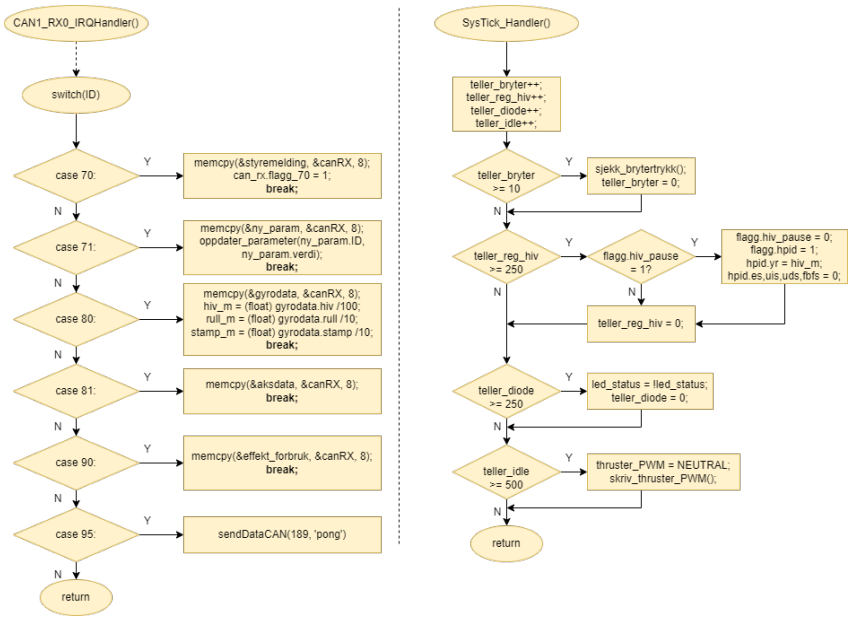
Kode 5.2: Kodeutsnitt av *while*-løkken i main

```
1     while(1){
2         if(flagg.can_70){   // Flagget signaliserer ny      mottatt styremelding
3           oppdater_styrebidrag();   // Behandler pilotinstruksjoner for ...
                navigasjon
4           manipulator_styring();    // Behandler pilotinstruksjoner for ...
                manipulator
5
6           if(flagg.spid){  stamp_regulator(); } // PID regulering av ...
                stampvinkelen
7           if(flagg.rpid){  rull_regulator();  } // PID regulering av ...
                rullvinkelen
8           if(flagg.hpid){  hiv_regulator();  }  // PID regulering av hivposisjon
9
10          behandle_paadrag();   // Paadragsprosessering
11
12          memcpy(&csend, &thrusterdata, 8); // Kopierer paadragsdata ...
13          sendDataCAN(170, &hcan1); // og sender til toppsiden
14
15          flagg.can_70 = 0; // Styring og regulering fullfort
16        }
17          if(brytertrykk){               // Utfoeres hvis det er registrert ...
                brytertrykk
18              // Testfunksjoner
19              brytertrykk = 0;
20        }
21    }
```

I stedet for å utføre utregninger og operasjoner, blir hovedprogrammet brukt til å starte opp andre programmer og funksjoner. Dette gjør at programflyten i den uendelige løkken er lett å lese, og enkel å forstå. Det er også en sjekk for brytertrykk i løkken. Bryteren er en fysisk bryter på utviklerkortet. Denne knappen er derfor kun beregnet for testing i utviklingsperioden, og irrelevant når ROV-en opererer under vann.

5.3.2 Avbruddshåndtering

Mikrokontrolleren er konfigurert til å benytte to aktive avbruddsmetoder. *SysTick* er et periodisk systemavbrudd, som kjøres hvert millisekund. *CAN_RX0* er et avbrudd som skjer hver gang mottaksregisteret til CAN-modulen er fylt med en ny melding. Kontrolleren pauser det aktive programmet som kjøres når den skal utføre avbruddsmetoder. Derfor er avbruddsmetodene satt opp til å være så effektive som mulig, slik at det ikke oppstår store forsinkelser i systemet. Flytskjema over begge avbruddsmetodene er vist i figur 5.8.



Figur 5.8: Avbruddshåndtering.

SysTick_Handler() kjøres hvert millisekund, og er derfor satt til å fungere som en smart stoppeklokke. Forskjellige tellervariabler blir iterert hver gang avbruddet inntreffer, og når tellerverdiene når bestemte størrelser, settes flagg som påvirker hvordan tilhørende funksjoner oppfører seg. For eksempel brukes tellervariabelen *teller_diode* til å slå av og på påvirkede dioder hvert 250 ms, slik at de blinker. En annen teller, *teller_reg_hiv*, brukes til å gjenstarte hiv-reguleringen. Hiv-translasjon kan enten styres manuelt, eller med en PID-regulator for å holde en konstant dybde. Systemet realiserer dette ved at funksjonen *oppdater_styrebidrag()* setter hiv-flagget lavt når piloten gir instruksjon om å endre dybde, som resulterer i at PID-regulatoren deaktiveres, se kode i utsnitt 6.1 i kapittel 6. Telleren i *SysTick* restarter PID-regulatoren etter telleren når 250 ms, og samtidig endres dybdereferansen *hpid.yr* til å være nyeste dybdemåling. Variablene som lagrer tidligere verdier resettes til å være null, slik at ikke regulatoren starter med store verdier i etterslep fra tidligere.

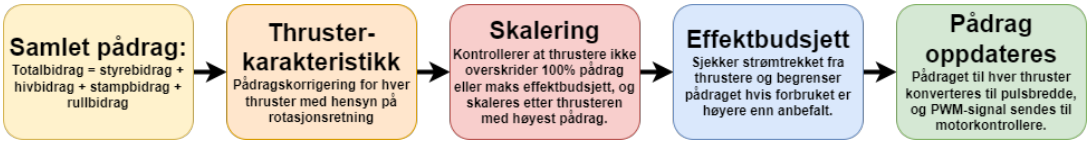
CAN1_RX0_IRQHandler() håndterer avbrudd fra mottaksregisteret til CAN-modulen. Funksjonen starter en *switch case* for meldingens ID. Hver aktuelle ID har en unik håndtering. For eksempel brukes ID 95 for å bekrefte tilkobling til resten av systemet. Når mini-PC-en sender PING over ID 95, svarer styringskontrolleren med PONG på ID 189. Håndtering av andre meldinger er vist over i figur 5.8.

Den viktigste QOL[2]-funksjonen som er implementert i mikrokontrolleren omhandler meldinger med CAN-ID 71. Avbruddsfunksjonen for CAN-meldinger kjører da en funksjon som heter *oppdater_parameter(ID, verdi)*. Funksjonen tar inn en *uint32_t* ID-variabel og en *float*-verdi. I denne funksjonen er alle flagg og verdier som er ønskelig å endre fra toppsiden tildelt hver sin ID. Eksempelvis kan regulatorflaggene endres. Dempingsverdier, filterparameter og PID-paramter oppdateres ved hjelp av denne funksjonen. Funksjonen gjør det enkelt å utføre tester av forskjellige innstillinger og parameterverdier fra toppsiden uten å måtte åpne elektronikkhuset for å manuelt endre verdiene. Funksjonen kan brukes uansett hvilken tilstand ROV-en er i, som for eksempel under flyving.

[2]**Q**uality **o**f **L**ife

5.3.3 Pådragshåndtering

Kapittel 6 og 8 vil senere gå gjennom hvordan pådragsberegninger skjer i henhold til styring og regulering. Beregningene skjer individuelt for manuell styring, og for tre ulike PID-regulatorer, hvor hver av de fire beregningene resulterer i et eget bidrag for pådraget. Mikrokontrolleren må behandle disse bidragene for å kunne sende signaler som motorene kan oppfatte. Det er satt opp et eget program for pådragshåndteringer, *behandle_paadrag()*, som samler inn bidragene fra de fire individuelle pådragsberegningene, og behandler verdiene før totalen blir sendt til motorkontrollerne. Figur 5.9 viser et flytskjema over pådragshåndteringen i mikrokontrolleren.



Figur 5.9: Kontrollering og skalering av pådragene til thrusterne.

Kodeutsnittet under viser hvordan programmet er implementert på mikrokontrolleren.

Kode 5.3: Kodeutsnitt av programmet som håndterer pådragene.

```
1  void behandle_paadrag(void){
2      // Paaforer et IIR filter pÃ¥ styrebidraget for aa begrense kristisk ...
           raske paadragsendringer
3      paadrags_filter();
4
5      // Summering av alle thrusterbidrag
6      totalbidrag.hhf = styrebidrag.hhf + stampbidrag.hhf + rullbidrag.hhf + ...
           hivbidrag.hhf;
7      ...
8      totalbidrag.vvf = styrebidrag.vvf + stampbidrag.vvf + rullbidrag.vvf + ...
           hivbidrag.vvf;
9
10     // kompensering for motorkarakteristikk
11     thruster_retning_korreksjon();
12     // Kontroll av paadragsmetning, og skalerer i henhold
13     skaler_paadrag();
14     // Kontroll av effektnivaa
15     effekt_kontroll();
16
17     // Konverterer til pulsbredde i heltall
18     thruster_PWM.hhf = (int16_t) (NEUTRAL + gain*totalbidrag.hhf);
19     ...
20     thruster_PWM.vvf = (int16_t) (NEUTRAL + gain*totalbidrag.vvf);
21
22     // oppdaterer diodemonster
23     oppdater_ledlys();
24     // skriver pulsbredden til CCR-register
25     skriv_thruster_PWM();
26
27     // Konverterer thrusterpaadrag til 8-bit, for pilotrapport
28     thrusterdata.hhf = (int8_t) (gain*totalbidrag.hhf/4);
29     ...
30     thrusterdata.vvf = (int8_t) (gain*totalbidrag.vvf/4);
31 }
```

Programmet går altså gjennom følgende steg for å samle opp et totalpådrag som sendes til

motorene:

1. Pådragsfilter: Filtrerer styrebidraget med et IIR filter med filtreringsgrad på 80%, for å begrense veldig raske og store pådragsendringer. Etter noen tester av styringen av ROV-en i vann, ble vi oppmerksomme på at ved kritisk store pådragsendringer kunne den plutselige strømtrekken slå ut sikringen til thrusterne. Spesielt når thrusterne måtte plutselig endre kjøreretning.
2. Totalpådrag: Summerer bidrag fra manuell styring, og hiv-, stamp- og rullregulator.
3. Korreksjon for motorkarakteristikk: Korrigerer pådrag for hver thruster basert på thrusterens rotasjonsretning. Rotorene er mer effektive i den ene retningen, slik at pådraget må korrigeres med modellen av motorkarakteristikken for å kompensere for forskjellig *thrust*. Utviklingen av motorkarakteristikken er dokumentert i testrapporten i vedlegg B.1.
4. Skalering: Kontrollerer at ingen av pådragene overskrider 100%. I så tilfelle skaleres *alle* thrusterene ned slik at det høyeste pådraget blir 100.
5. Effektbegrensning: Demper totalpådraget til alle thrusterne hvis effektforbruket fra thrusterne overskrider anbefalt nivå. Grensen for anbefalt effektnivå er bestemt av gruppen med ansvar for kraftforsyningskortet.
6. PWM-konvertering: Pådragsverdiene blir konvertert fra flyttall til heltalls pulsbredde for PWM-signal. Som tidligere nevnt er pulsbredden for nøytralt pådrag til motorkontrollerne 1500 μs. Pulsbredden blir da 1500 pluss pådraget multiplisert med forsterkningen. Piloten kan bestemme hvor stor denne forsterkningen er, men hvis maks pådrag er tilgjengelig, er denne forsterkningen lik 4. Ved maks pådrag og maks forsterkning har pulsbredden intervall fra 1100 til 1900. Thrusteren roterer i negativ retning ved en pådragsverdi lavere enn 1500.
7. LED-lys: Oppdaterer diodemønsteret basert på pådragsverdiene. Ettersom diodene ikke er synlige fra utsiden av elektronikkhuset benyttes dette i hovedsak for testing.
8. Generere PWM-signal: Verdiene for pulsbreddene skrives direkte inn i *timer*-modulenes *capture-/compare*-register.
9. Utgående data: Konverterer pådragsverdiene og lagrer dem i sin struct, *thrusterdata*, slik at de er tilgjengelige for å rapporteres til piloten.

5.3.4 Kommunikasjon

Kommunikasjonen mellom styringskontrolleren og resten av systemet består av åtte forskjellige meldinger, seks innkommende og to utgående. Kommunikasjonen i ROV-en er designet til å operere på 20 Hz, slik at styringskontrolleren mottar pilotinstruksjoner, målinger fra sensorkortet og strømmålinger hvert 50 ms. Dette har ikke spesielt stor innvirking på styringskontrolleren, siden den i hovedsak opererer reaktivt på meldingene som mottas. Det eneste den aktivt utfører, er kontrollsjekker av eksempelvis kontinuitet i datastrømmen. I tabell 5.3 vises en oversikt over de innkommende CAN-meldingene og innholdet i disse.

Tabell 5.3: Innkommende CAN-meldinger

Meldingsnavn + ID	Variabel	Datatype	Verdiområde	Benevning
Styremelding	jag	int8	-100, 100	prosent
(ID: 70)	svai	int8	-100, 100	prosent
	hiv	int8	-100, 100	prosent
	gir	int8	-100, 100	prosent
	manipulator	int8		bitmønster
	tom	int8	0, 0	(inaktiv)
	tom	int8	0, 0	(inaktiv)
	throttling	int8	0, 100	prosent
Parameter	paramID	uint32	0, -	ID
(ID: 71)	verdi	*float* (32 bit)		
Gyromelding	hiv	int16	0, 5000	cm
(ID: 80)	rull	int16	-1800, 1800	$\frac{grader}{10}$
	stamp	int16	-1800, 1800	$\frac{grader}{10}$
	gir	int16	0, 0	(inaktiv)
Akselerasjon	jag	int16		$\frac{mm}{s^2}$
(ID: 81)	svai	int16		$\frac{mm}{s^2}$
	hiv	int16		$\frac{mm}{s^2}$
	tom	int16	0, 0	(inaktiv)
Effektbruk	thrustere	uint16	0, 13000	$\frac{W}{10}$
(ID: 90)	manipulator	uint16	0, 2400	$\frac{W}{10}$
	5v	uint16	0, 650	$\frac{W}{10}$
	tom	uint16		
Ping			Ping	
(ID: 95)				

I tabell 5.4 vises en oversikt over de utgående CAN-meldingene og innholdet i disse.

Tabell 5.4: Utgående CAN-meldinger

Meldingsnavn + ID	Variabel	Datatype	Verdiområde	Benevning
Thrusterdata	HHF	int8	-100, 100	prosent
(ID: 170)	HHB	int8	-100, 100	prosent
	HVB	int8	-100, 100	prosent
	HVF	int8	-100, 100	prosent
	VHF	int8	-100, 100	prosent
	VHB	int8	-100, 100	prosent
	VVB	int8	-100, 100	prosent
	VVF	int8	-100, 100	prosent
Pong			Pong	
(ID: 189)				

5.4 Konklusjon

Mikrokontrolleren på kretskortet er programmert for å behandle pilotinstruksjoner og målinger fra ROV-ens sensorer. Datapakkene kommuniseres over CAN-buss hvert 50 ms. Hovedoppga-

vene er å styre ROV-en i vannet, drive manipulatorarmen og å automatisk regulere at den står stabilt i vannet. I tillegg til å regne ut hvor stort pådrag som skal settes til hver enkelt thruster, blir pådragsverdiene kontrollert og behandlet av flere smarte funksjoner. Funksjonene tar hensyn til effektbruk, motorkarakteristikk og fordelingen av pådraget fra de forskjellige bidragskildene. Kontrolleren genererer PWM-signal til motorkontrollerne og sender pådragsdata tilbake til toppsiden. Programvaren som er utviklet, gjør det mulig å implementere styrings- og reguleringssystemet på ROV-en.

Den fullstendige koden brukt på mikrokontrolleren kan lastes ned via linken i vedlegg A.

Kapittel 6

Styring

Kapitteloversikt

Hensikten med kapittelet er å utvikle ROV-ens manuelle styringssystem, og derfor tar kapittelet for seg alt som omfatter manuell styring av ROV-en. For å styre den, må man først fastslå noen definisjoner på ROV-ens orientering. Videre presenteres kontrollerdataen mottatt fra toppsiden, og pådragsutregning basert på disse. Til slutt er det vist hvordan styresystemet for både thrustere og manipulatormotorer implementeres på mikrokontrolleren som er tatt i bruk.

6.1 Orientering

Styresystemet skal baseres på styredata fra toppsiden. Før styredata fra kontrolleren kan behandles og regnes om til thrusterpådrag, må man definere ROV-ens orientering i vannet. Dette innebærer kartlegging av bevegelsesretninger, valg av koordinatystem ROV-en skal bevege seg relativt til og navngivning av de ulike thrusterene. En tydelig definisjon på orientering vil gjøre arbeidet med å manøvrere ROV-en etter ønsket oppførsel betydelig enklere.

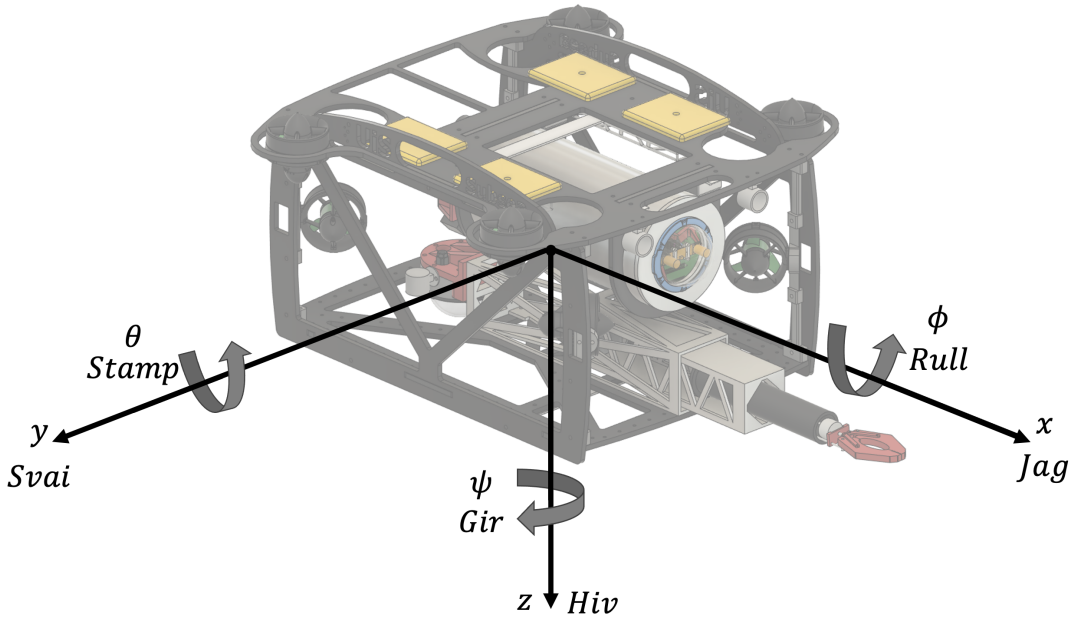
6.1.1 Frihetsgrader

Man definerer vanligvis at farkosters bevegelse foregår i seks frihetsgrader [12]. De tre første frihetsgradene utgjør translatorisk bevegelse langs tre akser i rommet, mens de tre siste utgjør rotasjon om de samme aksene. Under vises en tabell med oversikt over de ulike frihetsgradene og tilhørende navn.

Tabell 6.1: Farkosters frihetsgrader av bevegelse.

Frihetsgrad	Bevegelse	Navn	Posisjon
1	Translasjon langs x-aksen	Jag	x [m]
2	Translasjon langs y-aksen	Svai	y [m]
3	Translasjon langs z-aksen	Hiv	z [m]
4	Rotasjon om x-aksen	Rull	ϕ [rad]
5	Rotasjon om y-aksen	Stamp	θ [rad]
6	Rotasjon om z-aksen	Gir	ψ [rad]

Figuren under viser en grafisk fremstilling av de seks frihetsgradene.

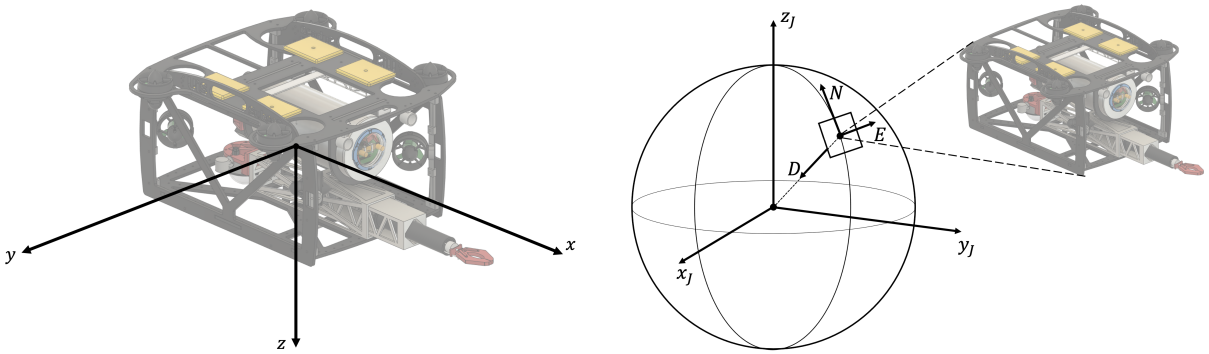


Figur 6.1: ROV-en sine frihetsgrader i vann. Bevegelse langs aksene omtales som translasjon, og dreiing om aksene omtales som rotasjon.

Da undervannsfarkosten er ment for å dykke under vann, definerer man gjerne positiv bevegelse i hiv-retning nedover. Positiv jag-retning defineres til å være fremover, ofte omtalt som kjøreretning. For å oppnå et standard høyrehåndssystem blir positiv svai-bevegelse til høyre sett relativt til kjøreretningen.

6.1.2 Koordinatsystem

For å beskrive ROV-ens bevegelse og posisjon trengs ett eller flere koordinatsystemer. Thor Inge Fossen definerer i boka si, *Marine Control Systems* [12], flere koordinatsystemer som er vanlige å bruke for marine farkoster. For å beskrive ROV-ens bevegelsesretning, har vi valgt å ta i bruk et *Body Fixed Frame* (BFF), som er vist i figur 6.2a.



(a) Koordinatsystemet *Body Fixed Frame* (BFF). Her er koordinatsystemet fastlåst til ROV-en. X-aksen er definert i fremoverretning, y-aksen til høyre relativt til fremoverretning og z-aksen nedover.

(b) Koordinatsystemet *North East Down* (NED). Koordinatsystemet er gitt av to akser i jordens kompassretninger nord og øst, og én akse pekende nedover mot jordens sentrum. Dette gjelder uansett hvor man befinner seg på jordoverflaten.

Figur 6.2

BFF er et koordinatsystem som er festet til ROV-en, og følger den under bevegelse. Aksene defineres på samme måte som positive retninger av frihetsgradene til ROV-en. Dette koordinatsystemet er derfor passende å bruke for å vurdere bevegelse i hver frihetsgrad. Origo settes til å være i ROV-ens massesentrum. ROV-ens massesentrum defineres i kapittel 7 til å også utgjøre rotasjonssentrumet av ROV-en. Ved å sette origo av koordinatsystemet her, kan man vurdere rotasjon i hver frihetsgrad uten å måtte ta hensyn til at rotasjonspunktet beveger seg i forhold til koordinatsystemet.

Koordinatsystemet BFF kan brukes til å beskrive ROV-ens bevegelse, men ikke til å beskrive posisjon i forhold til omgivelsene. I utgangspunktet trengs posisjonsbeskrivelse av ROV-en kun til regulering. Da den skal operere i et basseng, har vi valgt å bortprioritere data på ROV-ens posisjon i noen frihetsgrader. Sensorgruppen har også indikert at det vil være vanskelig å skaffe sensorer som kan måle ROV-ens posisjon i hver av dem. ROV-en skal derfor kun reguleres i dybde og stabilitet, som tilsvarer frihetsgradene hiv, rull og stamp. Koordinatsystemet må derfor bestemmes basert på dette.

For å beskrive posisjonen av en farkost, kan det være nyttig å bruke et koordinatsystem av typen *North East Down* (NED), vist i figur 6.2b. Dette er et koordinatsystem med akser pekende langs jordens magnetiske nord og øst, samt en akse pekende mot jordens sentrum fra jordoverflaten. Aksene pekende mot nord og øst vil ikke være relevante for å beskrive ROV-ens posisjon i vårt tilfelle, men koordinatsystemet velges på grunn av aksen som peker mot jordens sentrum. Regulering av hiv-bevegelse kan gjøres direkte basert på ROV-ens posisjonen langs denne aksen. Posisjon i rull- og stamp-retning må beskrives ut fra hvordan koordinatsystemet BFF beveger seg i forhold til NED. Rull-posisjonen beskrives med z-aksens (BFF) utslag i xz-planet i forhold til D-aksen (NED), og stamp-posisjonen beskrives med z-aksens utslag i yz-planet i forhold til D-aksen. Alle tre frihetsgradene er mulige å måle ved hjelp av sensorer. Det brukes en trykkmåler

for dybdeposisjonen, og et gyroskop kombinert med et DC-akselerometer for å måle rull- og stamp-vinkel. Vinklene måles basert på tyngdeakselerasjonen, som til vår fordel virker langs D-aksen i NED.

6.1.3 Motornavn

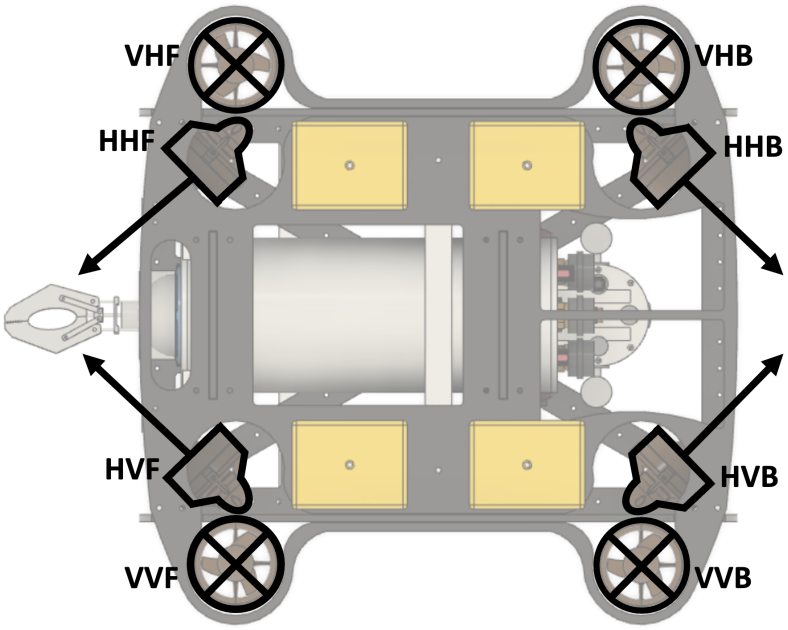
Hver enkelt thruster bidrar til bevegelse kun i gitte frihetsgrader, alt etter hvordan den er plassert på ROV-en. Da ROV-en er utstyrt med åtte thrustere, vil det være fordelaktig å navngi hver av dem med et kort, men beskrivende, navn. Dette bidrar til god oversikt når man senere skal gjøre pådragsberegninger basert på de ulike thrusterenes bidrag i hver bevegelsesretning. Forkortede navn vil også være spesielt hensiktsmessig for implementasjon av styring i kode, da det vil gjøre koden mer kompakt.

Thrusterne navngis basert på hvilken kraftretning de bidrar i, og plassering på ROV-en. Kraftretning deles inn i vertikalt og horisontalt bidrag, med fire thrustere i hver kategori. Plasseringen defineres ut fra om thrusteren befinner seg på venstre eller høyre side relativt til kjørerenting, og om den befinner seg fremme eller bak på ROV-en. Tabell 6.2 viser en oversikt over hver thruster med tilhørende navn. Thrusternummeret er basert på plasseringene i forhold til BFF-koordinatsystemets oktanter.

Tabell 6.2: Thrusterenes orientering, og tilhørende navn.

Thruster	Orientering	Forkortet navn
1	Horisontal høyre fremme	HHF
2	Horisontal høyre bak	HHB
3	Horisontal venstre bak	HVB
4	Horisontal venstre fremme	HVF
5	Vertikal høyre fremme	VHF
6	Vertikal høyre bak	VHB
7	Vertikal venstre bak	VVB
8	Vertikal venstre fremme	VVF

Thrusterne med tilhørende navn er også fremvist i figur 6.3.



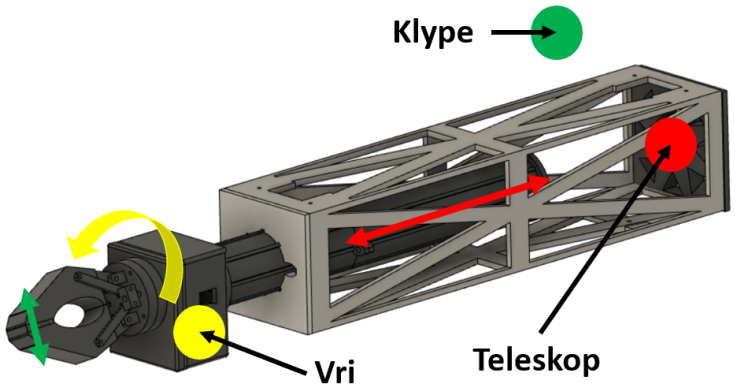
Figur 6.3: Navnene motorene omtales med, defineres ut fra orientering og plassering på ROV-en. Det skilles mellom vertikalt og horisontalt kraftbidrag, plassering på venstre eller høyre side og plassering bak eller fremme.

Manipulatorarmen påvirker ikke ROV-ens posisjon og bevegelse direkte. Derfor vil manipulatormotorene kun navngis basert på hvilken funksjon de har. Manipulatorarmen er produsert med mulighet for klyping og vridning av klypen, samt en teleskopfunksjon for å rekke lenger. Da ingen av oppgavene i konkurransen krever spesiell rask håndtering av manipulatorarmen, har prosjektledelsen bestemt at det også skal settes prioritet på manipulatormotorene, slik at kun én av dem kan kjøres om gangen. Dette er implementert med tanke på effektbegrensing. Manipulatoren drives som nevnt av tre BLDC-motorer, som hver kan trekke opp til 180 W, og strømforsyningskretsen disse motorene er koblet til har en øvre tilførsel på 230 W. Det å kjøre kun én om gangen er derfor enkleste løsning. Tabellen under viser en oversikt over manipulatormotorene med tilhørende navn og prioritet.

Tabell 6.3: Oversikt over de tre manipulatormotorene med tilhørende prioritet og navn.

Funksjon	Prioritet	Navn
Gripe og slippe med klypen	1	Klype
Vridning av klypen i begge retninger	2	Vri
Bevege armen ut og inn	3	Teleskop

Figur 6.4 viser hvordan manipulatormotorene med tilhørende navn er plassert. Pilene viser bevegelsesretningene av de ulike leddene på manipulatoren, og sirklene viser til tilhørende motorenes plassering.



Figur 6.4: Navnene motorene omtales med, defineres ut fra hvilken funksjon de har for manipulatorarmen.

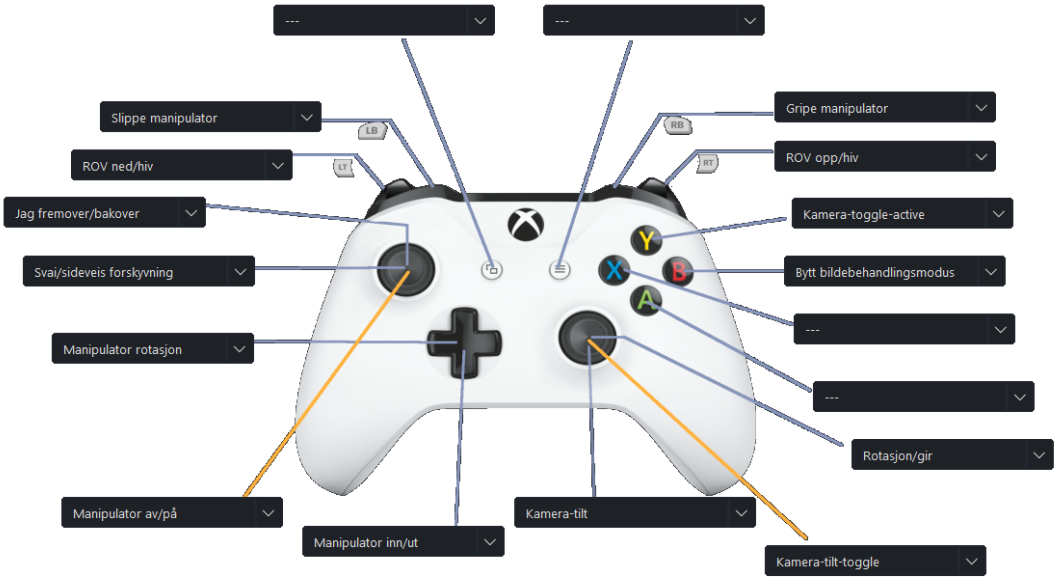
Teleskopmotoren er plassert bakerst på armen. Denne er koblet til et gjengestag, som flytter armen inn og ut når det roterer. Motoren for vridning er montert inne i huset bak klypen, og er koblet til et ormegir for å overføre rotasjonen 90°. Motoren for klypefunksjon er plassert eksternt fra selve manipulatorarmen. Motoren skal være koblet til en vaier som kan strammes når motoren roterer. Oppstramming av vaieren vil lukke klypen, og fjærer vil åpne klypen igjen når vaieren slippes opp.

6.2 Pådragsutregning

Pådragsregning gjøres basert på styredata fra kontrolleren som brukes på toppsiden. Derfor må man kartlegge hvilke verdier man får som styredata, og hvordan disse skal regnes om til et pådrag. Dette må gjøres for alle frihetsgrader som skal styres manuelt med thrustere. For manipulatormotorene vil dette være enklere, da de kun styres med et på/av-signal i begge retninger. Til slutt må bidraget fra hver frihetsgrad summeres for hver thruster, og skaleres etter thrusteren med høyest pådragsverdi. Dette ble nevnt i kapittel 5, og vil bli nærmere gjennomgått i de kommende delkapitlene.

6.2.1 Kontrollerdata

Operatørgrensesnittgruppen har hatt hovedansvaret for kontrolleren som brukes til styring av ROV-en. Funksjonskartlegging av de ulike knappene og stikkene på kontrolleren er gjort i samarbeid med gruppen vår, og prosjektledelsen. Kontrolleren som brukes i år er av typen *Xbox One*, og figur 6.5 viser en oversikt over funksjonene kontrolleren skal ha.



Figur 6.5: Utsnitt fra *GUI* på toppsiden. Figuren viser funksjonskartlegging av den planlagte kontrolleren.

Knappene som gir verdier for bevegelse i frihetsgradene jag, svai, hiv og gir, er relevante for den manuelle styringen av thrustere. Disse er valgt til å være analoge knapper, som gir ut verdier basert på fysisk utslag av knappen. Knappene som styrer manipulatormotorene er digitale, og gir kun et på/av-signal. Hver manipulatormotor trenger derfor en knapp for begge bevegelsesretninger. Tabellen under viser en oversikt over knappene med tilhørende verdiområde.

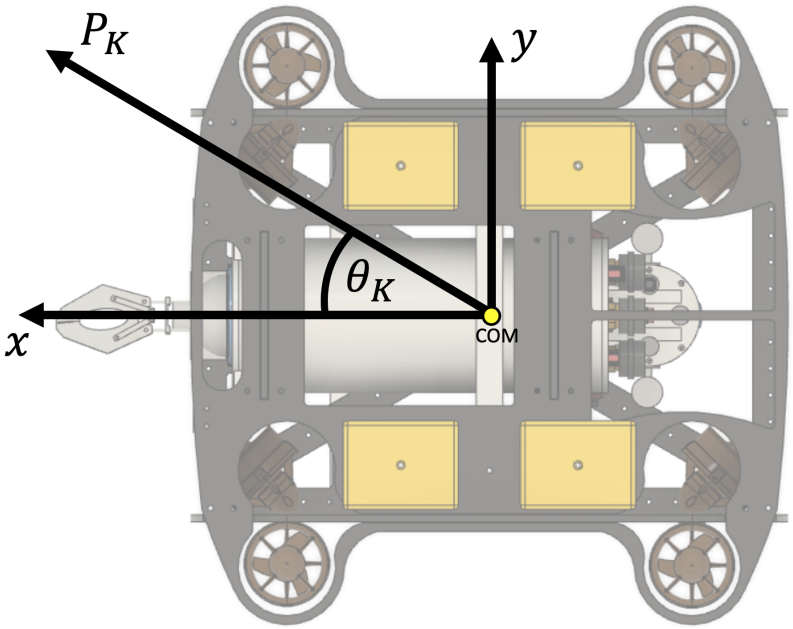
Tabell 6.4: Verdiområder for knappene som utgjør den manuelle styringen av motorene.

Frihetsgrad	Knapp	Verdiområde
Jag	Venstre styrestikke, frem/bak	[-100, 100]
Svai	Venstre styrestikke, sideveis	[-100, 100]
Hiv ned	Venstre *trigger* (LT)	[0, 100]
Hiv opp	Høyre *trigger* (RT)	[-100, 0]
Gir	Høyre styrestikke, sideveis	[-100, 100]
Gripe klype	Høyre *bumper* (RB)	[0, 1]
Slippe klype	Venstre *bumper* (LB)	[0, 1]
Vri klype med klokken	*D-pad*, høyre	[0, 1]
Vri klype mot klokken	*D-pad*, venstre	[0, 1]
Teleskop ut	*D-pad*, opp	[0, 1]
Teleskop inn	*D-pad*, ned	[0, 1]

Verdiene som gir thrusterbidrag er bestemt til å være mellom -100 og 100 for å enkelt kunne regne om styredataen til prosentvis pådrag i positiv og negativ retning. Spesielt for en kontroller av typen *Xbox One*, er at styrestikkene begrenses av deres fysiske utslagsområde. Med dette menes at verdiområdet for jag og svai kombinert ligger innenfor en sirkel med radius 100. Det betyr eksempelvis at både jag og svai ikke kan være 100 samtidig. Dette vil være fordelaktig for videre beregning av pådrag under horisontal translasjon, og grunnen er forklart i neste delkapittel. For manipulatormotorene gis det kun et på/av-signal, og størrelsen av pådraget vil bli bestemt basert på fysisk testing av armen. Det vil derfor ikke bli gjennomgått videre pådragsberegning for manipulatormotorene.

6.2.2 Horisontal translasjon

Bevegelse i jag- og svai-retning blir påvirket samtidig av alle horisontale thrustere. Det at venstre styrestikke gir verdier for begge frihetsgradene, gjør at vi kan regne både en retningsvektor, og en retningsvinkel direkte ut fra styredataen. Retningsvektoren P_K, og retningsvinkelen θ_K er illustrert i figuren under.



Figur 6.6: Det defineres en kjøreretning med en vinkel θ_K fra x-aksen. Vinkelutslag mot y-aksen defineres som positiv, og bort fra y-asken som negativ.

Verdiene av P_K og θ_K kan regnes direkte ut fra styredataen på venstre styrestikke. Fordelen med at verdiområdet til jag og svai kombinert ligger innenfor en sirkel, er at størrelsen av P_K aldri overgår 100. P_K kan vi derfor bruke som en referanse på hvor stort maksimalt pådrag for hver thruster skal være. Denne størrelsen regnes på følgende måte, med X_{VS} definert som verdien venstre styrestikke gir ut i x-retning, og Y_{VS} i y-retning:

$$P_K = \sqrt{(X_{VS})^2 + (Y_{VS})^2} \tag{6.1}$$

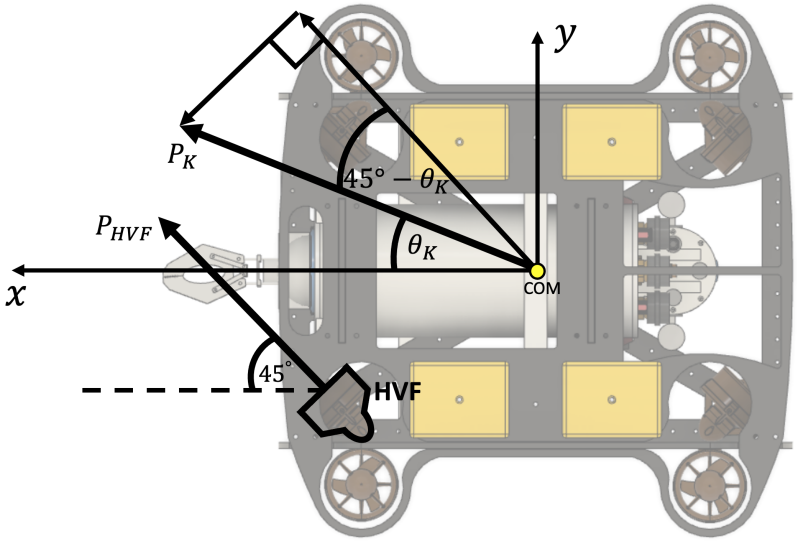
Videre brukes operatoren *atan*2, eller *dobbel arcus tangens*, for å finne retningsvinkelen. Dette er for å få en vinkel som går mellom -180° og 180°. Det vil være enklere å regne med en retningsvinkel i positiv og negativ retning, enn en vinkel som hopper fra 0° til 360° når man slår om kjøringen fra høyre mot venstre.

$$\theta_K = atan2(Y_{VS}, X_{VS}) \tag{6.2}$$

Under videre pådragsregning er det hentet inspirasjon fra Morten Tengesdals dokument, *Litt om utrekning av motorpådrag for ein ROV* [48].

P_K brukes i hovedsak som en maksgrense på hvor stort prosentvis pådrag en enkelt thruster skal

påtrykkes. Likevel må pådraget på de enkelte thrusterne skaleres ut fra vinkelen de er plassert på ROV-en relativt til θ_K. For å illustrere hvordan dette kan gjøres, viser figuren under en dekomponering av retningsvektoren P_K normalt på, og langs, retningen til thruster HVF.



Figur 6.7: Retningsvektoren er dekomponert i en parallell og en normal komponent i retningen thruster HVF er plassert.

Dersom retningsvektoren P_K går parallelt med P_{HVF}, skal thrusteren virke med samme pådrag som P_K. Dersom de står normalt på hverandre, skal ikke thrusteren gi pådrag. Ut fra dette, og ut fra illustrasjonen i figur 6.7, kan pådraget for thruster HVF beregnes:

$$P_{hvf} = P_K \cdot cos(45° - \theta_K) \tag{6.3}$$

Det samme prinsippet gjelder også for de andre thrusterene. I tillegg kan det nevnes at diagonalt motstående thrustere virker i samme retning, men med motsatt fortegn, og resultatet er at det absolutte pådraget blir det samme. Dette kan også vises algebraisk:

$$P_{hvb} = P_K \cdot cos(135^o - \theta_K) = -P_K \cdot cos(135^o - \theta_K - 180^o) = -P_K \cdot cos(-45^o - \theta_K) = -P_{hhf}$$

$$P_{hvf} = P_K \cdot cos(45^o - \theta_K) = -P_K \cdot cos(45^o - \theta_K - 180^o) = -P_K \cdot cos(-135^o - \theta_K) = -P_{hhb}$$

Oppsummert blir derfor pådraget til de horisontale thrusterene som følge av horisontal translasjon:

$$\begin{aligned} P_{hhf} &= P_K \cdot cos(-45^o - \theta_K) \\ P_{hhb} &= P_K \cdot cos(-135^o - \theta_K) \\ P_{hvb} &= -P_{hhf} \\ P_{hvf} &= -P_{hhb} \end{aligned} \tag{6.4}$$

Det kan videre være interessant å se på hva det maksimale pådraget i kjøreretning kan bli. Dersom man dekomponerer hver kraftvektor fra thrusterene i forhold til P_K, kan man videre summere opp bidragene i kjøreretning fra hver thruster:

$$\begin{aligned} P_{\text{tot}} = &\underbrace{P_{hhf} \cdot cos(-45^o - \theta_K)}_{Bidrag\ hhf} + \underbrace{P_{hhb} \cdot cos(-135^o - \theta_K)}_{Bidrag\ hhb} \\ &+ \underbrace{P_{hvb} \cdot cos(135^o - \theta_K)}_{Bidrag\ hvb} + \underbrace{P_{hvf} \cdot cos(45^o - \theta_K)}_{Bidrag\ hvf} \end{aligned} \tag{6.5}$$

Dersom man setter verdiene for alle thrustere fra ligning 6.4 inn i ligning 6.5 viser det seg at det maksimale pådraget ikke avhenger av retningsvinkelen, og at det alltid er lik det dobbelte av pådragsvektoren P_K.

$$P_{tot} = 2 \cdot P_K \tag{6.6}$$

Da det totale pådraget i kjøreretning er dobbelt så stort som retningsvektoren P_K, vil den maksimale thrusten i kjøreretning være lik maksimal thrust fra to av thrusterne uavhengig av hvilken retning man kjører. Ser man derimot på den absolutte pådragsbruken til thrusterene samlet, vil denne variere med hvilken vinkel man kjører i. Den absolutte pådragsbruken er gitt som:

$$P_T = |P_{hhf}| + |P_{hhb}| + |P_{hvb}| + |P_{hvf}| \tag{6.7}$$

Minimal pådragsbruk får man i retningene -135°, -45°, 45° og 135°. Dette kan verifiseres ved å legge inn disse vinklene i ligning 6.4. I dette tilfellet blir $P_T = 2P_K$. Maksimal pådragsbruk får man i retningene -90°, 0°, 90° og 180°, der $P_T = 2\sqrt{2}P_K$. Det dette betyr for systemet, er at for å få samme totale kraftbidrag fra thrusterene i alle retninger, kreves det ulik total pådragsbruk. De horisontale thrusterene er mer effektive når ROV-en eksempelvis kjører i 45° enn i 0°.

6.2.3 Gir-rotasjon

Gir-rotasjon utføres også ved hjelp av de horisontale thrusterene. Pådraget som sendes til thrusterene blir derfor et tillegg til pådraget fra horisontal translasjon. Da giring styres uavhengig av andre frihetsgrader kan pådraget på thrusterene bestemmes direkte fra styredataen. Som vist i tabell 6.4 styres giring av sideveis bevegelse med høyre styrestikke. Verdiområdet på denne er fra -100 til 100, og thrusterpådraget kan derfor bestemmes direkte fra denne verdien. Man trenger kun å ta hensyn til rotasjonsretning. Når verdien fra styrestikken, X_{HS}, er negativ, skal ROV-en rotere mot klokken. Positiv X_{HS} skal føre til rotasjon med klokken. Ut fra oversikten over orientering i figur 6.3 kan man videre bestemme retningen på pådragene:

$$
\begin{aligned}
P_{hhf} &= -X_{HS} \\
P_{hhb} &= X_{HS} \\
P_{hvb} &= -X_{HS} \\
P_{hvf} &= X_{HS}
\end{aligned} \tag{6.8}
$$

Disse pådragene vil summeres med bidragene fra horisontal translasjon. Til slutt vil pådraget som beskrevet i delkapittel 5.3.3 skaleres mot motorkarakteristikk, totalpådrag og en begrensende parameter.

6.2.4 Vertikal translasjon

Under vertikal translasjon kreves det heller ingen avanserte beregninger. De vertikale thrusterene står parallelt med bevegelsesretningen, og kan derfor få pådrag direkte fra verdien i styredataen. For å unngå utslag i rull- eller stamp-vinkel er det derimot viktig at alle thrustere bidrar i samme retning, og med samme kraftbidrag. Totalt prosentvis pådrag på hver thruster får man ved å summere verdien fra *LT*- og *RT*-knappen. Da positiv hiv-retning er nedover, er knappen LT, som skal styre ROV-en nedover, tilegnet positivt verdiområde, og RT, som styrer ROV-en oppover, er tilegnet negativt verdiområde.

$$
\begin{aligned}
P_{vhf} &= LT + RT \\
P_{vhb} &= LT + RT \\
P_{vvb} &= LT + RT \\
P_{vvf} &= LT + RT
\end{aligned} \tag{6.9}
$$

Denne summeringen skjer egentlig på toppsiden, og den mottatte styremeldingen for hiv-pådrag i mikrokontrolleren inneholder kun én verdi mellom -100 og 100. Denne brukes altså direkte som pådragsverdi for de vertikale thrusterene. Ligningene over er likevel presentert for å vise hvordan signalbehandlingen foregår.

Dette er eneste bidrag fra manuell styring på de vertikale thrusterene. Disse pådragene summeres med bidragene fra regulatorene som senere skal utvikles. Som nevnt i delkapittel 5.3.2 vil et bidrag fra manuell styring i vertikal translasjon deaktivere regulatoren for hiv, slik at det ikke oppstår konflikter. Til slutt skaleres pådragene på samme måte som for de horisontale thrusterene, og effektbegrensning tas høyde for som beskrevet i kapittel 5.

6.3 Implementering av thrusterstyring

Koden som behandler pilotinstruksjonene for manøvrering er en funksjon som er kalt *oppdater_styrebidrag()*. Funksjonen beregner det første elementet i samlingen av fire thrusterbidrag, *styrebidrag, rullbidrag, stampbidrag og hivbidrag*, som til sammen utgjør *totalpaadrag*. Koden til funksjonen er vist nedenfor i kodeutsnitt 6.1. Pilotinstruksjonene ligger i styremeldingen som

sendes fra toppsiden. Meldingen inneholder 8 byte med instruksjoner, hvor hver byte er en egen variabel: jag, svai, hiv, gir, manipulator, tom, tom, *throttling*. De to tomme variablene er *place holders* for instruksjoner for rull og stamp, men er ikke implementert ettersom ROV-en både har aktiv og passiv stabilisering i disse frihetsgradene. Den er altså utviklet til å alltid ligge horisontalt i vannet. Manipulatorvariabelen er instruksjoner til gripearmen basert på bitmønsteret til de 8 bitene, i stedet for *verdien* til variabelen, og er forklart i detalj i delkapittel 6.4. *Throttling* er en variabel piloten kan bestemme for å begrense maks pådrag til thrusterne. Variabelen representerer prosentvis demping av maks pådrag.

Kode 6.1: Funksjonen oppdater_styrebidrag() som behandler navigatoriske oppgaver.

```
1
2  void oppdater_styrebidrag(void){
3      // Regner ut argument og modulus av jag og svai
4      styreretning = atan2(styremelding.svai, styremelding.jag);
5      maks_paadrag = sqrt(styremelding.jag*styremelding.jag + ...
           styremelding.svai*styremelding.svai);
6
7      // Utregning av paadragsnivaa til hver horisontale thruster
8      styrebidrag.hhf = cos(HR_kompass_rad.NW - styreretning)*maks_paadrag;
9      styrebidrag.hhb = cos(HR_kompass_rad.SW - styreretning)*maks_paadrag;
10     styrebidrag.hvb = - styrebidrag.hhf;            // thrusteren er ...
           orientert 180 grader paa hhf
11     styrebidrag.hvf = - styrebidrag.hhb;            // thrusteren er ...
           orientert 180 grader paa hhb
12
13     // Addisjon av gir-rotasjon
14     if (styremelding.gir){
15         gir = styremelding.gir * param.demping.gir;
16         styrebidrag.hhf -= gir;
17         styrebidrag.hhb += gir;
18         styrebidrag.hvb -= gir;
19         styrebidrag.hvf += gir;
20     }
21
22     // Bidrag til vertikale thrustere i tilfelle hiv
23     if (styremelding.hiv){
24         flagg.hiv_pause = 1;
25         flagg.hpid = 0;
26         teller_reg_hiv = 0;
27         hiv = styremelding.hiv * param.demping.hiv;
28         styrebidrag.vhf = hiv;
29         styrebidrag.vhb = hiv;
30         styrebidrag.vvb = hiv;
31         styrebidrag.vvf = hiv;
32     }
33     else{
34         styrebidrag.vhf = 0;
35         styrebidrag.vhb = 0;
36         styrebidrag.vvb = 0;
37         styrebidrag.vvf = 0;
38     }
39
40     // Paadragsbegresning
41     if (styremelding.throttling){gain = (100-styremelding.throttling)/25;}
42     else{                       gain = 4;}
43
44 }
```

Styringsberegningen starter med å regne ut vinkelen ROV-en skal bevege seg i relativt til sitt eget

koordinatsystem, BFF, og å regne ut hvor stort pådraget i den gitte retningen skal være relativt til maksimalt pådrag. Disse utregningene ble utledet i ligning 6.1 og 6.2. Dette er med andre ord en trigonometrisk omgjøring fra kartesisk- til polarkoordinat med argument og modulus. Argumentet blir videre brukt til å bestemme hver enkelt thrusters relevans i styreretningen. Thrusterne som står parallelt med styreretning vil ha 100% effektivitet, men en thruster som står normalt på styreretning vil ha 0% bidrag i ønsket retning. De horisontale thrusterne er parvis parallelle og har derfor lik effektivitet, men siden de er montert i motsatt retning, vil pådragene ha motsatt fortegn.

Styring av hiv og gir krever ingen omregning, men er en flat mengde fordelt utover alle thrusterene som arbeider i ønsket retning. For gir vil dette si at alle horisontale thrustere får like stort pådrag, men hvor to gis negativt pådrag slik at alle arbeider i samme retning. For hiv blir pådraget fordelt likt på alle vertikale thrustere. Både gir- og hivbidraget kan dempes med hver sin dempingsparameter. Disse kan bestemmes av piloten for å gi ønsket sensitivitet.

Funksjonen sjekker om styringsmeldingen har hiv- eller girinstrukser før den beregner disse. I tillegg setter hiv-biten av koden et flagg, *flagg.hiv_pause*, som forteller hovedprogrammet at PID-regulatoren for hiv skal settes på pause inntil ny ønsket dybde er nådd. PID-regulatoren startes opp igjen av en teller i *SysTick*, forklart i delkapittel 5.3.2. Pådraget til de vertikale thrusterne blir også satt til null hvis styremeldingen ikke inneholder hivinstruksjoner, slik at gamle pådrag ikke henger igjen.

6.4 Implementering av manipulatorstyring

I motsetning til den navigatoriske styringen, er manipulatoren styrt av bitmønsteret i den 8 bit lange meldingen. Grunnen for valget av å bruke bitmønster er først og fremst at manipulatoren styres med på/av-signaler. Bitmønsteret må reflektere tre tilstander for hver av de bidireksjonale motorene: rotering mot klokken, nøytral og rotering med klokken. Dette kan løses med 2 bit for hver motor, totalt seks bit. Et av de to resterende bitene blir brukt for å bestemme om manipulatoren er på eller av, og det siste bitet er ledig, men er tenkt å være en *place holder* hvis det skulle oppstå behov for en *reset*-funksjon eller lignende. Styring basert på bitmønster gjør også instruksjonspakken kompakt, og frigjør derfor plass i styringsmeldingen.

En funksjon som bruker bitmønster for styring med prioriteringslogikk kan være tung å lese, og ikke minst vanskelig å feilsøke. For å gjøre den mer oversiktlig er det opprettet masker for de relevante bitmønstrene og pulsbredden til nøytralt pådrag, vist i tabellen under:

Tabell 6.5: Masker og bitmønster

Maske	**Bitmønster**	Forklaring
MAN_ON	0b'0100'0000	Manipulatoren er på
MAN_KLYPE	0b'0011'0000	Manipulatorklypa
MAN_VRI	0b'0000'1100	Manipulatorrotasjon
MAN_TELESKOP	0b'0000'0011	Manipulatortranslasjon
MAN_KLYPE_LUKKE	0b'0010'0000	Manipulatorklypa lukkes
MAN_VRI_CCW	0b'0000'1000	Manipulatorrotasjon mot klokken
MAN_TELESKOP_UT	0b'0000'0010	Manipulatortranslasjon utover
NEUTRAL	1499	Tilsvarer null pådrag til motorene

Det er ikke satt opp egne masker for begge rotasjonsretningene av manipulatormotorene. Styremeldingen vil først sammenlignes med en maske som sjekker om manipulatormotoren skal aktiveres, for eksempel *MAN_KLYPE*, og deretter med en maske som sjekker om den skal rotere i en gitt retning, for eksempel *MAN_KLYPE_LUKKE*. Dersom masken som bestemmer retningen ikke gir utslag, vil motoren settes til å rotere den andre retningen. I kodeutsnitt 6.2 vises funksjonen som realiserer manipulatorstyringen.

Kode 6.2: Funksjon for manipulatorstyring.

```
1
2  v, labeloid={} manipulator_styring(void){
3      uint8_t COM = styremelding.manipulator; // bitmoensteret fra pilotinstruksen
4      if (COM & MAN_ON){
5          // foerste prioritet: gripeklo
6          if(COM & MAN_KLYPE){              // manipulatorklo instruksjon
7              if(COM & MAN_KLYPE_LUKKE){  manipulator_PWM.klype = ...
                   NEUTRAL+(param.demping.klype*400);} // Lukk kloa
8              else{                        manipulator_PWM.klype = ...
                   NEUTRAL-(param.demping.klype*400);} // Aapne kloa
9              manipulator_PWM.vri =       NEUTRAL;
10             manipulator_PWM.teleskop =  NEUTRAL;
11         }
12         // andre prioritet: rotasjon av kloen
13         else if(COM & MAN_VRI){          // manipulator roteringinstruks
14             if(COM & MAN_VRI_CCW){       manipulator_PWM.vri = ...
                   NEUTRAL-(param.demping.vri*400);}     // kloa roterer mot klokka
15             else{                        manipulator_PWM.vri = ...
                   NEUTRAL+(param.demping.vri*400);}     // kloa roterer med klokka
16             manipulator_PWM.klype =     NEUTRAL;
17             manipulator_PWM.teleskop =  NEUTRAL;
18         }
19         // tredje prioritet:
20         else if(COM & MAN_TELESKOP){     // manipulator teleskopinstruks
21             if(COM & MAN_TELESKOP_UT){  manipulator_PWM.teleskop = ...
                   NEUTRAL+(param.demping.teleskop*400);}       // manipulatoren ...
                   gaar ut
22             else{                        manipulator_PWM.teleskop = ...
                   NEUTRAL-(param.demping.teleskop*400);}       // manipulatoren ...
                   gaar inn
23             manipulator_PWM.klype =     NEUTRAL;
24             manipulator_PWM.vri =       NEUTRAL;
25         }
26         else{
27             manipulator_PWM.klype =     NEUTRAL;
28             manipulator_PWM.vri =       NEUTRAL;
29             manipulator_PWM.teleskop =  NEUTRAL;
30         }
31     }
32     else{
33         // Setter motorene i NEUTRAL hvis ingen aktiv instruks er mottatt
34         memcpy(&manipulator_PWM, &oppstart_PWM, sizeof(manipulator_sett));
35     }
36     skriv_manipulator_PWM(); //Pwm-signalene blir sendt til motorkontrollerne
37 }
```

Funksjonen begynner med å opprette en lokal variabel, *COM*, hvor manipulatorinstruksen lagres. Funksjonen sammenligner *COM* med de forskjellige maskene for å avgjøre hva som skal utføres. Først sjekker funksjonen om manipulatoren er aktiv, *MAN_ON*, slik at funksjonen ignoreres hvis piloten har deaktivert manipulatoren. Neste sjekk er hvilken motor som skal drives. Hvis en av

bitene i *MAN_KLYPE* er satt i *COM*, sjekker funksjonen hvilken retning motoren skal drives, og setter de andre to motorene til *NEUTRAL*. Hvis ikke bitene i *MAN_KLYPE* var satt i *COM* hopper funksjonen til andre prioritet, *MAN_VRI*, og sjekker instruksjonene på tilsvarende måte. Motorpådragene til hver motor kan dempes individuelt, og kan endres av piloten ved å oppdatere den aktuelle parameteren. Til slutt skrives pulsbredden til sine tilhørende *Capture/Compare*-register.

6.5 Konklusjon

Styresystemet som er utviklet skal sørge for at ROV-en kan manøvreres manuelt. Ved å definere orienteringen av ROV-en, er det lagt til rette for å kunne beskrive ROV-ens bevegelse, og å kunne henvise til de ulike motorene med navn. Videre er det gjennomført pådragsberegning basert på kontrollerdata for manuell styring av frihetsgradene jag, svai, gir og hiv, samt for styring av manipulatorarmen. Styresystemet har blitt testet tørt på elektrolaben, og er dokumentert i en testrapport i vedlegg B.3. Her er det verifisert at pådragsberegningene er implementert riktig på mikrokontrolleren, og gir ut ønskede PWM-signaler. Styresystemet er også testet fysisk med ROV-en i et basseng. Under testing av reguleringen, dokumentert i delkapittel 8.5, ble det også verifisert at styresystemet fungerer etter sin hensikt.

Kapittel 7

Matematiske modeller

Kapitteloversikt

For at det senere skal være mulig å sette opp funksjonelle regulatorer for systemet, er det nødvendig med matematiske modeller som beskriver hvordan ROV-en oppfører seg i de ulike frihetsgradene. Modellene kan utledes ved å summere opp de kreftene som virker på ROV-en både når den står i ro, og er i bevegelse. Dette kapittelet tar for seg utledningen av disse modellene, og til slutt en simulering av dem. Modellene for bevegelse i hiv-retning og i stamp- og rull-rotasjon danner grunnlaget for å sette opp regulatorer for disse bevegelsene.

7.1 Valgte frihetsgrader

For å lage matematiske modeller er det hensiktsmessig å velge ut hvilke frihetsgrader som skal analyseres, og se på hver av dem enkeltvis. Det å skulle modellere bevegelse i flere frihetsgrader samtidig vil være en matematisk veldig krevende oppgave, da det er mange aspekter med væskedynamikk man må ta hensyn til. Ofte kan forenklede modeller være en bedre tilnærming til virkeligheten, enn modeller som prøver å beskrive systemet pinlig nøyaktig. Modellene skal altså prøve å beskrive hvordan ROV-en oppfører seg i vann i hver frihetsgrad. Selv om ikke

alle frihetsgrader skal reguleres, kan det likevel være nyttig å lage modeller på dem. Dette vil kunne gi hele prosjektgruppen en dypere innsikt i hvordan ROV-en oppfører seg, og eventuelt hva man må ta hensyn til. Dette kan for eksempel være å estimere ROV-ens topphastighet i de ulike retningene. Modellene for translatorisk bevegelse langs aksene baserer seg på summen av alle krefter i den gitte retningen, og modellene for rotasjon om aksene baserer seg på summen av alle dreiemomenter rundt aksen.

7.2 Translasjon

Translasjon betyr at ROV-en beveger seg relativt til omgivelsene uten å rotere, og det skal lages modeller for bevegelse i frihetsgradene jag, svai og hiv, som presentert i delkapittel 6.1.1. I utgangspunktet lages modellene for å gi et grunnlag for utvikling av regulatorer. Jag og svai skal ikke reguleres, men det er likevel interessant å lage modeller for disse. Modellene skal beregne ROV-ens hastighet, og posisjon der det er relevant, i de ulike retningene. Det å vite ROV-ens topphastighet i jag- og svai-retning kan bidra med informasjon om hvor langt ROV-en kan kjøre før piloten kan se en endring på videostrømmen i operatørgrensesnittet. Det kan nemlig være en forsinkelse på opp mot 200 ms fra sanntid i videostrømmen. Forsinkelsen er oppgitt fra gruppen med ansvar for bildebehandling.

Grunnlaget for beregningene er Newtons 2. lov:

$$\sum \vec{F}_{x,y,z} = m \cdot \vec{a}_{x,y,z} \tag{7.1}$$

Allerede ut fra Newtons 2. lov ser man et bidrag som virker på translatorisk bevegelse. Endringen i bevegelsestilstand av et objekt, eller med andre ord akselerasjonen, er bekrevet som forholdet mellom kraftsummen og objektets masse. Alle objekter har en masse som bidrar til treghet, og for å finne akselerasjonen av ROV-en må vi altså dividere kraftsummen på massen av den. For å finne kraftsummen må man vurdere bidraget fra alle krefter som virker på den frihetsgraden som analyseres. Kommende delkapitler går gjennom disse kraftbidragene hver for seg.

7.2.1 Thrusterbidrag

I testrapporten i vedlegg B.1 ble det gjennomført tester av thrusterne, og opprettet en karakteristikk på hvor mange kilogram kraft de kunne produsere i forhold til prosentvis pådrag. Kraftsummen fra thrusterne avhenger av hvilken vinkel de står i forhold til bevegelsesakse, og antall thrustere som fungerer langs aksen. For å få kreftene målt i Newton må man også multiplisere med tyngdeakselerasjonen.

Jag og svai

I jag- og svai-retning får vi kraftbidrag fra de fire thrusterne som står horisontalt. Som beskrevet i kapittel 6, er det maksimale kraftbidraget i hver retning begrenset til å være lik summen av kreftene fra 2 thrustere.

$$F_{T,tot} = 2 \cdot u_{T,k}(\alpha) \cdot g$$

Hvor

- $u_{T,k}$ er thrusterkarakteristikken til hver thruster målt i kg som funksjon av prosentvis pådrag. Karakteristikken er utledet i testrapporten i vedlegg B.1.
- α er det prosentvise pådraget
- g er jordens tyngdeakselerasjon

Hiv

For hiv-retning er det de vertikale thrusterne som gir kraftbidraget. Alle disse er rettet parallelt med z-aksen, og virker derfor med full effekt.

$$F_{T,tot} = 4 \cdot u_{T,k}(\alpha) \cdot g$$

7.2.2 Vannmotstand

Ved bevegelse av ROV-en i et medium som vann vil det oppstå en økende motstand med økende hastighet. Denne motstanden kalles dragkraft [59], og fungerer på samme måte som luftmotstand. Kraften fungerer alltid mot bevegelsesretning. Formelen er gitt ved:

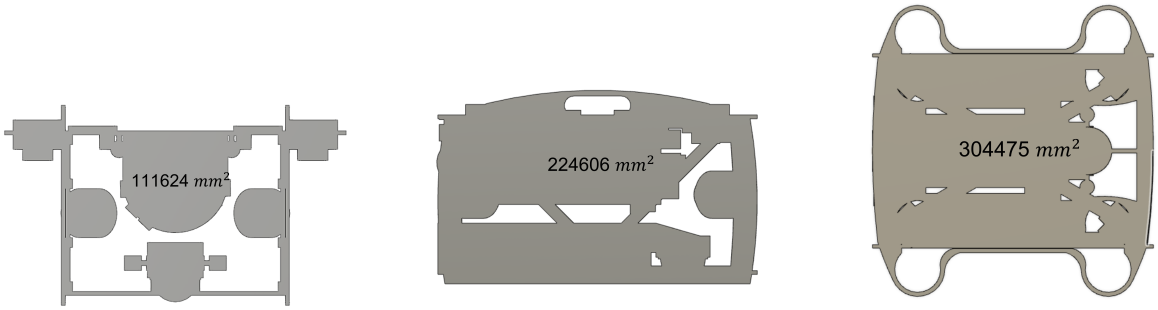
$$F_D = \frac{1}{2} \cdot \rho \cdot v^2 \cdot C_D \cdot A \tag{7.2}$$

Hvor

- F_D er dragkraften
- ρ er vannets massetetthet. Denne blir satt til å være 1000 kg/m^3
- C_D er ROV-ens dragkoeffisient
- A er arealet på ROV-en sett fra bevegelsesretningen

Dragkoeffisienten forteller hvor dynamisk ROV-en er i vann, og den vil bli gjennomgått i delkapittel 7.2.3.

For å finne størrelsen på vannmotstanden er man avhengig av å vite ROV-ens areal sett fra hver side. For å finne dette har vi brukt *Autodesk Fusion 360* til å prosjektere den fullstendige CAD-modellen gjennom et egendefinert plan. Programmet regner også ut arealet målt i mm^2. Dette må dog kompenseres for i ligningen, og omgjøres til m^2. Figur 7.1 viser prosjektert areal av hver side.



(a) Arealet sett forfra. **(b)** Arealet sett fra siden. **(c)** Areal sett ovenfra.

Figur 7.1

Vi valgte å fjerne thrusterne sett fra oversiden, da disse vil rotere, og sørge for at vannet flyter fritt under bevegelse i hiv-retning. De vil dermed ikke bidra nevneverdig til vannmotstanden.

Et siste punkt som er verdt å merke seg med ligningen for dragkraft er at den øker kvadratisk med farten til objektet. Ved å kvadrere farten til ROV-en mister man informasjon om hvilken retning den beveger seg langs aksen. Modellen skal ideelt sett virke i både positiv og negativ retning langs aksene. Problemet løses ved å multiplisere farten med absoluttverdien av farten. Man sørger da for å beholde et negativt fortegn dersom det skulle være aktuelt. Kraftbidraget fra vannmotstanden blir altså:

$$F_{D,x,y,z} = \frac{1}{2} \cdot \rho \cdot v \cdot |v| \cdot C_D \cdot A_{x,y,z}$$

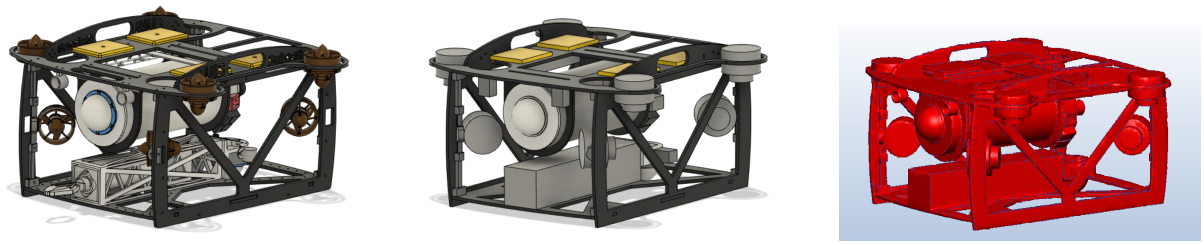
7.2.3 Drag-koeffisient

Som nevnt tidligere, forteller drag-koeffisienten hvor dynamisk ROV-en er i vann. Denne koeffisienten beskriver hvordan formen på et objekt påvirker friksjonskreftene som oppstår når gjenstanden beveger seg gjennom et medium.

Man kan for mange former finne målte drag-koeffisienter som kan brukes i utregninger, men for en ROV med en inhomogen overflate er det vanskelig å finne en drag-koeffisient teoretisk. I mange tilfeller er det mest praktisk å bygge konstruksjonen for deretter å måle de virkelige kreftene som påvirker konstruksjonen. Vindtuneller har blitt brukt til dette formålet i mange tiår. Ettersom ROV-en ikke er ferdig bygget og vi ikke har tilgang på en vindtunell, eller en vanntunell, er det nest beste alternativet å simulere kreftene som virker på ROV-en

Programmet *Autodesk CFD*, hvor CFD står for *computational fluid dynamics*, er laget for å utføre slike simuleringer. Utfordringen er at slike simuleringer er ekstremt prosessorkrevende. CAD-modellen som simuleres må derfor forenkles slik at unødvendige detaljer fjernes, samtidig som den generelle formen til objektet beholdes. Utgangspunktet for simuleringen er en 3D-modell av ROV-en sammensatt av maskingruppen med ansvar for rammen, vist i figur 7.2a. Denne modellen innholder mange detaljer som er viktige for å regne ut massen, volumsenter og massesenter til ROV-en, men som er unødvendige for analysen av drag-kreftene. For å forenkle ROV-en brukes et verktøy i Autodesk Fusion 360 som lar brukeren erstatte komplekse deler med enklere geometriske former. Et eksempel på dette er thrusterne, som har blitt gjort om

til sylindere. Den forenklede modellen er vist i figur 7.2b. Videre vil Autodesk CFD forenkle modellen i sin helhet og fjerne mindre detaljer som ikke allerede er fjernet. Modellen forenklet i CFD er vist i figur 7.2c.



(a) Original modell laget av maskingruppen

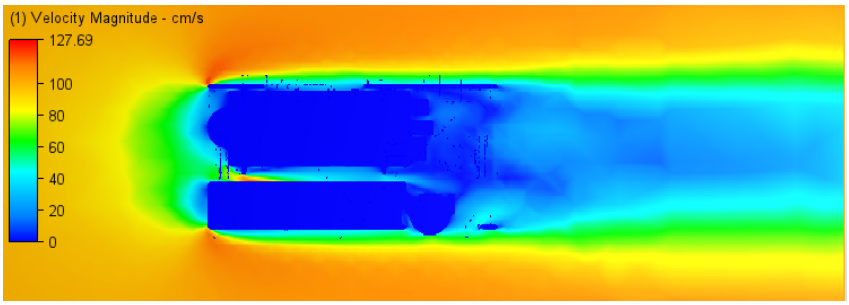
(b) Forenklet modell i Fusion 360

(c) Forenklet modell i Autodesk CFD

Figur 7.2

Når modellen er klar til å brukes i simuleringen, oppretter man et prisme rundt modellen for å definere simuleringsområdet. Innenfor prismet simuleres det en vannstrøm. Det er viktig at prismet er en god del større enn modellen ettersom veggene i prismet vil virke som fysiske vegger i simuleringen. Det er derfor en fare for at det oppstår uventede krefter av at vannet «presses» mellom ytterkanten av prismet og overflaten av ROV-modellen. For å simulere at ROV-en beveger seg i vannet, velger man en av sidene i prismet, og angir hvilken hastighet vannet skal komme inn fra denne siden med. For enkelhets skyld er hastigheten på vannet satt til å være 1 m/s. Den motstående siden av prismet må defineres som en utgang for vannet slik at det ikke bygges opp trykk. Tettheten på vannet som fyller prismet settes til å være 1000 kg/m^3.

Selve simuleringen er en enormt prosessorkrevende prosess, og tar lang tid selv på en kraftig datamaskin. For å minke simuleringstiden kan man sette en begrensning for hvor mange steg simuleringen skal gjøre. Alternativt kan simuleringen kjøre til programmet ser at endringene for hvert steg er så små at man har nådd *steady-state*. For simuleringen av at ROV-en beveger seg langs x-aksen brukte programmet 412 steg og omtrent 3 timer. Visualiseringen av resultatet er vist i figur 7.3.



Figur 7.3: Resultat fra simuleringen av ROV-en i bevegelse langs x-aksen. I bildet ser man den fra venstre side i kjøreretning.

Autodesk CFD vil også oppgi summen av kreftene som virker på modellen i jag-retning. I dette tilfellet var dragkraften 70.3897 N. Hvis man omstrukturerer formel 7.2, og bruker verdiene fra simuleringen, kan man finne den simulerte drag-koeffisienten:

$$C_D = \frac{2 \cdot F_D}{\rho \cdot v^2 \cdot A} \tag{7.3}$$

Setter man inn verdiene for det prosjekterte arealet, vist i figur 7.1a, hastigheten på 1 m/s og tettheten til vannet, på 1000 kg/m^3, får man en dragkoeffisient på 1.261.

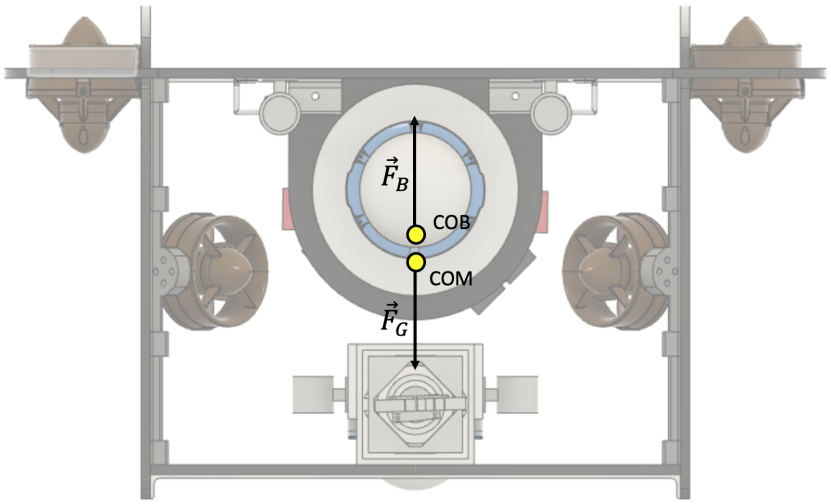
Prosessen gjentas for bevegelse i svai- og hiv-retning. For bevegelse i z-retning vil, som nevnt tidligere, de vertikale thrusterne fjernes ettersom disse ikke vil bidra til vannmotstand. Resultatet for de 3 simuleringene er vist i tabellen under:

Retning	Simulert F_D [N]	Prosjektert areal [m^2]	C_D
jag	70.3897	0.111624	1.261
svai	158.318	0.224606	1.410
hiv	191.407	0.304475	1.257

De simulerte drag-koeffisientene er kun estimater, og tar ikke hensyn til blant annet kabler som vil gå inne i rammen. Forenklingene av CAD-modellen, samt påvirkninger fra prismet, bidrar også til usikkerhet i verdiene. Ettersom simuleringen kun er gjort med vannstrøm mot én av endene i hver retning vil det også her kunne oppstå feilbidrag. Dette gjelder spesielt for bevegelse i hiv- og jag-retning, da ROV-en ikke er lik sett fra begge ender i disse retningene. Drag-koeffisienten for bevegelse i svai-retning er også en del større enn for jag og hiv. Det er vanskelig å anslå nøyaktig hva som forårsaker dette, men antakeligvis kan det forklares med vannturbulens som oppstår når vannet kurver rundt elektronikkhuset. Dette vil kunne gi økt motstand når vannet samler seg opp under topplaten av rammen. Når det er sagt, ville alternativet til simuleringen være å estimere drag-koeffisientene ved å forenkle ROV-en til en enkel geometrisk form, og bruke en tabellverdi for disse formene. Verdiene vi får fra simuleringen, er ikke langt fra drag-koeffisienter oppgitt for en flat plate. Disse er nemlig estimert til å være 1.28 [16]. Det er derfor ikke urimelig å anta at verdiene fra simuleringen er representative for det faktiske systemet. Vi velger på grunn av dette å ha større tillit til simuleringsprogrammet enn til andre forenklede metoder.

7.2.4 Oppdrift

Et siste kraftbidrag som virker på translatorisk bevegelse, er summen av oppdriftskraft og gravitasjonskraft på ROV-en. Oppdriftskraften virker alltid parallelt med tyngdekraften, uavhengig av ROV-ens orientering. Ser man isolert på translatorisk bevegelse, og hvor ROV-en står stabilt vannrett, vil oppdriftskraften kun være gjeldende for hiv-retningen. Figur 7.4 viser en skisse over hvordan kraftvektorene virker.



Figur 7.4: COM (***C**entre **o**f **m**ass*) er ROV-ens massesenter. Det er her gravitasjonskraften fra jorden virker. COB (***C**entre **o**f **b**uoyancy*) er ROV-ens volumsenter. Her virker oppdriftskraften fra vannet.

Gravitasjonskraften er kun avhengig av objektets masse, og jordens tyngdeakselerasjon:

$$F_G = m \cdot g \tag{7.4}$$

Dersom man senker et objekt ned i vann, vil tyngden av objektet minke. Dette kommer av oppdriftskraften som virker mot tyngdekraften. Grunnen til at det oppstår oppdrift er at vannet som fortrenges av ROV-en vil presse inn mot overflaten av den. Dess dypere man kommer, dess høyere er trykket i vannet. Dette betyr at vannet trykker med større kraft mot bunnen av ROV-en enn mot toppen. Denne trykkforskjellen er alltid like stor uavhengig av hvor dypt ROV-en befinner seg, da det kun er høydeforskjellen mellom topp og bunn av rammen som har betydning. Arkimedesloven konstaterer at denne oppdriftskraften er lik tyngden av vannet som objektet fortrenger [2]. Kraften er gitt ved:

$$F_B = \rho_{vann} \cdot g \cdot V_{vann} \tag{7.5}$$

Volumet av vannet som fortrenges er naturligvis lik volumet av ROV-en. Dette gjør at man kan designe den etter ønsket oppførsel i hiv-retning. Ved å balansere masse mot volum kan man i forveien bestemme netto oppdriftskraft. Det er hensiktsmessig å ha en netto oppdriftskraft som gjør at ROV-en flyter av seg selv. Dette sørger for at den ikke synker til bunns ved en eventuell stans i krafttilførsel. Gruppen med ansvar for rammedesign av ROV-en har oppgitt at netto oppdriftskraft er:

$$F_{b,netto} = 1.2\ N$$

7.2.5 Simulering av modeller

Modellene baserer seg som sagt på Newtons 2. lov ved å summere opp alle kraftbidrag. Retningen kreftene virker langs aksene blir tatt hensyn til, da både farten og akselerasjonen kan

være negativ. Likevel legger vi til et minustegn fremfor både dragkraften og oppdriftskraften. Dragkraften virker alltid motsatt av bevegelsesretning, hvor fortegnet gis av hastigheten, og oppdriftskraften virker alltid i negativ retning uavhengig av bevegelse. I kapittel 6 ble positive retninger definert til å være fremover, til høyre og nedover. De summerte kraftbidragene er listet opp under:

$$F_{T,tot} - F_D - \underbrace{F_{b,netto}}_{For\ hiv} = m \cdot a \tag{7.6}$$

Ligningen man ender opp med er avhengig av både hastigheten og akselerasjonen til ROV-en. Da akselerasjonen er lik den tidsderiverte av hastigheten, har vi altså en differensialligning. For å lage en simuleringsmodell, er ligningen skrevet ut mer detaljert under. For hiv vil denne, altså ligning 7.7, være helt riktig. For jag og svai må det første 4-tallet erstattes med 2 på grunn av antall thrustere, og netto oppdrift har ingen innvirkning.

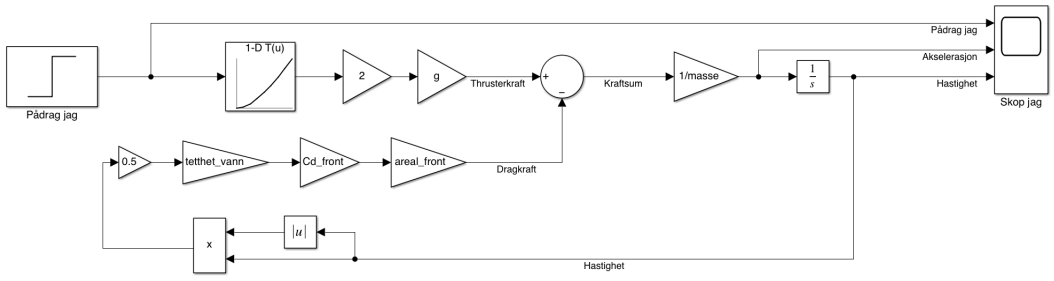
$$4 \cdot u_{T,k}(\alpha) \cdot g - \frac{1}{2} \cdot \rho \cdot v \cdot |v| \cdot C_D \cdot A_{x,y,z} - \underbrace{1.2}_{For\ hiv} = m \cdot a \tag{7.7}$$

For å løse ligningen for et gitt pådrag har vi valgt å bruke *Simulink*, som er et grafisk programmeringsverktøy fra *MATLAB*. Ved å omstrukturere ligning 7.7, og sette akselerasjonen lik 0, kan man verifisere at *Simulink* gir ut riktige topphastigheter ved å bruke formelen:

$$v = \sqrt{\frac{4 \cdot u_{T,k}(\alpha) \cdot g - \underbrace{1.2}_{For\ hiv}}{\frac{1}{2} \cdot \rho \cdot C_D \cdot A_{x,y,z}}} \tag{7.8}$$

Modell for jag

Simulink er et program som baserer seg på blokker for ulike operasjoner. Under vises modellen for jag-retning. Simuleringen er gjennomført med et steg i thrusterpådraget fra 0 til 100 %. Alle variabler i de ulike blokkene kommer fra et MATLAB-script som er kjørt i forkant av simuleringen.



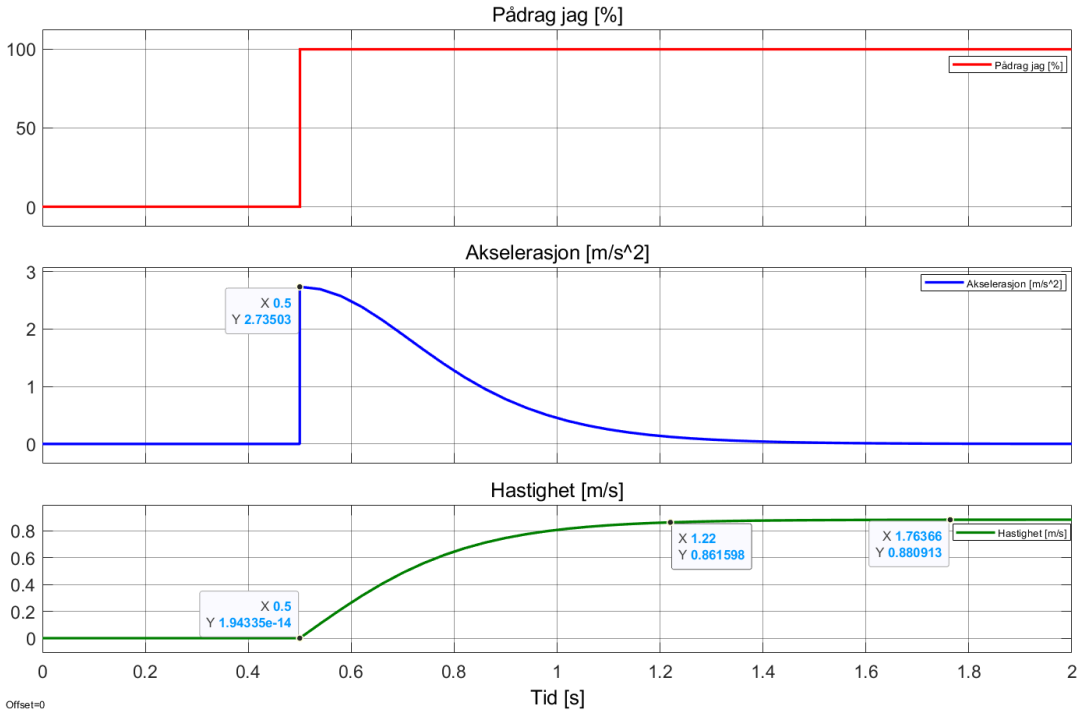
Figur 7.5: Modell i simulink for jag-bevegelse.

Blokken markert med «Pådrag jag» simulerer det prosentvise pådraget på thrusterne. Dette signalet sendes inn på en *look-up table*, som er blokken med motorkarakteristikk. Denne gir altså ut

pådraget i antall kilogram. Trekantformede blokker er av typen *gain*, og er multiplikasjonsblokker. Maksimal total thrusterkraft i horisontal retning er som nevnt lik kraften fra 2 thrustere, som også multilpiseres med tyngdeakselerasjonen for å få ut kraften i Newton. Den sirkelformede blokken er en summasjonsblokk, som summerer verdier med tilhørende fortegn markert ved inngangspilene på blokken.

Utgangen av summasjonsblokken tilsvarer kraftsummen i ligningen. Denne divideres på massen for å ta hensyn til massetregheten til ROV-en. Blokken markert med $\frac{1}{s}$ er en integratorblokk. Denne integrerer akselerasjonen opp til hastighet. Integratorblokken muliggjør en tilbakekobling av hastigheten, som bidrar til kraftsummen i form av dragkraft. Hastigheten sendes inn på en absoluttverdi-blokk, og en produktblokk. Denne kombinasjonen finner farten kvadrert, og tar samtidig hensyn til fortegn. Signalet multipliseres videre med nødvendige skalarer for å få riktig dragkraft inn på summasjonsblokken. Dragkraften er summert negativt i kraftsummen av den grunn at den alltid virker i motsatt retning av hastigheten.

Til høyre i modellen er det lagt inn en skop-blokk, som viser frem ønskede signaler. Hastigheten er den mest interessante verdien for modellen, men akselerasjonen og steget i pådrag er også lagt inn for å få bedre oversikt over bevegelsen. Skopbildet etter simulering er vist i figuren under.



Figur 7.6: Skopet etter simulering av jag-bevegelse. Den røde grafen viser pådraget målt i %. Blå graf viser akselerasjonen som oppstår etter at pådraget er satt, og grønn viser integrert hastighet.

Modellen er simulert i 2 sekunder, og pådragssteget skjer etter 0,5 sekunder. Ifølge modellen kommer ROV-en opp i en hastighet på 0,88 m/s. Dette kan verifiseres med ligning 7.8:

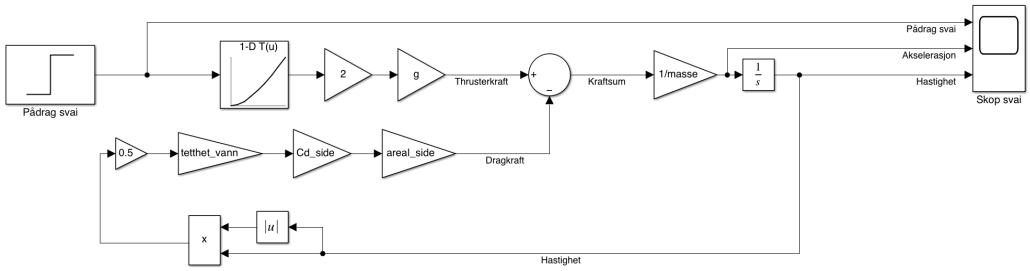
$$v = \sqrt{\frac{2 \cdot 2.788 \cdot 9.81}{\frac{1}{2} \cdot 1000 \cdot 1.261 \cdot 0.111624}} \approx 0.88$$

Her er tallet 2.788 hentet fra thrusterkarakteristikken som tilsvarer et pådrag på 100% målt i

kg. Ettersom topphastigheten regnes riktig av Simulink, velger vi derfor å stole på simuleringen av svai og hiv uten å vise verifiseringen av topphastigheten. Med en hastighet på 0.88 m/s, og en potensiell forsinkelse på 200 ms i videostrømmen, kan ROV-en kjøre ca. 17.6 cm før piloten får informasjon fra kameraet i operatørgrensesnittet. Videre tar det omtrent 0.72 sekund fra thrusterne skrus på, til ROV-en når 98% av topphastigheten.

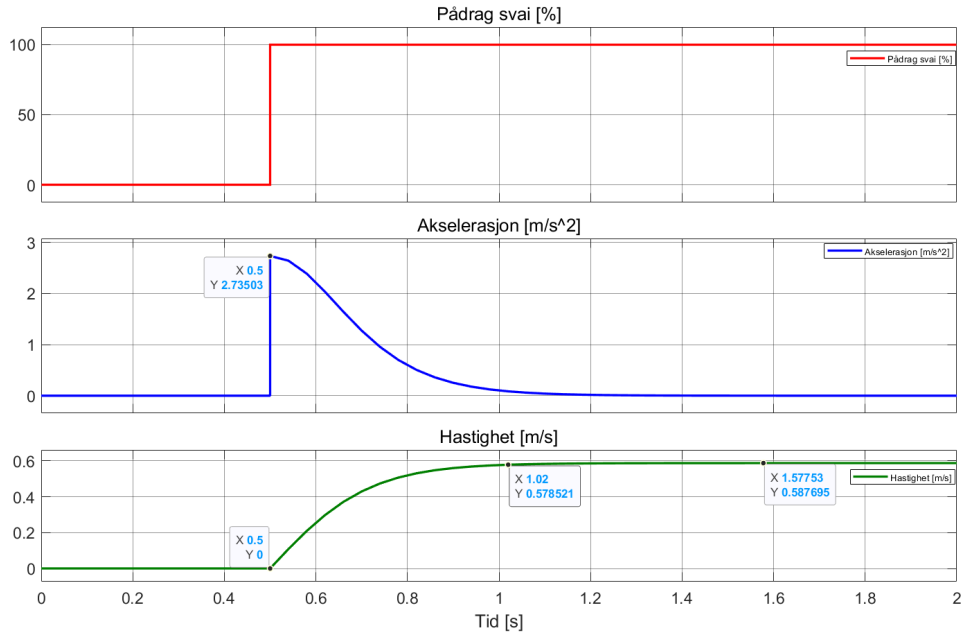
Modell for svai

Simulinkmodellen for svairetning er tilnærmet den samme som for jag. De eneste to forskjellene er arealet på overflaten som beveger seg mot vannet, samt den simulerte dragkoeffisienten. Her blir det også simulert et sprang i det prosentvise pådraget fra 0 til 100 %. Figur 7.7 viser Simulink-modellen for svai-bevegelse.



Figur 7.7: Modell i simulink for svai-bevegelse.

Figur 7.8 viser skopbildet etter simulering.



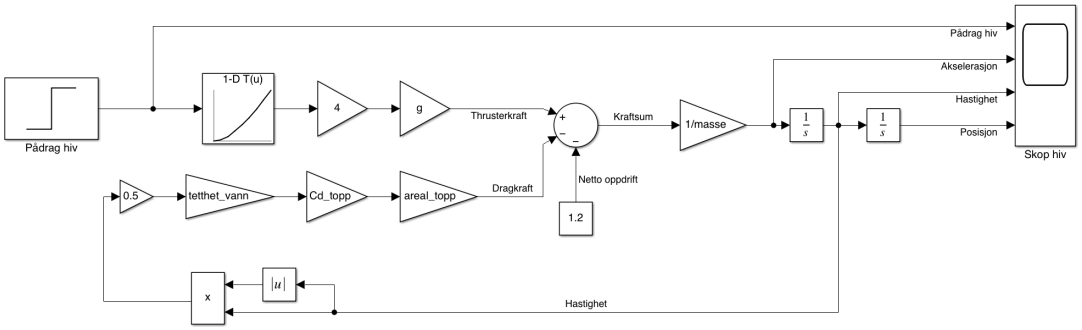
Figur 7.8: Skopet etter simulering av svai-bevegelse.

Modellen er simulert like lenge, og spranget går på samme tidspunkt som i jag-modellen. Tiden det tar for ROV-en å nå 98% av topphastigheten er omtrent 0.52 s. Topphastigheten i dette tilfellet ligger på 0.59 m/s. Det vil være naturlig at denne farten er litt lavere enn toppfarten for jag-bevegelse. Både arealet og dragkoeffisienten er større for sidene. Dette vil bidra til mer

vannmotstand enn ved kjøring fremover og bakover. Med en hastighet på 0.59 m/s, og en potensiell forsinkelse på 200 ms i videostrømmen, kan ROV-en kjøre ca. 11.8 cm før piloten får informasjon fra kameraet i operatørgrensesnittet.

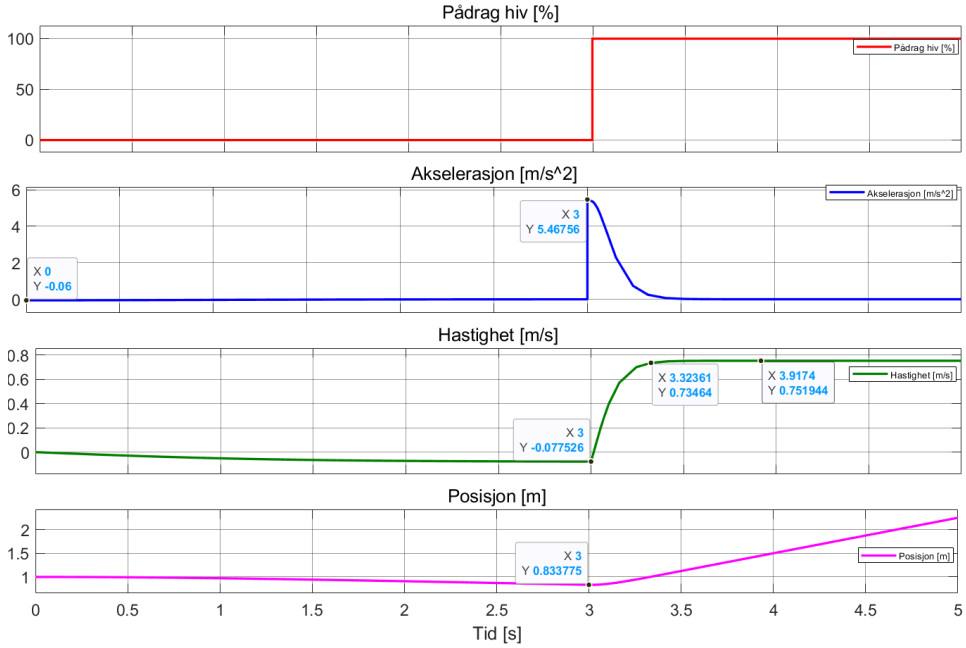
Modell for hiv

Modellen for hiv-retning tar i betraktning at det kan virke fire thrustere i bevegelsesretningen, og derfor er den ene «gain»-blokken byttet fra 2 til 4. I tillegg til at også hiv-modellen bruker egne parameterer for overflateareal og dragkoeffisient, er den konstante oppdriften lagt inn som et bidrag til kraftsummen. Denne er lagt inn i summen med negativt fortegn siden den alltid virker oppover, som er den negativt definerte retningen. I modellen er det også lagt inn en ekstra integrator, som gjør at skopet kan vise frem dybdeposisjonen. Denne er også nødvendig for at vi senere skal kunne utvikle og simulere dybderegulatoren. Modellen er vist i figur 7.9.



Figur 7.9: Modell i simulink for hiv-bevegelse.

For at man i skopet skal kunne se virkningen av oppdriften, er det lagt inn en initialverdi på posisjonsintegratoren før simulering. Denne er satt til å starte på 1 meter, slik at ROV-en skal kunne flyte litt oppover før spranget i pådraget går. Skopbildet etter simuleringen er vist under.



Figur 7.10: Skopet etter simulering av hiv-bevegelse. Her er også posisjonen vist frem.

Modellen er i dette tilfellet simulert i 5 sekunder, hvor spranget i pådrag går etter 3 sekunder. Før spranget går, ser man at ROV-en begynner å flyte oppover. Dette viser seg som en negativ akselerasjon og hastighet i første del av simuleringen. Hastigheten den stabiliserer seg rundt er på -0.077 m/s. Etter 3 sekunder har ikke ROV-en flyttet seg lenger opp enn fra 1 meter til omtrent 0.83 meter. Dette ser vi på som fordelaktig. Det trengs mindre thrusterpådrag fra regulatoren for å holde en angitt dybde dersom kraftbidraget fra oppdrift er minst mulig. Når spranget fra 0 til 100 % pådrag går, bruker ROV-en rundt 0.32 sekunder på å nå 98% av topphastigheten nedover. Merk at ROV-en starter med en negativ hastighet før spranget går, som vil øke denne tiden noe. Dette er fortsatt dobbelt så raskt som for de andre retningene, som er et resultat av at alle fire thrustere virker samtidig i fartsretningen. Hastigheten den oppnår er 0.75 m/s, som kan føre til 15 cm bevegelse iløpet av kameraforsinkelsen. Modellen for hiv vil i kapittel 8 brukes som utgangspunkt for hiv-regulatoren som skal settes opp.

7.3 Rotasjon

Det skal også lages modeller for rotasjon om de tre aksene, som tilsvarer tre av frihetsgradene presentert i delkapittel 6.1.1. Modellen for gir lages av samme grunn som modellene for jag og svai. Kreftene som påvirker rotasjon må ses på som tangentielle komponenter til rotasjonsbevegelsen, og som bidrar til et totalt dreiemoment i en gitt retning. Det totale dreiemomentet resulterer i en vinkelakselerasjon om den aktuelle aksen man vurderer. Summasjonsformelen for alle dreiemoment er en ekvivalent til Newtons 2. lov [1]:

$$\sum \tau_{\phi,\theta,\psi} = I \cdot \alpha_{\phi,\theta,\psi} \tag{7.9}$$

Hvor

- $\sum \tau$ er summen av alle dreiemoment som virker på rotasjonen
- α er vinkelakselerasjonen om hver akse
- I er treghetsmomentet til objektet som roterer rundt den gitte aksen

Som for translatorisk bevegelse, må man for rotasjon beskrive alle de ulike kreftene som bidrar til dreiemoment om hver akse. Kommende delkapitler tar for seg hvert av disse kraftbidragene. I tillegg vil treghetsmomentet fra ROV-ens masse beskrives mer i detalj, da det bidrar annerledes enn for translasjon.

7.3.1 Treghetsmoment

Tregheten av et objekt under translastorisk bevegelse er gitt av massen i seg selv. Under rotasjon må man også ta hensyn til hvordan massen av objektet er fordelt rundt rotasjonsaksen. Dersom ytterkantene av ROV-en er tyngre enn senteret av den, vil det kreve mer kraft for å skape rotasjon. Å gi en eksakt verdi av treghetsmomentene ved rotasjon rundt aksene er utfordrende, men det er laget en CAD-modell av hele ROV-en i *Autodesk Inventor*, og treghetsmomentene kan beregnes ut fra denne. I tillegg til å inneholde data om fysiske mål, er det også lagt inn

estimerte data på tettheten til de ulike komponentene. Dette gjør at programmet kan regne ut fordelingen av masse rundt alle akser relativt til massesentrum. Små komponenter, som for eksempel ledninger, er ikke lagt inn i modellen og vil ikke kunne bli tatt høyde for, men den totale massen av ROV-en som programmet regnet ut stemmer overens med det prosjektledelsen har planlagt. Det vil uansett være usikkerhet knyttet til forskjellen i modell og ferdig produkt. Programmet har regnet ut følgende verdier for treghetsmomentene:

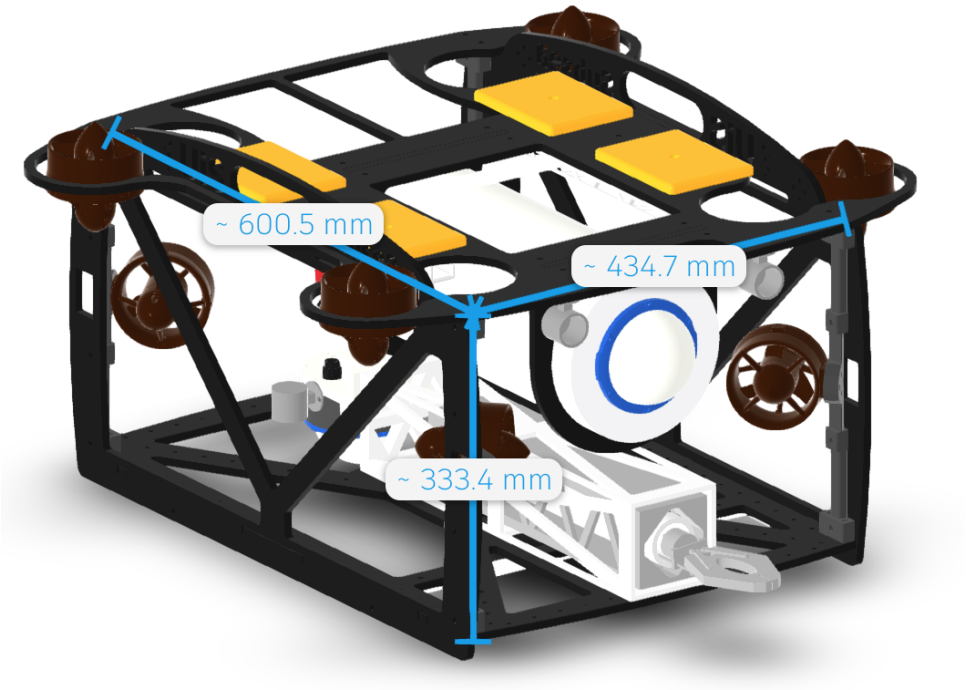
Tabell 7.1: Treghetsmomenter regnet ut av *Autodesk Inventor*

	Rull	Stamp	Gir
Treghetsmoment I $[kg \cdot m^2]$	0.496497	0.792810	0.869830

Hvorvidt verdiene stemmer eller ikke, er vanskelig å vurdere. For å verifisere at estimatene kan brukes, kan man sammenligne dem med treghetensmomentet av et objekt med enklere form. ROV-en kan sammenlignes med et prisme, og man kan da regne ut treghetsmomentet ved hjelp av følgende formel [60]:

$$I = \frac{1}{12} m \cdot (l^2 + b^2) \tag{7.10}$$

Lengden og bredden er angitt av sidekantene til det planet av prismet som står vinkelrett på rotasjonsaksen. For å sammenligne ROV-en med et enkelt prisme trenger man lengden av alle ytterkantene. Vi har valgt å måle sidene innenfor der de vertikale thrusterne er plassert.



Figur 7.11: CAD-modell av ROV med målinger av sidekanter.

Bruker man ligning 7.10 og setter inn lengdene fremvist i figur 7.11, samt massen på 20 kg, ender man opp med estimatene i tabell 7.2.

Tabell 7.2: Treghetsmomenter regnet ut fra approksimasjon.

	Rull	Stamp	Gir
Lengde [m]	0.4347	0.6005	0.6005
Bredde [m]	0.3334	0.3334	0.4347
Treghetsmoment I $[kg \cdot m^2]$	0.500199	0.786260	0.915941

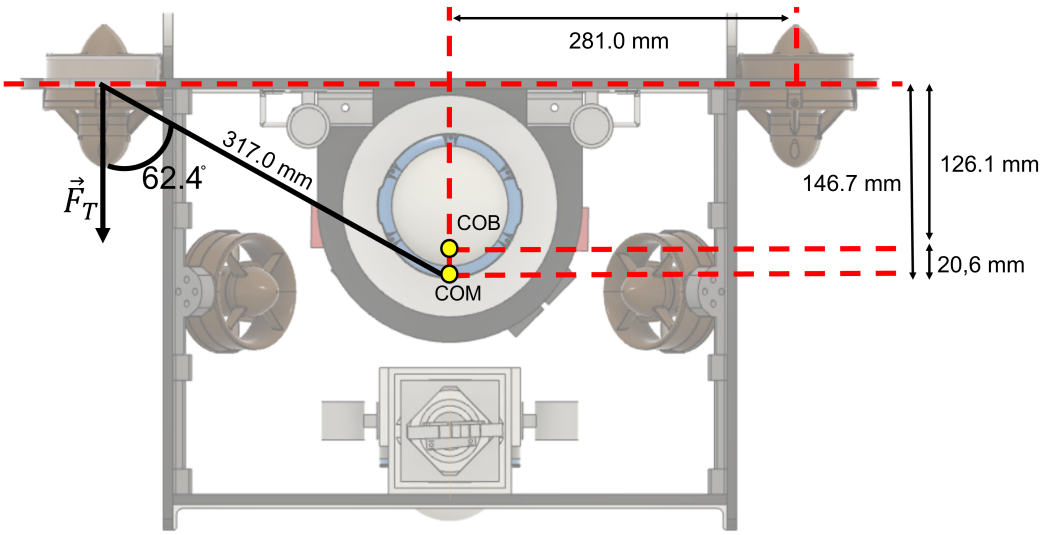
Ved å sammenligne verdiene fra tabell 7.1 og 7.2 konkluderer vi med at treghetsmomentene regnet ut av *Inventor* kan brukes. Differansen i treghetsmoment er relativt liten, og simuleringen ville ikke blitt særlig annerledes ved å bruke det ene estimatet over det andre. Vi velger likevel å bruke verdiene regnet ut av programmet, da vi ser på disse som de mest pålitelige i forhold til vektfordeling rundt aksene. ROV-en er ikke et objekt med homogen fordeling av massen, slik som prismet er.

7.3.2 Dreiemoment fra thrustere

Thrusterne yter samme kraft for rotasjon som for translasjonsbevegelse. Både thrusterkarakteristikken og tyngdeakselerasjonen tas hensyn til for å finne hvor mange newton hver thruster bidrar med. Dreiemomentet disse kreftene bidrar til, er avhengig av både posisjonsvektoren fra rotasjonssentrum ut til thrusteren, og vinkelen mellom posisjonsvektor og kraftvektor. I modellene har vi gjort en antakelse om at ROV-en roterer om massesentrumet, uavhengig av hvilken akse det roteres om. Det vil si at posisjonsvektoren til hver thruster, går fra massesentrum av ROV-en, og ut til selve thrusteren.

Rull

I rull-rotasjon vil de vertikale thrusterne virke sammen to og to for å rotere ROV-en. Alt etter om den skal roteres med positiv eller negativ vinkel, virker høyre og venstre thruster-par motsatt retning av hverandre. Figuren under viser orientasjonen av hvert thruster-par i forhold til massesentrumet.



Figur 7.12: Plassering av vertikale motorer i forhold til rullrotasjon

Som vist i kapittel 2, ligning 2.1, er dreiemomentet lik kraft multiplisert med lengden av momentarmen. I tillegg må kraftvektoren fra thrusterne dekomponeres i en parallell, og en normal, på posisjonsvektoren. Komponenten som står normalt på posisjonsvektoren er den som bidrar til dreiemomentet. Størrelsen på denne finner man ved å bruke vinkelen θ_T, som i figuren tilsvarer 62.4°, og formelen.

$$F_{T,normal} = F_T \cdot cos(90^o - \theta_T)$$

Siden thuster-parene på hver side virker motsatt vei av hverandre, vil også vinkelen θ_T bli ulik. Dette trenger man ikke å ta hensyn til under utregningene. Normalkomponenten fra hver thruster i rotasjonsretning blir like stor fra hver av dem såfremt de bidrar med like stor kraft. Dette kan også begrunnes med formlikhet, da kraftvektorene står parallelt med hverandre. Totalt momentbidrag fra thrusterne kan da regnes ut fra formelen:

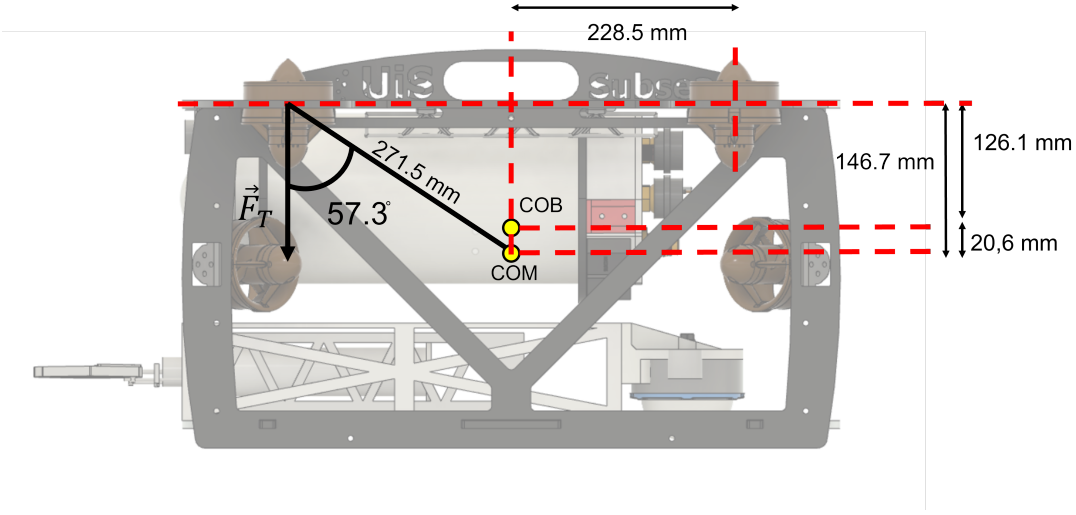
$$\sum_{i=1}^{4} \tau_i = \sum_{i=1}^{4} F_{T,i} \cdot r_T \cdot cos(90^o - \theta_T) = 4 \cdot F_T \cdot r_T \cdot cos(90^o - \theta_T) \tag{7.11}$$

Hvor

- $F_{T,i}$ er størrelsen av kraftvektor for hver thruster
- r_T er størrelsen av posisjonsvektor fra massesentrum til thrusterne
- θ_T er vinkelen mellom kraftvektor og posisjonsvektor

Stamp

Thrusterbidraget til stamprotasjon blir tilsvarende som for rull. Det er de vertikale thrusterne som sørger for dreiemomentet, men i dette tilfellet blir thruster-parene de fremre og de bakre. Figuren under viser lengden av posisjonsvektor, og vinkelen θ_T sett fra siden av ROV-en.

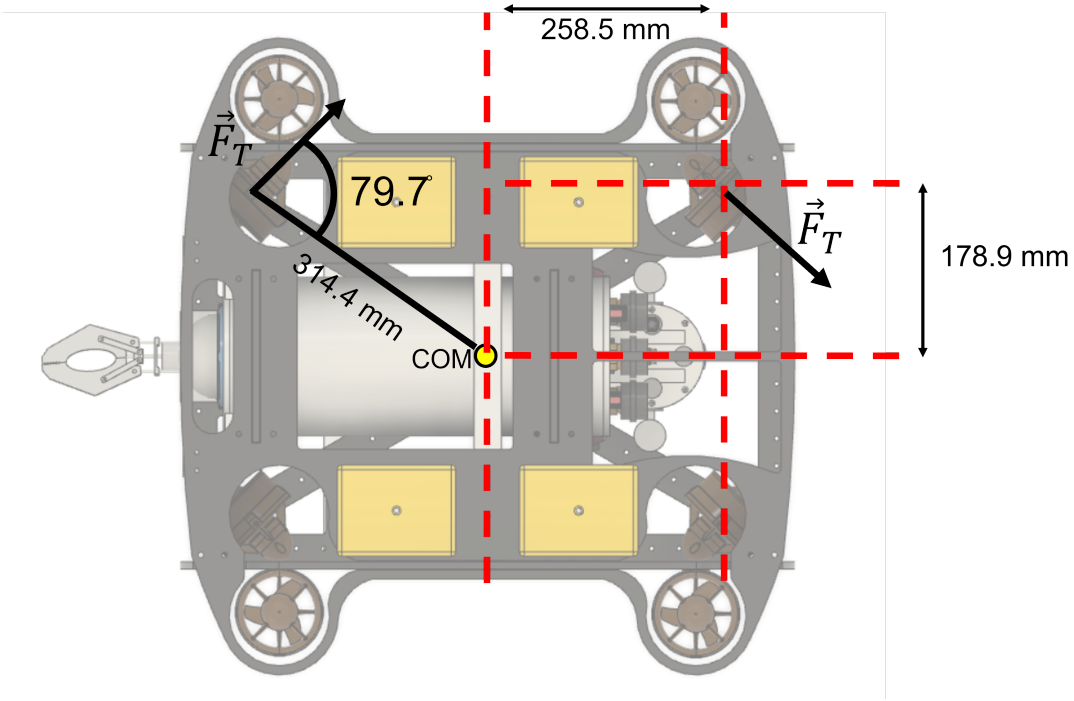


Figur 7.13: Plassering av vertikale motorer i forhold til stamprotasjon

For å finne momentbidraget bruker vi igjen ligning 7.11, og setter inn de riktige verdiene for r_T og θ_T. Ut fra figuren kan man se at det kreves et høyere pådrag fra thrusterne for å bidra til stamp-rotasjon enn det kreves for å bidra til rull-rotasjon. Momentarmen er i stamp-tilfellet kortere, og vinkelen er mindre. Begge faktorene bidrar til et lavere momentbidrag.

Gir

Til forskjell fra rull og stamp, er det for gir-rotasjon de horisontale thrusterne som bidrar. I dette tilfellet bidrar ikke thrusterne lenger i par, men alle kjøres med samme pådrag. Som vist i kapittel 6 vil diagonalt motstående thrustere virke motsatt rettet av hverandre. Orienteringen av thrusterne er vist i figuren under.



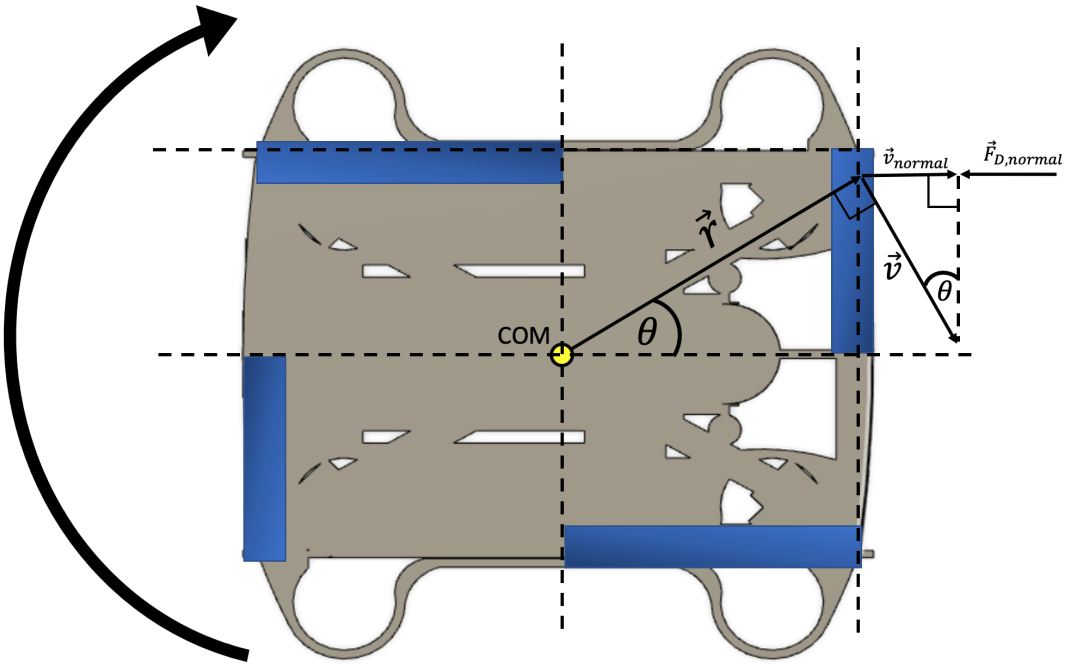
Figur 7.14: Plassering av vertikale motorer i forhold til girrotasjon. Det er her markert kraftvektorer på to av thrusterne, men i realiteten kjøres alle thrustere i samme rotasjonsretning. Vinkelen mellom posisjonsvektor og kraftvektor er også den samme for samtlige.

Igjen brukes ligning 7.11 med tilhørende verdier for r_T og θ_T for å regne ut momentbidraget. Posisjonsvektoren til thrusterne er hakket kortere enn for rull-rotasjon, men til gjengjeld er θ_T her større. Vinkelen gjør at thrusterne i gir-rotasjon bidrar med mer effekt på dreiemomentet enn for rull og stamp.

7.3.3 Vannmotstand

Som for translasjonsbevegelse, vil det også under rotasjon være motstand i vannet som øker med hastigheten til ROV-en. Likevel vil det være forskjeller i utregningen av motstanden når et objekt roterer. Under rotasjon opereres det med vinkelhastighet, og ikke absolutt hastighet. Da dragkraften er avhengig av farten til et objekt, vil ikke kraften motvirke rotasjonen likt over hele overflaten til ROV-en. Figur 7.15 viser hvordan dragkraften virker i et tilfeldig valgt punkt på overflaten av ROV-en under gir-rotasjon. For å kunne regne på kraften, antar vi at det kun

er normalkomponenten til dragkraften som bidrar til motstand i hvert punkt. Under rotasjon vil det i tillegg oppstå vesentlig turbulens i det omliggende vannet, men dette velger vi å se bort fra for å unngå de mest kompliserte beregningene.



Figur 7.15: Områdene der dragkraften virker på ROV-en er markert med blått, gitt at den roterer slik pilen indikerer.

Drag-kraften $\vec{F}_{D,normal}$, som er tegnet inn på figur 7.15 virker i dette tilfellet på bakenden av ROV-en. For å finne denne kraften, må man dekomponere fartsvektoren i hvert punkt, slik at vi finner hastigheten normalt ut fra overflaten av ROV-en. Man kan ut fra figuren se at dersom vi lar θ gå mot 0^o, vil hastighetsvektoren $\vec{v}$ virke parallelt med overflaten, og ikke bidra til motstand. Dersom θ blir satt negativ, vil hastighetsvektoren peke inn mot overflaten av ROV-en. Dette gjør at vi betrakter nedre halvdel av bakenden som en «bakside», der det ikke virker motstand fra vannet. Dette blir en ekvivalent til baksiden av ROV-en under translasjonsbevegelse. Feltene som betraktes som «fremsiden» av ROV-en under gir-rotasjon i klokkeretning er markert med blått.

Dragkraften er altså avhengig av normalhastigheten i hvert punkt av overflaten. Den absolutte hastigheten er avhengig av vinkelhastigheten objektet roterer med, og avstanden fra rotasjonssentrum til det gitte punktet.

$$v = r \cdot \omega \tag{7.12}$$

Hvor

- v er momentanhastigheten i hvert punkt på overflaten av ROV-en
- r er størrelsen av posisjonsvektoren fra massesentrum til punktet
- ω er vinkelhastigheten ROV-en roterer med.

Denne dekomponeres for å finne normalhastigheten:

$$v_{normal} = r \cdot \omega \cdot sin(\theta) \tag{7.13}$$

For å finne dragkraften i hvert punkt setter vi denne hastigheten inn i ligning 7.2:

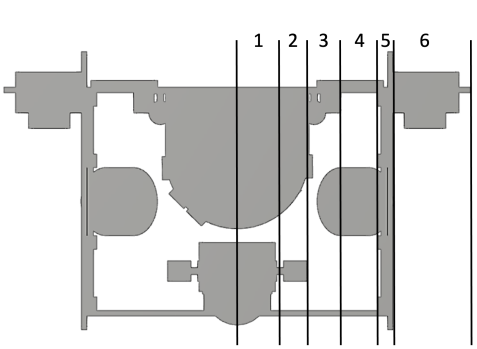
$$F_{D,normal} = \frac{1}{2} \cdot \rho \cdot (r \cdot \omega \cdot sin(\theta))^2 \cdot C_D \cdot A \tag{7.14}$$

Overflaten av ROV-en vil oppleves annerledes dersom den observeres fra ulike innfallsvinkler. Dette vil påvirke effekten av dragkraften. Dess mer parallelt med overflaten hastighetsvektoren $\vec{v}$ går, dess mindre blir dragkraften. I argumentasjonen over er normalhastigheten i hvert punkt brukt for å finne dragkraften som bidrar med motstand under rotasjon. Alternativt kan det tenkes at ved å se overflaten fra en gitt vinkel, vil dragkoeffisienten variere. Likevel er det i utgangspunktet utfordrende å anslå en dragkoeffisient, og på grunn av at vi regner med normalhastigheten velger vi å bruke de dragkoeffisientene vi allerede har simulert oss frem til siden disse er funnet basert på vannstrøm vinkelrett på overflaten.

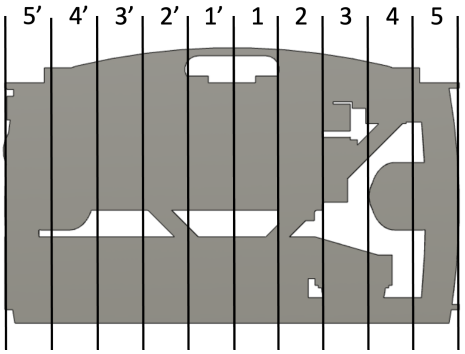
Dragkraften vil videre gi et momentbidrag til rotasjonen. Siden dragkraften virker med en vinkel $180° - \theta$ på momentarmen, må dette tas med i ligningen. For å beholde retningen av vinkelhastigheten, har vi brukt samme metode med absoluttverdi som for translasjon.

$$\begin{aligned}
\tau &= F_{D,normal} \cdot r \cdot sin(180° - \theta) \\
\tau &= \frac{1}{2} \cdot \rho \cdot (r \cdot \omega \cdot sin(\theta))^2 \cdot C_D \cdot A \cdot r \cdot sin(\theta) \\
\tau &= \frac{1}{2} \cdot \rho \cdot r^3 \cdot \omega \cdot |\omega| \cdot C_D \cdot A \cdot sin^3(\theta)
\end{aligned} \tag{7.15}$$

Ligning 7.14 angir dragkraften som virker på hvert punkt av overflaten til ROV-en. For å finne den samlede dragkraften som bremser, er man nødt til å integrere over områdene der dragkraften bidrar. Integrasjonen er ekstremt krevende, og skulle helst blitt gjort av et digitalt verktøy. Vi har dessverre ikke klart å finne programmer som kan gjøre dette for oss. Som et alternativ har vi derfor valgt å utføre numerisk integrasjon manuelt på overflatene. Fremgangsmåten vi bruker, går vi gjennom for eksempelet med gir-rotasjon. Først må vi dele opp sideflatene av ROV-en i passende seksjoner. Dette er illustrert i figur 7.16.



(a) Fremsiden er delt opp i søyler der vi ser et tydelig mønsterskift i arealet. Da fremsiden er symmetrisk, kan vi integrere over en halv side, og multiplisere med 2. Dette ser man av figur 7.15. Halve overflaten sett forfra bidrar til vannmotstand i hver sin ende av ROV-en.



(b) Sidenflaten er delt opp i søyler med like mellomrom, da arealet er nokså homogent over hele området. Likevel er ikke siden symmetrisk, og vi integrerer derfor over hele for å finne total motstand fra venstre og høyre side kombinert.

Figur 7.16

Hver enkelt seksjon bidrar ulikt til det totale momentbidraget. Fra ligning 7.15 kan man identifisere de variablene som varierer fra seksjon til seksjon. Det som varierer er arealet av seksjonen, samt avstand og vinkelutslag fra massesentrum slik det er vist i figur 7.15. I tillegg bruker vi dragkoeffisienten til den siden de enkelte seksjonene tilhører. Det totalet momentbidraget kan derfor beregnes ved summasjon av alle seksjoner med tilhørende variabler. Bidraget fra fremsiden er en numerisk integrasjon over halve siden, som videre er multiplisert med 2. Bidraget fra sideflaten er en numerisk integrasjon over en hel side. Prinsippet er forklart i figur 7.16. Momentligningen blir for gir-rotasjon:

$$\sum \tau = \frac{1}{2} \cdot \rho \cdot \omega \cdot |\omega| \cdot \left(2 \cdot C_{d,front} \cdot \sum_{i=1}^{5} A_i \cdot \vec{r_i}^{\,3} \cdot sin^3(\theta_i) + C_{d,side} \cdot \sum_{i=5'}^{5} A_i \cdot \vec{r_i}^{\,3} \cdot sin^3(\theta_i) \right) \quad (7.16)$$

Seksjonene er delt opp ved hjelp av *Fusion 360*, og arealene kan leses av direkte i programmet. Avstanden fra massesentrum finner man også i programmet. Dette gjøres ved å måle avstanden fra massesentrum, til midtpunktet av hver seksjon når man ser ROV-en ovenfra. Vinkelutslaget regnes direkte når man leser av lengden på posisjonsvektoren. Videre definerer vi to bidragsfaktorer som tilsvarer de to summasjonene i ligning 7.16. Bidragsfaktoren er ikke en praktisk betydningsfull verdi, men beskriver størrelsene som varierer for hver seksjon, som vist i figur 7.16. Tabell 7.3 og 7.4 viser verdien av alle variablene tilknyttet de ulike seksjonene vi har delt overflatene opp i. I tillegg har vi regnet ut multiplikasjonen av disse, samt den totale summen av den numeriske integrasjonen.

Tabell 7.3: Oversikt over numerisk integrasjonsbidrag fra fremside. Summen av alle bidragene må her multipliseres med 2 for å finne den definerte bidragsfaktoren fra fremsiden under gir-rotasjon.

i	$A_i[mm^2]$	$r_i[mm]$	$\theta_i[^o]$	$A_i \cdot r_i^3 \cdot sin^3(\theta_i)[m^4]$
1	19535	321,3	5,80	$6,68707 \cdot 10^{-7}$
2	7841	330,7	14,80	$4,72683 \cdot 10^{-6}$
3	6224	344,9	22,00	$1,34238 \cdot 10^{-5}$
4	6949	368	29,70	$4,212 \cdot 10^{-5}$
5	7603	388,9	34,70	$8,25041 \cdot 10^{-5}$
6	7474	432,3	42,30	0,000184068
Sum				0,000327511

Tabell 7.4: Oversikt over numerisk integrasjonsbidrag fra siden. Summen i dette tilfellet vil direkte gi den definerte bidragsfaktoren fra siden under gir-rotasjon.

i	$A_i[mm^2]$	$r_i[mm]$	$\theta_i[^o]$	$A_i \cdot r_i^3 \cdot sin^3(\theta_i)[m^4]$
1	22617	235,4	8,20	$8,56007 \cdot 10^{-7}$
2	25492	254	23,50	$2,64853 \cdot 10^{-5}$
3	18972	287,5	35,90	$9,08965 \cdot 10^{-5}$
4	17247	331,6	45,40	0,00022701
5	20237	382,5	52,50	0,00056551
1’	22617	235,4	8,20	$8,56007 \cdot 10^{-7}$
2’	25687	254	23,50	$2,66878 \cdot 10^{-5}$
3’	23854	287,5	35,90	0,000114287
4’	24543	331,6	45,40	0,000323042
5’	23297	382,5	52,50	0,00065102
Sum				0,00202665

For rull- og stamprotasjon er det brukt nøyaktig samme fremgangsmåte som i eksempelet over. Alle sider er delt opp i seksjoner, og de tilhørende variablene er funnet i *Fusion 360*. Videre har vi regnet ut tilsvarende bidragsfaktorer som i tabell 7.3 og 7.4 for begge rotasjonsbevegelsene. Tabell 7.5 viser en oversikt over alle de totale bidragsfaktorene vi har regnet oss frem til.

Tabell 7.5: Oversikt over de overnevnte bidragsfaktorene til det totale momentet fra dragkraften i hver rotasjonsretning. Overflaten som står vinkelrett på rotasjonsaksen antas å ikke oppleve vannmotstand, og har derfor ingen bidragsfaktor.

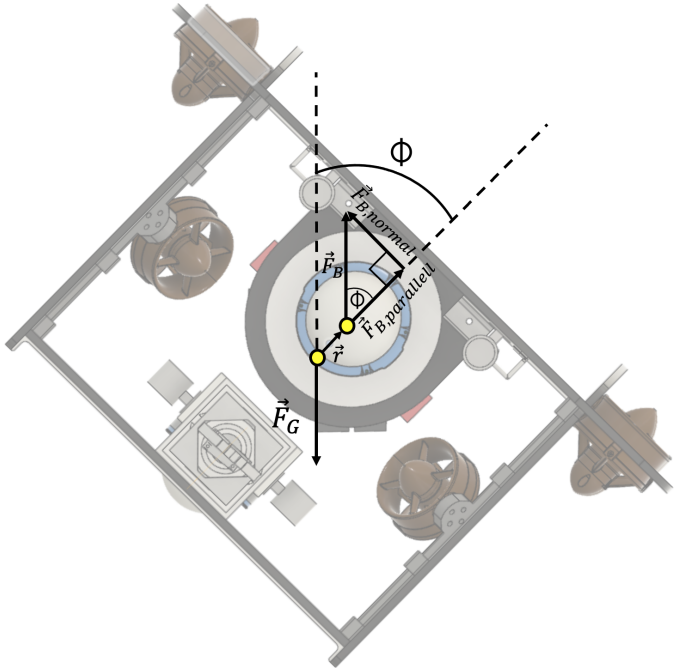
Bidragsfaktor	Rull	Stamp	Gir
Fremside	-	0,000170539	0,00327511
Side	0,000429746	-	0,00202665
Topp	0,000709818	0,002722379	-

Til slutt finner man det totale momentbidraget fra vannmotstand ved å erstatte summasjonene i ligning 7.16 med de utregnede bidragsfaktorene for hver retning. For gir-rotasjon blir vannmotstanden derfor:

$$\sum \tau = \frac{1}{2} \cdot \rho \cdot \omega \cdot |\omega| \cdot (2 \cdot C_{d,front} \cdot B_{gir,front} + C_{d,side} \cdot B_{gir,side}) \tag{7.17}$$

7.3.4 Flytestabilitet

Under design av en farkost, ønsker man vanligvis at den skal ha en viss grad av flytestabilitet. Kriteriet for flytestabilitet er at farkostens volumsenter skal ligge loddrett over massesenteret. Dette kommer av at oppdriftskraften virker fra volumsenteret, og gravitasjonskraften virker fra massesenteret. Når farkosten utsettes for forstyrrelser, vil volumsenteret bevege seg bort fra den loddrette linjen over massesenteret, og teoretisk sett er farkosten i en ustabil tilstand. Ved at ROV-en er designet med et volumsenter som ligger høyere enn massesenteret, får man et naturlig opprettingsbidrag i stedet for at hele ROV-en veltes rundt. Prinsippet er vist i figuren under:



Figur 7.17: Dersom ROV-en utsettes for for krefter som tilter den i rull- eller stampretning, vil volumsenteret forskyves bort fra den loddrette linjen over ROV-ens massesenter. Ved å dekomponere oppdriftskraften i forhold til volumsenterets posisjonsvektor $\vec{r}$ fra massesenteret, kan man se at normalkomponenten vil gi et momentbidrag på rotasjonen.

Flytestabiliteten vil gjøre at det totale momentbidraget ikke bare avhenger av ROV-ens rotasjonshastighet, men også av vinkelposisjonen den befinner seg i. Dette gjelder likevel kun for rull- og stamprotasjon. Under gir-rotasjon ligger både masse- og volumsenteret på z-aksen, som er aksen ROV-en roterer rundt. Det betyr at plasseringen av volumsenteret ikke endres, og ROV-en kan betraktes som stabil. For å regne ut momentbidraget fra naturlig oppretting, finner man normalkomponenten til oppdriftskraften vist i figur 7.17. Det at bare oppdriftskraften, og ikke gravitasjonskraften, har betydning for momentbidraget, er et resultat av at vi har definert massesenteret som rotasjonssenter.

$$\tau_{oppretting} = F_{B,normal} \cdot r = F_B \cdot r_{COM-B} \cdot sin(\phi) \tag{7.18}$$

Hvor

- F_B er den totale oppdriftskraften som virker på ROV-en
- r_{COM-B} er størrelsen av posisjonsvektoren fra massesenter til volumsenter

- ϕ er vinkelutslaget ROV-en er tiltet med i rull eller stamp.

Momentbidraget fra naturlig oppretting blir størst mulig ved en stor total oppdriftskraft, og stor avstand mellom volum- og massesenter. Den totale oppdriftskraften er 184,4 N og avstanden til volumsenteret er 2,06 cm. Begge verdiene har rammegruppen regnet ut fra CAD-modellen i *Inventor*. I tillegg kan man fra ligningen se at momentbidraget fra oppretting vil være størst mulig dersom ROV-en har et vinkelutslag på 90^o. Da virker oppdriftskraften normalt på momentarmen, og bidrar med full effekt. Dette tilsvarer et maksimalt moment på 3.80 Nm.

7.3.5 Simulering av modeller

Alle momentbidragene legges videre inn i evivalenten til Newtons 2. lov for rotasjon. Momentet fra thrusterne legges inn med positivt fortegn, da fortegnet på den totale thrusterkraften styrer hvilken retning momentbidraget går. Momentbidraget fra dragmotstand legges inn med negativt fortegn. Dette bidraget virker alltid mot bevegelsesretningen. Til slutt legges også opprettingsbidraget inn med negativt fortegn. Dette virker alltid motsatt vei av vinkelutslaget på ROV-en.

$$\tau_{thrustere} - \tau_{vannmotstand} - \underbrace{\tau_{oppretting}}_{Rull\ og\ stamp} = I \cdot \alpha \tag{7.19}$$

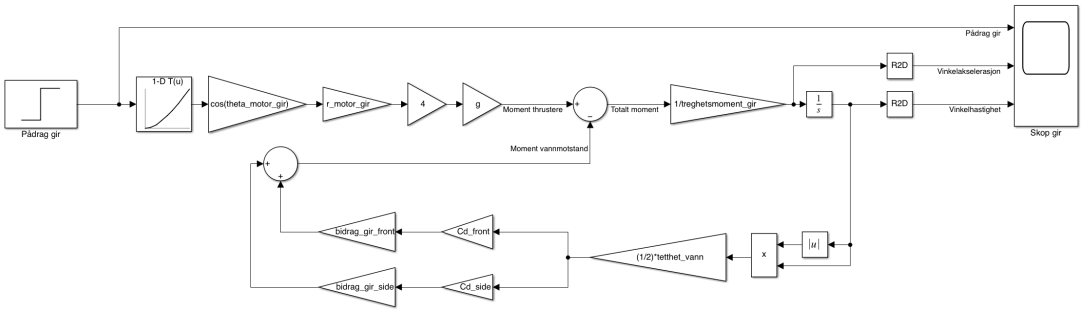
For translatorisk bevegelse endte vi opp med en ligning avhengig av både hastighet og akselerasjon. For rotasjon ender vi opp med at ligningen for rull og stamp er avhengig av både vinkelakselerasjon, vinkelhastighet og vinkel. Hastighet er den tidsderiverte av posisjonen, og akselerasjonen er igjen den tidsderiverte av hastigheten. Den totale momentligningen blir derfor:

$$4 \cdot F_T \cdot r_T \cdot cos(90^o - \theta_T) - \frac{1}{2} \cdot \rho \cdot \omega \cdot |\omega| \cdot (2 \cdot C_{d,1} \cdot B_1 + C_{d,2} \cdot B_2) - \underbrace{F_B \cdot r_{COM-B} \cdot sin(\phi)}_{Rull\ og\ stamp} = I \cdot \alpha \tag{7.20}$$

Differensialligningene for alle frihetsgrader løser vi igjen ved bruk av *Simulink*.

Modell for gir

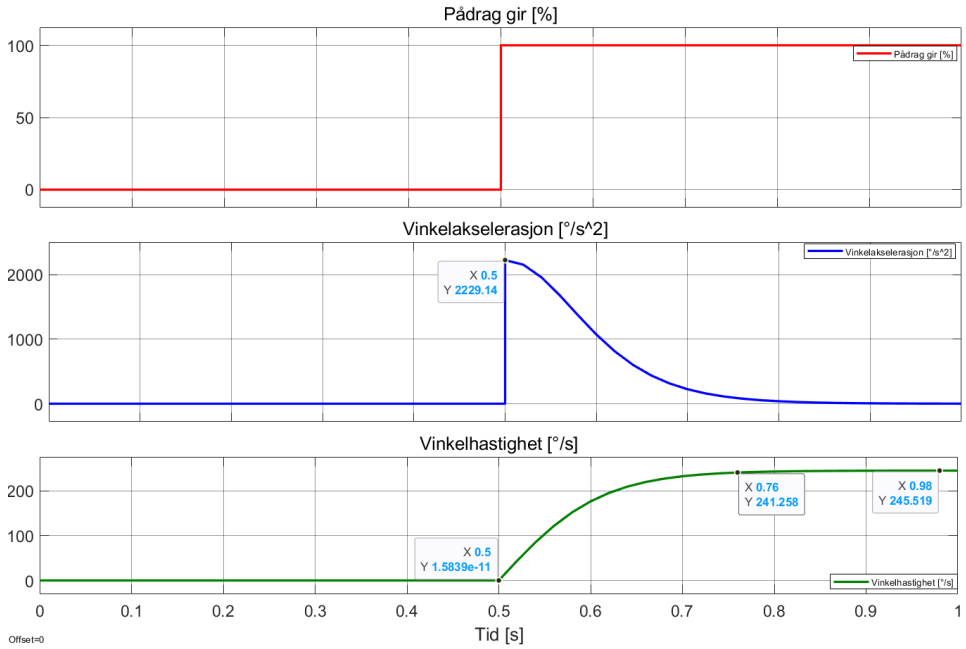
Modellen for gir er, som for jag og svai, kun for å få et inntrykk av hvordan ROV-en beveger seg i vannet. Modellen skal ikke brukes til regulering, da vi kun regulerer rull og stamp. Figur 7.18 viser Simulink-modellen.



Figur 7.18: Girmodell i Simulink.

Momentbidraget fra thrusterne tar hensyn til vinkelorientering og avstand fra rotasjonssentrum gjennom *gain*-blokkene etter motorkarakteristikken. Momentbidraget fra vannmotstand har vi valgt å regne ut ved å bruke *gain*-blokker på fellesvariabler, for så å dele signalet opp i to grener. I hver gren multipliseres signalet med bidragsfaktoren fra den aktuelle sideflaten, samt tilhørende dragkoeffisient. Videre summeres signalene sammen for å få totalt momentbidrag. For å få et mer brukervennlig skop, er det tatt i bruk en *R2D*-blokk som regner om vinklene fra radianer til grader. Variablene i ligning 7.20, er satt inn med relevante verdier for gir-rotasjon.

For å se hvor fort ROV-en klarer å rotere har vi simulert modellen med et pådragssprang fra 0 til 100 %. Dette gjør at alle thrustere bidrar i positiv gir-retning. Spranget går etter 0.5 sekunder, og modellen er simulert i 1 sekund. Skopet i figur 7.19 viser resultatet av simuleringen.



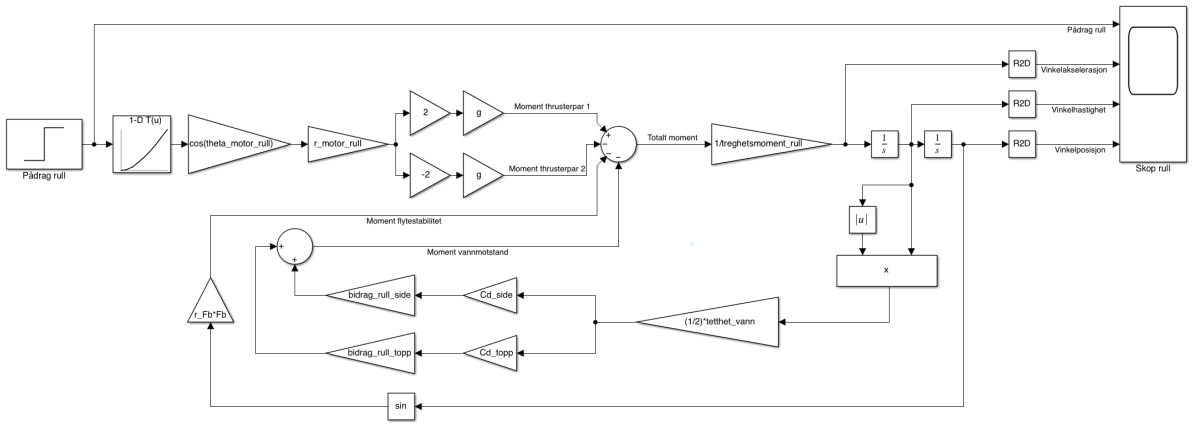
Figur 7.19: Skopet etter simulering av girmodellen. Vinkelakselerasjonen og vinkelhastigheten målt basert på grader, og ikke radianer.

Fra skopet kan vi lese av vinkelhastigheten ROV-en maksimalt kan oppnå. Hastigheten den stabiliserer seg på ifølge modellen, er 246 °/s. Det tar heller ikke mer enn ca. 0.26 sekunder fra spranget går, til den når 98% av topphastighet. ROV-en kommer ikke til å operere rundt denne hastigheten, men simuleringen vil hjelpe med å indikere hvor mye thrusterpådraget fra gir-kommando må nedskaleres. Nedskaleringen må gjøres for at piloten skal få en enklere oppgave, og

at pådraget ikke blir for sensitivt i forhold til styrestikken. Med denne topphastigheten vil ROV-en uten nedskalering av thrusterpådrag kunne rotere omtrent 49.1° i løpet av kameraforsinkelsen på 200 ms.

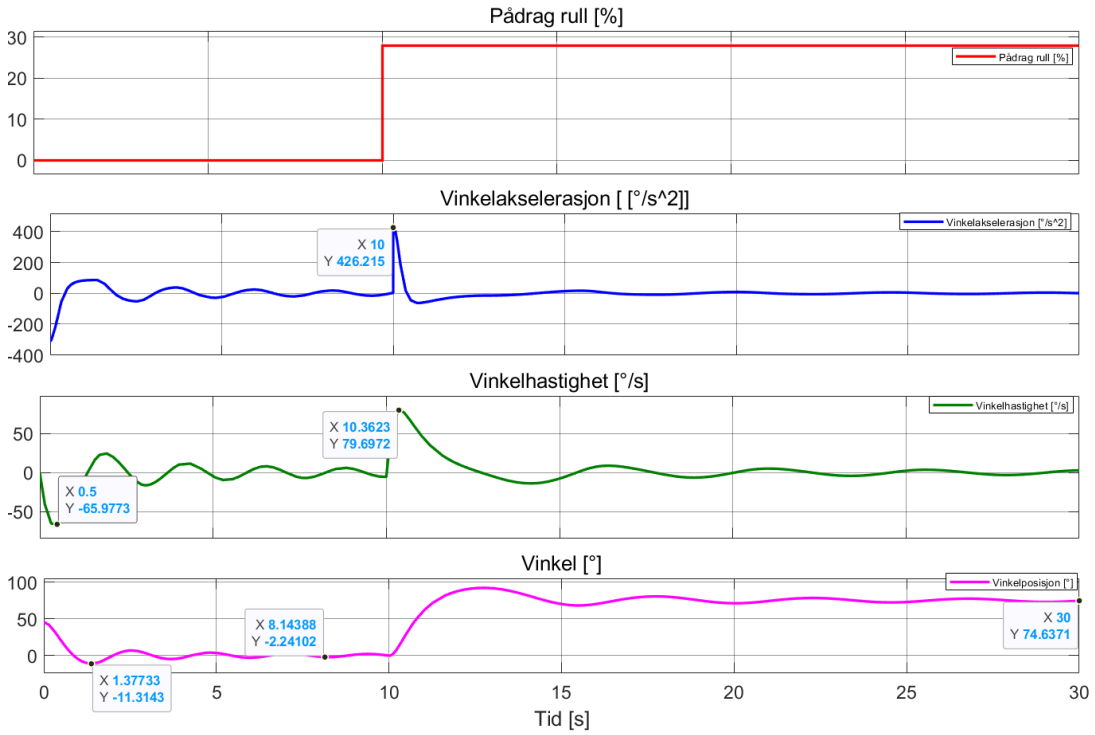
Modell for rull

Modellen for rull skal gi et utgangspunkt for regulatoren vi senere skal sette opp. Momentbidraget fra thrustere er satt opp med en gren for hvert thrusterpar. Den ene grenen multipliseres med -2, og er derfor satt opp med subtraksjon i *sum*-blokken. Thrusterparene bidrar altså med samme moment til totalsummen, men i modellen er de satt opp med motsatt fortegn for å vise at de kjører forskjellig retning. Videre er momentbidraget fra vannmotstand er lagt inn på samme måte som for gir. Modellen for rull er også avhengig av vinkelposisjonen til ROV-en gjennom flytestabilitet. Derfor er det lagt inn en ekstra integrator-blokk som integrerer vinkelhastigheten om til vinkelposisjon. Denne tilbakekobles via en sinus-blokk, og en *gain* som multipliserer signalet med oppdriftskraften og avstanden mellom massesenter og volumsenter. Momentbidraget fra flytestabilitet subtraheres fra totalsummen, da den alltid virker motsatt retning av den vinkelposisjonen ROV-en befinner seg i. Modellen er vist i figur 7.20.



Figur 7.20: Rullmodell i Simulink.

I rull-retning ønsker vi ikke at ROV-en skal gjøre fullstendige rotasjoner. Det er derfor ikke relevant å simulere med et pådragssprang som gjør at vi kan lese av en topphastighet. Mer interessant er det å se hvor god flytestabiliteten til ROV-en er ifølge modellen. Simuleringen er gjort med en initialverdi på 45° i posisjonsintegratoren. Dette gjør at man kan observere hvor fort ROV-en stabiliserer seg av naturlig oppretting. Videre er det satt på et pådragssprang som går etter at den ligger noenlunde stabilt i vannet. Spranget går fra 0 til 28 %. Denne verdien er funnet ved prøving og feiling under simulering. Dette er et sprang som ikke tillater ROV-en å rotere. I praksis tilsvarer det at thrusterpådraget ikke er stort nok til å overgå momentbidraget fra flytestabilitet. Resultatet etter simuleringen er vist i figur 7.21.

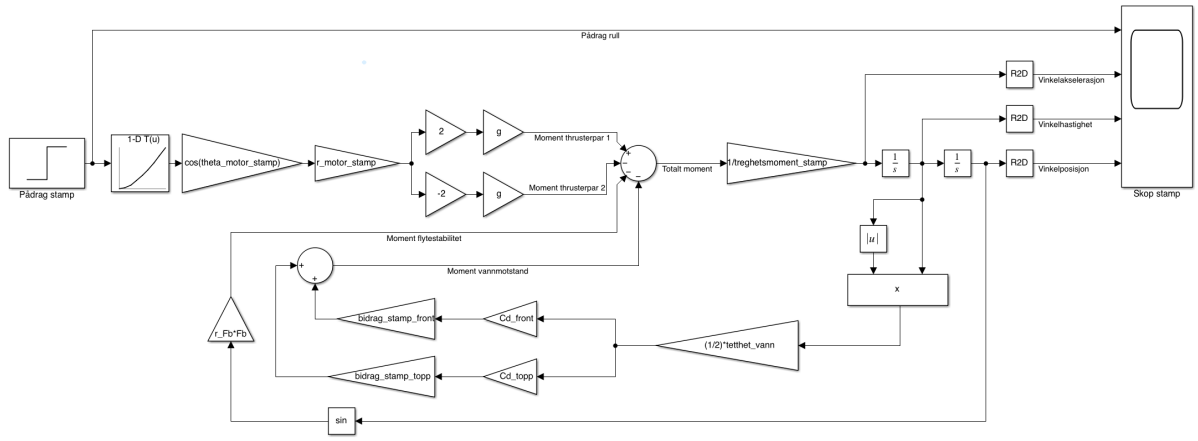


Figur 7.21: Skopet etter simulering av rullmodellen. I dette tilfellet er også posisjonen fremvist.

I første del av simuleringen ser man effekten av naturlig oppretting. Ved et initielt utslag på 45° tar det omtrent 8.1 sekunder for ROV-en å rette seg opp til ± 5% av vinkelutslaget uten hjelp fra thrustere. I tillegg opplever ROV-en et undersving på ca. 11.3° under opprettingen. Pådragsspranget på thrusterne går etter 10 sekunder, og ROV-en stabiliserer seg etterhvert rundt 74.6° ved dette pådraget. Simuleringen gir oss forhåpentligvis et godt bilde på hvordan ROV-en vil oppføre seg. Det kan se ut som at kortvarige forstyrrelser i vinkelutslag vil kunne kompenseres godt for kun fra naturlig oppretting. Likevel vil det være viktig å konstruere en god regulator. Vi antar at under translatorisk bevegelse vil motstanden fra vannet påvirke annerledes på ulike deler av overflaten på ROV-en. Dette vil igjen kunne føre til en kontinuerlig forstyrrelse i vinkelposisjonen. Regulatoren blir derfor viktig for å motvirke effekter som dette. Modellen gir oss også i dette tilfellet et inntrykk av hvordan vi skal nedskalere pådraget på thrusterne under reguleringen.

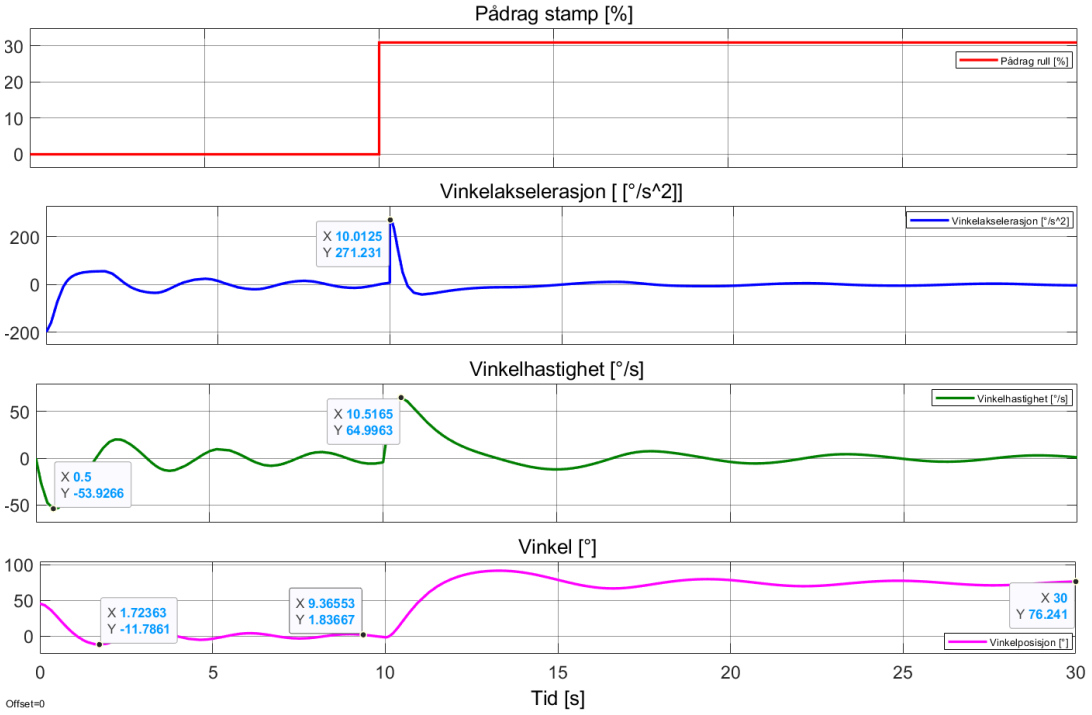
Modell for stamp

Modellen for stamp er laget i *Simulink* på nøyaktig samme måte som modellen for rull. Alle utregninger er de samme, men variablene er byttet ut. Modellen er vist i figur 7.22.



Figur 7.22: Stampmodell i Simulink.

Simuleringen er også gjort på samme måte som for rull. Forskjellen ligger i pådraget, som nå går fra 0 til 31 %. Grunnen til at pådraget kan være litt høyere i stampretning er at vannmotstanden er litt større. Momentbidraget fra oppretting er det samme, men vannmotstanden tillater ikke ROV-en å komme opp i like stor hastighet under stamprotasjon. I tillegg står thrusterene med en spissere vinkel mot momentarmen enn i rull-retning. Momentarmen er også kortere. Alle bidragene er små, men de gjør altså at pådraget kan økes litt. Resultatet av simuleringen er vist i skopet i figur 7.23.



Figur 7.23: Skopet etter simulering av stampmodellen.

Opprettingen i starten av simuleringen tar i dette tilfellet omtrent 9.4 sekunder før vinkelen er innenfor ± 5% av vinkelutslaget. Dette er litt lenger enn for rull. Undersvinget er i dette tilfellet ca. 11.8°. Etter at pådragsspranget går, stabiliserer ROV-en seg etterhvert rundt 76.2°. Modellen for stamp gir hovedsaklig samme type informasjon som modellen for rull gav. Den indikerer også

at regulatoren for stamp vil måtte være hakket hissigere enn regulatoren for rull. Dette kommer av at thrusterne trenger litt mer pådrag for å gi like stort momentbidrag, og at den naturlige opprettingen tar litt lenger tid.

7.4 Konklusjon

I dette kapittelet har vi presentert den teoretiske utledningen av matematiske modeller som beskriver hvordan ROV-en oppfører seg ved beveglese i de forskjellige frihetsgradene.

Å modellere et komplekst system som dette, innebærer mange utfordringer. For eksempel er vannmotstand og treghetsmoment fysiske størrelser som er vanskelige å fastslå teoretisk med stor nøyaktighet. Det er ved hjelp av simulering funnet drag-koeffisienter for bevegelse i translasjonsretningene. Ved help av en 3D-modell av ROV-en er det også funnet treghetsmomenter ved rotasjon om de ulike aksene. Verdiene fra simuleringen og fra 3D-modellen er sammenlignet med andre forenklede metoder, og virket å kunne stemme godt. Selv om det er gjort et forsøk på å estimere ulike størrelser, vil modellene inneholde store usikkerheter basert på forenklingene som er gjort. De største forenklingene og antakelsene som er gjort er vist under:

- La være å ta hensyn til vanndynamikk, som for eksempel kan være vannturbulens.
- Alle fysiske mål på ROV-en er estimert ut fra 3D-modeller.
- Masse og volum av ROV-en er estimert.
- Posisjonen av massesenter og volumsenter er også estimert.
- Treghetsmoment og dragkoeffisienter er estimert.
- Det å vurdere hver enkelt frihetsgrad isolert sett er en forenkling. Bevegelse i én frihetsgrad, vil påvirke bevegelse i de andre.

Selv om disse forenklingene er gjort, vil modellene gi oss en indikasjon på hvordan ROV-ens bevegelse vil være. De legger også grunnlaget for å i neste kapittel kunne sette opp regulatorer for ønskede frihetsgrader.

Simuleringsfilene for modellene i Simulink kan lastes ned via linken i vedlegg A.

Kapittel 8

Regulering

Kapitteloversikt

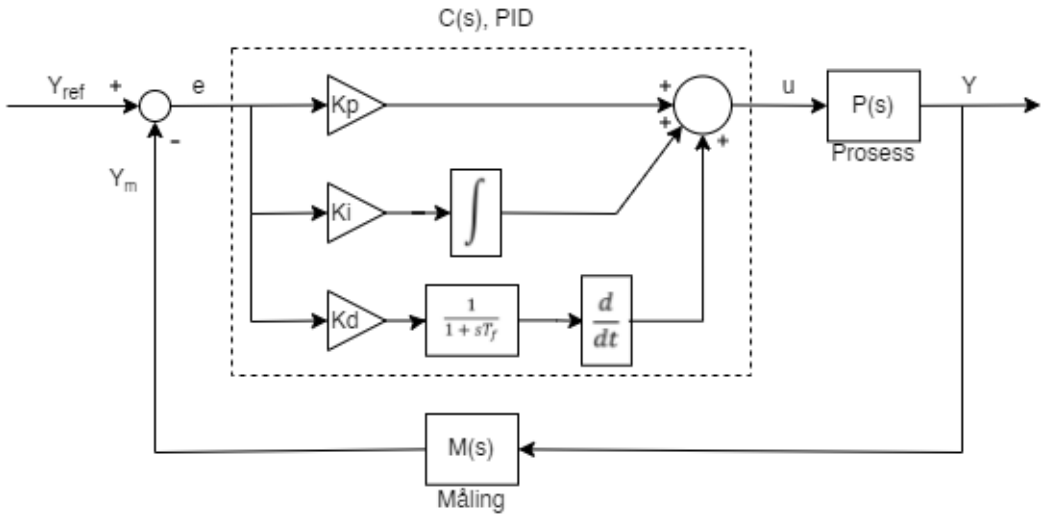
I dette kapittelet skal reguleringsystemet for ROV-en utvikles. Det er ønskelig at ROV-en er i stand til å regulere frihetsgradene hiv, rull og stamp presentert i delkapittel 6.1.1, og dermed kunne være stabil i vannet uavhengig av om den står i ro eller kjøres. Reguleringssystemet skal realiseres ved å implementere 3 individuelle regulatorer som regulerer hver sin frihetsgrad. Disse settes opp basert på de matematiske modellene utledet i kapittel 6. Kapittelet vil også omhandle implemetering av systemet på mikrokontrolleren. Avslutningsvis vil det presenteres en test av ferdige systemet, altså ROV-en, gjort fysisk i et basseng.

8.1 Overordnet reguleringssystem

Ettersom sensorgruppen legger opp til måling av rull- og stampvinkel, samt dybde i vannet, er det mulig å bruke tilbakekobling for å realisere reguleringen av de aktuelle frihetsgradene. Reguleringen baseres på prinsippet med å sammenligne den målte dybden eller vinkelen fra sensoren, med en ønsket referanseverdi. Videre beregnes det et thrusterpådrag basert på hvor stort avvik det er mellom de to verdiene. Pådraget skal sørge for at ROV-en manøvreres for å redusere, og helst eliminere, dette avviket.

Regulatorene som skal tas i bruk er av typen PID. Det finnes også andre typer regulatorer som alternativt kunne vært tatt i bruk, men ettersom vi fra tidligere i utdanningsløpet har jobbet med PID-regulatorer, og hatt god erfaring med disse, velger vi å ikke bruke tid på å sette oss inn i andre typer regulatorer. Dette valget tas på grunn av tidsbegrensningen ved prosjektet.

Oppsettet for en standard PID-regulator på parallell form er vist i figur 8.1.



Figur 8.1: Standard modell for PID-regulator

På inngangen til PID-regulatoren sendes avviket mellom en satt referanseverdi, Y_{ref}, og målt verdi på utgangen av en prosess, Y_m. Basert på avviket, regner regulatoren ut et pådrag, u, som sendes videre inn på prosessen.

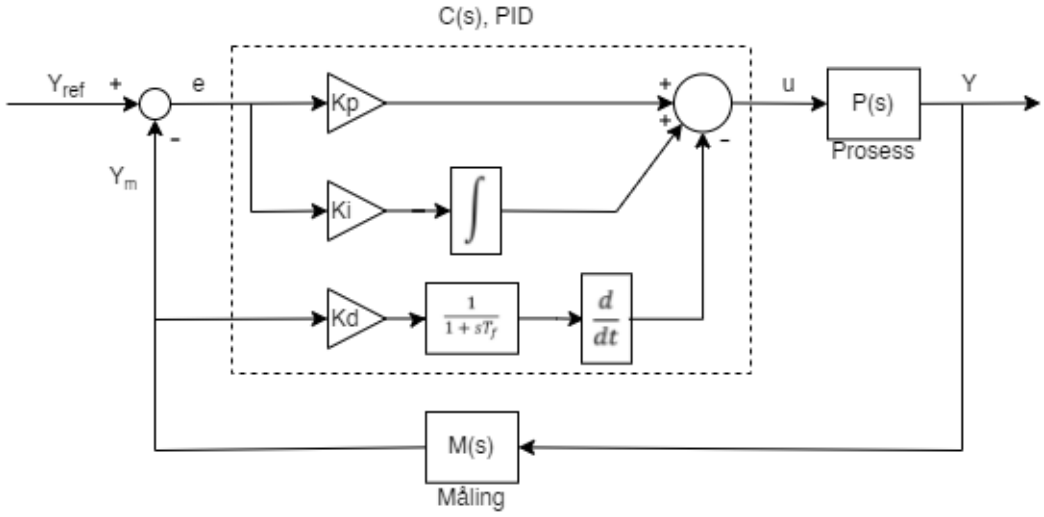
Selve PID-regulatoren består altså av tre ledd: et proporsjonalledd, et integralledd og et derivatledd. I proporsjonalleddet multipliseres avviket direkte med en konstant K_p, og pådraget får et bidrag som er proporsjonalt med avviket på inngangen. I integralleddet multipliseres avviket med konstanten K_i, før bidraget videre integreres. Bidraget til pådrag fra integralleddet er altså avhengig av hvor stort avviket er, men også hvor lenge avviket har eksistert. I derivatleddet multipliseres avviket med konstanten K_d, og bidraget deriveres. Bidraget til pådrag vil altså her avhenge av endringsraten til avviket. Det er også lagt inn et lavpassfilter med tidskonstant T_f på derivatleddet for å unngå at støy fra målingene gir et kraftig bidrag til pådraget. Tidkonstanten finnes det ulike metoder for å bestemme, og denne vil forklares senere i kapittelet. Til slutt vil de tre bidragene i regulatoren summeres, og til sammen utgjøre pådraget som sendes til prosessen. For ROV-en tilsvarer dette et thrusterpådrag. Pådraget u regnes altså på følgende måte:

$$u(t) = K_p \cdot e(t) + K_i \int e(t)dt + K_d \cdot \frac{de_f(t)}{dt} \tag{8.1}$$

Hvor:

- e(t) er avviket.
- u(t) er pådraget.
- e_f(t) er det filtrerte avviket.

For ROV-en kan det være aktuelt å sette referanseverdier under kjøring, og dette vil kunne føre til sprangendringer i avviket. For å unngå at et referansesprang fører til en brå endring i avviket, og et stort utslag på derivatleddet, er det derfor vanlig å bruke et alternativt oppsett for PID-regulatoren, vist i figur 8.2.



Figur 8.2: Modell for PID regulatoren brukt av oss.

Derivatleddet regnes ut basert på den deriverte av målt verdi fremfor den deriverte av avviket. Dette kan gjøres på grunn av at referansen som oftest ligger på et konstant nivå, og endringen i avviket er lik endringen i målt verdi på utgangen av prosessen. Dersom referansen skulle endre seg, ønsker man som sagt ikke at derivatleddet skal reagere på endringen uansett. Da man ikke lenger subtraherer målt verdi fra referansen, må det også legges inn en faktor på -1 før derivatleddet regnes ut. Faktoren på -1 i derivatleddet er implementert med negativt fortegn i summasjonen av alle bidragene. Pådraget regnes da på følgende måte:

$$u(t) = K_p \cdot e(t) + K_i \int e(t)dt - K_d \cdot \frac{dY_{m,f}(t)}{dt} \tag{8.2}$$

Hvor $Y_{m,f}$ er den filtrerte måleverdien.

PID-regulatoren er relativt enkel å ta i bruk, men utfordringen ligger i å bestemme konstantene, også kalt regulatorparametrene, K_p, K_i og K_d for å oppnå en best mulig oppførsel av systemet. Det fins utallige måter å bestemme disse parametrene på, og i neste delkapittel vil noen av disse metodene presenteres. Ved å bruke flere av disse metodene, og teste forskjellige sett med regulatorparametre, er målet å ende opp med et godt fungerende reguleringssystem i ROV-en.

8.1.1 Parameterbestemmelse

For at regulatoren skal kunne drive aktiv stabilitetskontroll av ROV-en, må parameterene K_p, K_i og K_d bestemmes. Det finnes flere metoder for å bestemme disse parameterene. Noen av metodene baserer seg på å først utlede en overføringsfunksjon for prosessen som skal reguleres. Ut fra denne kan man bestemme teoretiske verdier på parametrene ved bruk av etablerte tabelloppslag for ulike typer overføringsfunksjoner. Andre metoder baserer seg på å implementere en standard PID-regulator før man bestemmer parametrene, for så å gjøre tester på det fysiske systemet og bestemme parametrene etter systemets oppførsel. Parametrene vil tunes helt til man oppnår en ønsket respons fra systemet. Det finnes også metoder hvor man kombinerer de to prinsippene. Først gjøres teoretiske beregninger, og videre justeres parametrene for å optimalisere systemets respons ytterligere. Videre presenteres noen etablerte metoder for parameterbestemmelse som er mye brukt i forbindelse med PID-regulatorer.

Skogestads metode

En kjent og velbrukt metode for å bestemme regulatorparametere er Skogestads metode [18]. Det overordnede målet med Skogestads metode er at følgeforholdet M(s), som ble vist i figur 8.2, skal være en 1.ordens transferfunksjon med dødtid. For å realisere dette, har Skogestad utledet en tabell for bestemmelse av parameterene ut fra prosessens overføringsfunksjon. Skogestads metode kan benyttes for reguleringssystemet i ROV-en ved å bruke differensialligningene funnet i kapittel 7, og finne overføringsfunksjonene for prosessene som skal reguleres ut fra disse. Dermed kan parameterene bestemmes ved å bruke tabelloppslaget utledet av Skogestad.

I en artikkel av Finn Haugen [18] er tabelloppslaget for parameterbestemmelse til flere typer overføringsfunksjoner gjengitt. Tabellen er vist i figur 8.3.

Process type	$H_{psf}(s)$ (process)	K_p	T_i	T_d
Integrator + delay	$\frac{K}{s}e^{-\tau s}$	$\frac{1}{K(T_C+\tau)}$	$c\,(T_C+\tau)$	0
Time-constant + delay	$\frac{K}{Ts+1}e^{-\tau s}$	$\frac{T}{K(T_C+\tau)}$	$\min\left[T,\, c\,(T_C+\tau)\right]$	0
Integr + time-const + del.	$\frac{K}{(Ts+1)s}e^{-\tau s}$	$\frac{1}{K(T_C+\tau)}$	$c\,(T_C+\tau)$	T
Two time-const + delay	$\frac{K}{(T_1s+1)(T_2s+1)}e^{-\tau s}$	$\frac{T_1}{K(T_C+\tau)}$	$\min\left[T_1,\, c\,(T_C+\tau)\right]$	T_2
Double integrator + delay	$\frac{K}{s^2}e^{-\tau s}$	$\frac{1}{4K(T_C+\tau)^2}$	$4\,(T_C+\tau)$	$4\,(T_C+\tau)$

Figur 8.3: Regulatorparametere for forskjellige overføringsfunksjoner. Hentet fra [18].

Hvor:

- c er en faktor som settes til 4 for gode følgeegenskaper, og 1.5 for bedre forstyrrelseskompensasjon.
- T_C er det regulerte systemets responstid

- τ er dødtiden før en endring måles av systemet.

Parametrene T_i og T_d har direkte sammenheng med K_i og K_d som vi har valgt å bruke. Denne sammenhengen vises i formlene under:

$$K_i = \frac{K_p}{T_i} \ , \quad K_d = K_p \cdot T_d \tag{8.3}$$

Dødtiden τ må enten måles på det fysiske systemet, eller estimeres. Ettersom det er utfordrende å gjøre nøyaktige målinger på dødtiden til systemet i ROV-en, har denne blitt estimert. Måledataene fra sensorgruppen sendes med en frekvens på 20 Hz, og reguleringen skjer med samme frekvens. Det er derfor mulig å få en forsinkelse gjennom reguleringsløkken på opp mot 0.1 s. Forsinkelsen gjennom motorkontrolleren og frem til thrusteren gir et målbart pådrag, er ukjent. Den totale dødtiden, τ, estimeres derfor for ROV-en til å være det dobbelte av dødtiden gjennom reguleringsløkken, altså 0.2 s. Denne dødtiden blir satt som gjeldende for alle frihetsgradene. Skogestad anbefaler videre å sette $T_C = \tau$, som er gjort for de teoretiske utregningene i dette kapittelet. Faktoren c er valgt til å være 4, da dette også er et standardvalg anbefalt av Skogestad [18].

Verdt å merke seg, er at parameterene fra Skogestads metode er gitt for en PID-regulator på seriell form. Kort fortalt vil en PID-regulator på seriell form bruke P-leddet som inngangsverdi på I-leddet, og den summerte av disse som inngang på D-leddet. I- og D-leddet får altså ikke avviket direkte inn på inngangen. Parameterene må på grunn av dette omformes for bruk i en parallell PID-regulator, som ble vist i figur 8.1. Omformingen skjer ved bruk av formlene under:

$$K_{p_p} = K_{p_s} \cdot \left(1 + \frac{T_{d_s}}{T_{i_s}}\right) \tag{8.4}$$

$$T_{i_p} = T_{i_s} \cdot \left(1 + \frac{T_{d_s}}{T_{i_s}}\right) \tag{8.5}$$

$$T_{d_p} = T_{d_s} \cdot \frac{1}{1 + \frac{T_{d_s}}{T_{i_s}}} \tag{8.6}$$

Videre må T_i og T_d omformes til K_i og K_d ved hjelp av ligning 8.3 for at parametrene skal kunne brukes på den formen vi har valgt å sette opp regulatoren på.

Ziegler-Nichols

En annen metode for å bestemme parameterene til regulatoren, er en metode utviklet av John G. Ziegler og Nathaniel B. Nichols [58]. Denne baserer seg på å utføre tester med systemet hvor en PID-regulator allerede er implementert før parametrene er bestemt. De endelige parametrene bestemmes ved å følge en gitt prosedyre for justering av midlertidige parametre. I første steg settes PID-regulatoren opp med parametrene K_i og K_d lik null. Videre skal K_p økes til systemet nesten oppnår ustabilitet, og det oppstår vedvarende og konstante oscillasjoner på utgangen til systemet. K_p-verdien som gir vedvarende oscillasjoner kalles for K_{pu}. I tillegg skal perioden på de samme oscillasjonene som oppstår måles. Denne perioden kalles for T_u. De endelige parametrene bestemmes ut fra K_{pu} og T_u.

For en standard PID-regulator blir parametrene beregnet på følgende måte:

$$K_p = 0.6 \cdot K_{pu} \ , \quad K_i = \frac{1.2 \cdot K_{pu}}{T_u} \ , \quad K_d = \frac{3 \cdot K_{pu} \cdot T_u}{40} \tag{8.7}$$

Formålet med metoden er at responsens oscillasjoner på utgangen dempes med 25 % for hver periode, og ligningene over er utledet basert på dette.

Det at metoden baseres på å drive et system på grensen til ustabilitet med vedvarende oscillasjoner, kan by på utfordringer for visse systemer. For ROV-en kan for eksempel det å drive hiv-prosessen til vi oppnår store oscillasjoner føre til uønsket store bevegelser i vannet, og et potensielt utfall kan bli en kollisjon i bassenget. Metoden kan likevel prøves ut gjennom simulering av modellen, og videre kan de oppnådde parametrene vurderes opp mot parametersettene fra de andre metodene.

Parameterstyring

For noen systemer kan hele eller deler av den regulerte prosessen oppføre seg ulineært, og det kan følgelig være behov for å regulere systemet ulikt i gitte arbeidspunkter. Et slikt behov kan skape problemer for en standard PID-regulator ettersom parametererene gjerne er beregnet for et lineært system, eller optimalisert for ett spesifikt arbeidspunkt av prosessen. Derfor kan en ulineær prosess føre til at regulatoren virker mindre optimalt, eller i verste fall fører til ustabilitet. En løsning på problemet er å bestemme flere parametersett, alt etter hvilket arbeidspunkt prosessen befinner seg i. Regulatorparametrene vil da endres etter hvert som systemet beveger seg mellom de ulike arbeidspunktene. Denne metoden kalles parameterstyring [27].

Et arbeidspunkt er definert som en gitt tilstand av systemet. For systemer der endringer i prosessen skjer stegvis, vil arbeidspunkter være klart adskilt fra hverandre. Man vil da enkelt kunne bestemme et parametersett for hvert av disse arbeidspunktene ved bruk av parameterstyring. For andre systemer kan arbeidspunktet variere kontinuerlig. ROV-en vår er et eksempel på et ulineært system, hvor arbeidspunktet endres kontinuerlig i prosessen. Ulineariteten forårsakes av at vannmotstanden som påvirker ROV-en øker med kvadratet av hastigheten. Hastigheten er derfor en av tilstandene i arbeidspunktet, og denne varierer kontinuerlig når ROV-en er under bevegelse. Arbeidspunktene må i slike tilfeller bestemmes ut fra hvilken tilstand man ønsker at systemet skal operere i. Alternativt kan det utledes en formel for å oppnå kontinuerlig varierende parametre. Det må i dette tilfellet utledes et uttrykk for hver regulatorparameter som funksjon av arbeidspunktet systemet befinner seg i.

Det finnes flere måter å implementere parameterstyring for en PID-regulator, og det enkleste er å definere grenser mellom ulike arbeidspunkter og la regulatoren få nye parametre hver gang prosessen beveger seg over disse grensene. Man må ved bruk av denne metoden på forhånd ha regnet ut parameterverdier for de ulike arbeidspunktene, og videre interpolere mellom disse når systemet beveger seg mellom arbeidspunktene. Et problem med denne løsningen er at systemets parameterverdier vil endres i sprang ved overgangen til et nytt arbeidspunkt, og man kan få uønskede responser på utgangen som følge av dette. Dersom systemet befinner seg rundt grensen mellom to arbeidspunkter, kan også parameterene hoppe frem og tilbake, og resultatet kan bli en «hakkete» regulering. Dette kan likevel løses ved å innføre hysterese, hvor det defineres en terskel systemet må bryte før overgangen til et nytt arbeidspunkt fører til parameterendring. Dette eliminerer derimot ikke problemet med at det oppstår et sprang i det endringen skjer. Den beste løsningen på problemet er derfor å finne parametre for et større sett av ulike arbeidspunkter, og enten bruke en oppslagstabell for disse eller utlede et utrykk for parameterverdiene som funksjon av arbeidspunktet. Dette eliminerer sprang i parameterverdiene da de endres gradvis.

Metoden med parameterstyring er mer kompleks enn de andre metodene presentert tidligere, men kan likevel fungere godt for mange typer systemer. Derfor skal det gjøres et forsøk på å bruke denne metoden under simulering, og vurdere om den gir en bedre oppførsel på systemet enn de andre metodene.

Manuell tuning

Selv om metodene for å bestemme regulatorparametre teoretisk gir et godt utgangspunkt for regulatoren, er det som regel behov for å tune disse manuelt basert på systemets oppførsel. Tabellen under viser hvilken effekt det å øke hver parameter individuelt har på et generelt system.

Tabell 8.1: Tabell over endringer i respons ved å øke en av parameterene [62].

Parameter	*Rise time*	Oversving	*Settling time*	*Steady-state error*
K_p	Reduserer	Øker	Liten endring	Reduserer
K_i	Reduserer	Øker	Øker	Fjerner
K_d	Minimal endring	Reduserer	Reduserer	Ingen effekt (teoretisk)

Tabellen er et nyttig verktøy man kan bruke for å tune bort uønskede elementer av en respons, og kunne prioritere hva som er de viktigste egenskapene for et system. Dette kan eksempelvis være å tune parametrene for å oppnå en lav *rise time*, men dette vil gå på bekostning av oversvinget i responsen til systemet.

8.2 Regulering av hiv

For å regulere hiv-prosessen skal det først utledes en overføringsfunksjon for prosessen for å kunne bruke Skogestads metode. Overføringsfunksjonen vil utledes basert på differensialligningen for hiv-prosessen som ble etablert i kapittel 7. Parametrene vil også bestemmes ved bruk av de andre metodene nevnt i forrige delkapittel, og til slutt vil alle settene med parametre vurderes opp mot hverandre.

8.2.1 Overføringsfunksjon for systemet

For å utlede overføringsfunksjonen, tas det utgangspunkt i ligning 7.7, og man setter inn variablene som er aktuelle for translasjon i hiv-retningen:

$$4 \cdot g \cdot u_{T,k}(\alpha) - \frac{1}{2} \cdot \rho \cdot v_z \cdot |v_z| \cdot C_{D,topp} \cdot A_{topp} - 1.2 = m \cdot a_z \tag{8.8}$$

For å finne overføringsfunksjonen for systemet, vil det være nyttig å sette opp en tilstandsrommodell. Dette er en modell som beskriver prosessen ut fra definerte innganger, utganger og tilstander. Siden modellen er basert på beregning av kraftbidrag, vil det være naturlig å definere posisjon og hastighet som tilstandsvariabler. Dette kommer av at akselerasjonen er proporsjonal med kraften, og direkte knyttet til hastighet og posisjon. Inngangen på systemet vil være

motorpådraget som beregnes av regulatoren. Selv om pådraget til thrusterene avhenger av karakteristikken, setter vi for enkelhets skyld variabelen u til å definere pådrag målt i kg. Ved hjelp av karakteristikken utledet i testrapporten i vedlegg B.1, kan pådraget i kg senere regnes om til pådrag i prosent til thrusterne. Utgangen på systemet vil være dybdeposisjonen til ROV-en, siden det er denne som skal reguleres. Merk at dybdeposisjonen både er en tilstand, og en utgang i modellen. Dette er ikke et problem, men kan bemerkes for å unngå forvirring. Videre brukes variablene z, $\dot{z}$ og $\ddot{z}$ for henholdsvis dybdeposisjon, -fart og -akselerasjon. Dette gjør man fordi variablene er direkte avhengig av hverandre. Differensialligningen kan da skrives slik:

$$4 \cdot g \cdot u - \frac{1}{2} \cdot \rho \cdot \dot{z} \cdot |\dot{z}| \cdot C_{D,topp} \cdot A_{topp} - 1.2 = m \cdot \ddot{z} \tag{8.9}$$

For å sette opp en tilstandsrom-modell må utrykket skrives på følgende form:

$$\begin{cases} \dot{x}(t) &= f(x(t), u(t), t) \\ y(t) &= g(x(t), u(t), t) \end{cases} \tag{8.10}$$

Her er x en notasjon på vektoren $[x_1, x_2]$, og utgjør tilstandsvektoren. Ettersom z i differensialligningen vår er dobbelderivert, er dette en verifisering av at det er behov for 2 tilstander i modellen. Tilstandene til hiv-prosessen er dybde og hastighet:

$$x_1 = z \quad x_2 = \dot{z}$$

Dette gir oss tilstandsrommodellen vist under. Utgangen y er som nevnt tidligere dybdeposisjonen, og pådraget u er thrusterpådrag målt i kg.

$$\begin{cases} f_1 &= \dot{x}_1 = \dot{z} = x_2 \\ f_2 &= \dot{x}_2 = \ddot{z} = \frac{4 \cdot g}{m} \cdot u - \frac{1.2}{m} - \frac{1}{2 \cdot m} \cdot \rho \cdot x_2 \cdot |x_2| \cdot C_{D,topp} \cdot A_{topp} \\ g &= y = x_1 \end{cases} \tag{8.11}$$

For å finne overføringfunksjonen til systemet må man skrive tilstandsrommodellen på kompakt form, og ettersom systemet er ulineært, må det også lineariseres før dette kan gjøres. For å linearisere må systemet beskrives som en endring rundt arbeidspunktet det lineariseres rundt.

$$\begin{aligned} x &= \tilde{x} + \delta x \\ u &= \tilde{u} + \delta u \\ y &= \tilde{y} + \delta y \end{aligned}$$

Her er $\tilde{x}, \tilde{u}$ og $\tilde{y}$ arbeidspunktet systemet lineariseres rundt, og $\delta z, \delta u$ og δy defineres som en «liten» endring rundt arbeidspunktet. Videre setter vi opp tilstandsrom-modellen på kompakt form basert på endringene rundt arbeidspunktet.

$$\begin{cases} \delta\dot{x}(t) &= A \cdot \delta x(t) + B \cdot \delta u(t) \\ \delta y(t) &= C \cdot \delta x(t) + D \cdot \delta u(t) \end{cases} \tag{8.12}$$

Hvor A, B, C og D er *Jakobimatrisene* til systemet. Disse er definert som:

$$A = \begin{bmatrix} \frac{\partial f_1(x,u,t)}{\partial x_1} & \frac{\partial f_1(x,u,t)}{\partial x_2} \\ \frac{\partial f_2(x,u,t)}{\partial x_1} & \frac{\partial f_2(x,u,t)}{\partial x_2} \end{bmatrix}_{\substack{x=\tilde{x} \\ u=\tilde{u}}} \qquad B = \begin{bmatrix} \frac{\partial f_1(x,u,t)}{\partial u} \\ \frac{\partial f_2(x,u,t)}{\partial u} \end{bmatrix}_{\substack{x=\tilde{x} \\ u=\tilde{u}}}$$

$$C = \begin{bmatrix} \frac{\partial g(x,u,t)}{\partial x_1} & \frac{\partial g(x,u,t)}{\partial x_2} \end{bmatrix}_{\substack{x=\tilde{x} \\ u=\tilde{u}}} \qquad D = \begin{bmatrix} \frac{\partial g(x,u,t)}{\partial u} \end{bmatrix}_{\substack{x=\tilde{x} \\ u=\tilde{u}}}$$

Ved å utføre partiellderivasjonene ender vi opp med følgende matriser:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & \frac{1}{m} \cdot \rho \cdot C_{D,topp} \cdot A_{topp} \cdot x_2 \end{bmatrix}_{x_2=\tilde{x}_2} \qquad B = \begin{bmatrix} 0 \\ \frac{4 \cdot g}{m} \end{bmatrix}$$

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

Utrykket for overføringsfunksjonen er basert på *Jakobimatrisene*:

$$H_{hiv}(s) = C \cdot (sI - A)^{-1} \cdot B + D \tag{8.13}$$

Den generelle overføringsfunksjonen for translasjon i hiv-retning blir dermed:

$$H_{hiv}(s) = \frac{\frac{4 \cdot g}{m}}{s^2 + \frac{\rho \cdot C_{D,topp} \cdot A_{topp} \cdot \tilde{x}_2}{m} \cdot s} \tag{8.14}$$

For å enkelt kunne lese parameterverdier fra Skogestads tabell, skriver vi overføringsfunksjonen om til standard form. I tillegg tar vi med dødtiden τ på 0.2 s.

$$H_{hiv}(s) = \frac{K}{(Ts+1)s} \cdot e^{-\tau} = \frac{\frac{4 \cdot g}{\rho \cdot C_{D,topp} \cdot A_{topp} \cdot \tilde{x}_2}}{\left(\frac{m}{\rho \cdot C_{D,topp} \cdot A_{topp} \cdot \tilde{x}_2} s + 1\right) s} \cdot e^{-0.2} \tag{8.15}$$

Det at systemet er ulineært, gjenspeiles også av at overføringsfunksjonen inneholder $\tilde{x}_2$. Dette betyr at overføringsfunksjonen for systemet avhenger av tilstanden, og må bestemmes ut fra et satt arbeidspunkt. Arbeidspunktet $[\tilde{x}_1, \tilde{x}_2]$ = [dybdeposisjon, dybdehastighet], må derfor tilegnes verdier. Ut fra overføringsfunksjonen ser man dog at det kun er dybdehastigheten som påvirker prosessen. Fra standardformen kan man se at hastigheten påvirker både prosessforsterkningen K, og prosessens tidskonstant T. Arbeidspunktet avhenger altså også av dybdeposisjonen, men denne har ingen betydning for prosessens overføringsfunksjon.

8.2.2 Parameterbestemmelse

Nå som overføringsfunksjonen for prosessen er utledet, kan parametrene bestemmes ved bruk av Skogestads metode. For hiv-prosessen gjøres det også et forsøk på å bruke parameterstyring.

Skogestads metode for ulike arbeidspunkter

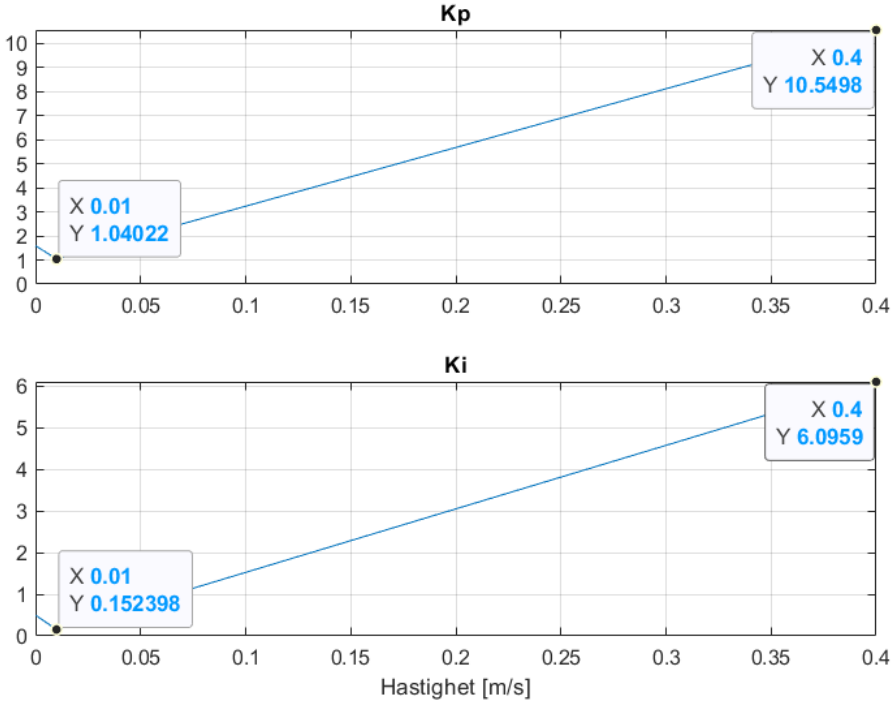
Fra den generelle overføringsfunksjonen kan man med Skogestads metode finne parameterverdier for ulike arbeidspunkt ved å sette inn disse i ligning 8.15. Overføringsfunksjonene for forskjellige arbeidspunkt og tilhørende parametere er vist i tabellen under. Arbeidspunktene er valgt basert på hiv-modellen som ble utledet i kapittel 7.

Arbeidspunkt [m/s]	H_{hiv}(s)	K_p	K_i	K_d
0	$\frac{1.962}{s^2} \cdot e^{-0.2s}$	1.59	0.5	1.27
0.02	$\frac{5.13}{(2.61s+1)s} \cdot e^{-0.2s}$	1.28	0.3	1.27
0.1	$\frac{1.03}{(0.52s+1)s} \cdot e^{-0.2s}$	3.23	1.52	1.27
0.4	$\frac{0.26}{(0.13s+1)s} \cdot e^{-0.2s}$	10.5	6.1	1.27

Disse parameterene vil bli testet gjennom simulering av regulatoren for å se hvilke som gir best regulering av systemet.

Parameterstyring av hiv-prosessen

Det er også gjort et forsøk på å parameterstyre regulatoren. For å gjøre dette settes det opp grafer for parameterene som funksjon av arbeidspunktet de er regnet ut fra. Regulatorparametrene for hvert arbeidspunkt bestemmes ved bruk av Skogestads tabell, og for å kunne utlede en formel plotter vi parameterverdiene mot tilhørende arbeidspunkt i MATLAB. Grafen for parameterene som funksjon av hastighet er vist i figur 8.4.



Figur 8.4: Figur som viser de forskjellige parameterene for forskjellige hastigheter.

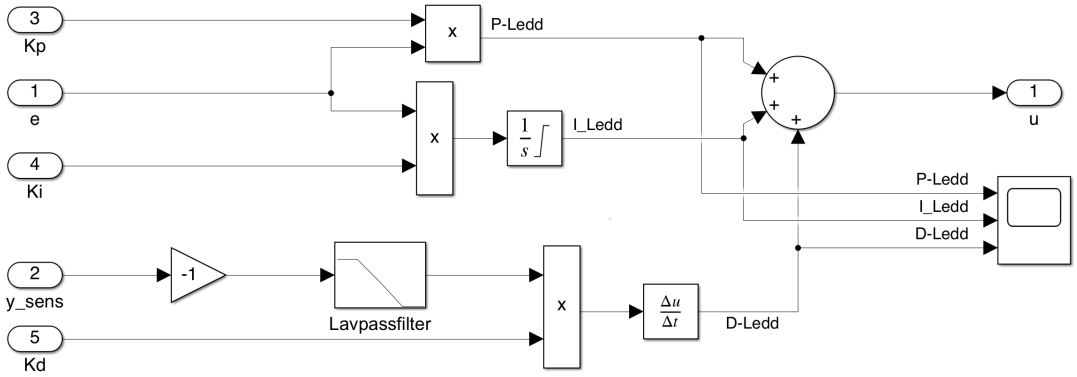
I figuren kan man se at K_p og K_i øker lineært foruten ved hastighet 0 m/s. For å utlede formelen for parameterene brukes det lineære området av grafen som utgangspunkt. K_d er den samme for alle punktene og er derfor ikke vist i figuren. Ved å bruke punktene markert i figur 8.4 finner vi et uttrykk for K_p og K_i som funksjon av hastighet:

$$K_p = 0.8 + 24 \cdot v \ , \quad K_i = 0.5 + 14.5 \cdot v \ , \quad K_d = 1.27 \tag{8.16}$$

Disse uttrykkene vil, sammen med parametersettene fra Skogestads metode, simuleres i Simulink og vurderes opp mot hverandre.

8.2.3 Simulering

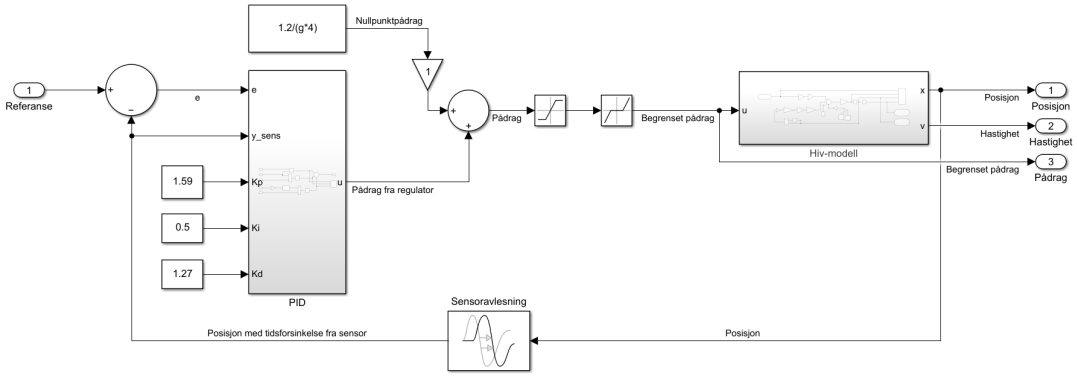
For å kunne simulere regulatoren sammen med modellene for de ulike frihetsgradene i Simulink, må PID-regulatoren som ble vist i figur 8.2 implementeres. Oppsettet for PID-regulatoren slik den er realisert i Simulink er vist i figur 8.5.



Figur 8.5: Modell av PID-en som skal brukes for alle regulerte frihetsgrader. Signalet *y_sens* tilsvarer i dette tilfellet den målte verdien på utgangen av prosessen.

PID-regulatoren er satt opp som et eget *subsystem* i Simulink, slik at den enkelt kan brukes sammen med alle prosessene som skal reguleres. Tidskonstanten til lavpassfilteret, T_f, velges basert på en metode utviklet av Finn Haugen [17]. Den settes derfor til å være lik $0.05T_d$, hvor T_d er derivattiden for hvert parametersett. Uttrykket for denne ble gitt i ligning 8.3. Filterkonstanten i simuleringen vil kun brukes for simuleringen, og ikke for regulatoren som vil bli implementert på mikrokontrolleren. Dette er fordi oppdateringsfrekvensen på tilsendte måleverdier i det fysiske systemet er en helt annen enn i simuleringen.

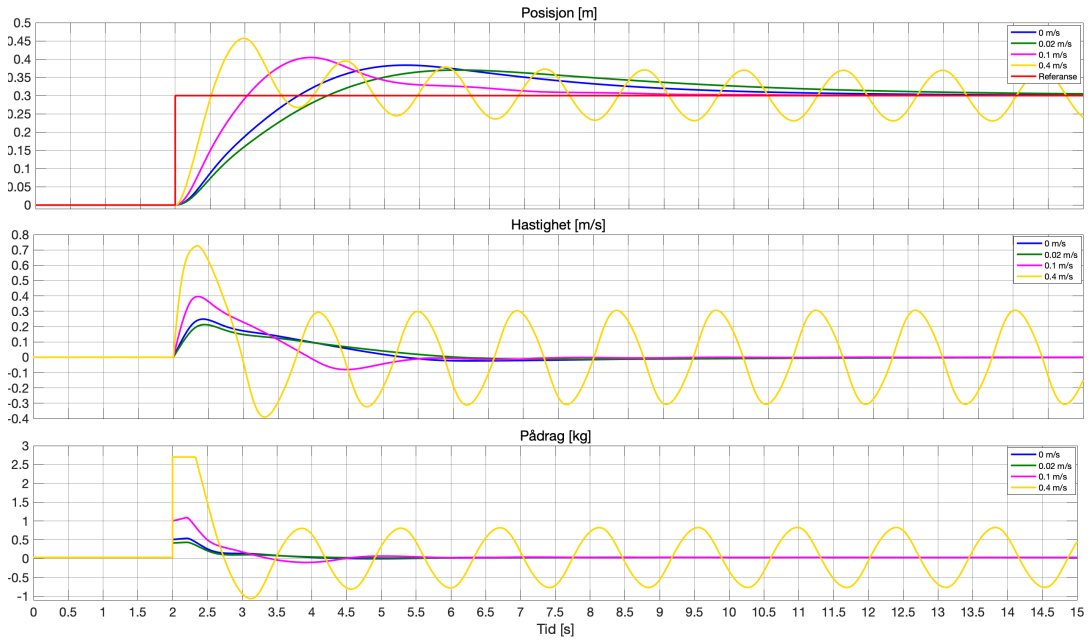
I figur 8.6 vises Simulinkopsettet for regulering av hiv-prosessen. PID-regulatoren, vist i figur 8.5, er lagt inn som et *subsystem* for å gjøre modellen mer oversiktlig. Det samme gjelder modellen for hiv-prosessen. Dødtiden på 0.2 s er lagt inn som en forsinkelse på signalet, og er i modellen kalt «sensoravlesning».



Figur 8.6: Systemet med regulering av hivmodellen. De viste parameterverdiene gjelder for arbeidspunktet 0 m/s.

Ettersom det er en konstant oppdrift på 1.2 N i hiv-prosessen, er det lagt inn et nullpunktpådrag for å motvirke dette. Pådraget u er gitt i kg og må fordeles på 4 thrustere, og derfor divideres 1.2 N på $g \cdot 4$. Dette er strengt tatt ikke nødvendig da integratorleddet vil kunne kompensere for oppdriften, men ettersom den er kjent skader det ikke å legge til et pådrag som motvirker denne. Det er for totalpådraget lagt til en begrensning, og en dødsone. Dette er implementert fordi thrusterene både har en maksimal pådragsverdi de kan bidra med, og en minimal pådragsverdi de ikke kan å gå under uten å stoppe fullstendig. Dette er vist i testrapporten i vedlegg B.1.

Reguleringen simuleres først med parametrene fra de fire forskjellige arbeidpunktene fra Skogestads metode for å sammenligne effektiviteten mellom de ulike parametersettene. Resultatet av denne simuleringen er vist i figur 8.7.



Figur 8.7: Modellen simulert for ulike arbeidspunkt.

Parameterene for arbeidspunktet 0.4 m/s gir vedvarede oscillasjoner. K_p og K_i er ved dette arbeidspunktet veldig store, og regulatoren blir dermed veldig hissig. Oscillasjonene kommer sannsynligvis på grunn av dødtiden på 0.2 s. Denne, sammen med den hissige reguleringen, gjør at hastigheten til ROV-en blir stor nok til at regulatoren henger etter prosessen, og ikke klarer å dempe oscillasjonene på utgangen.

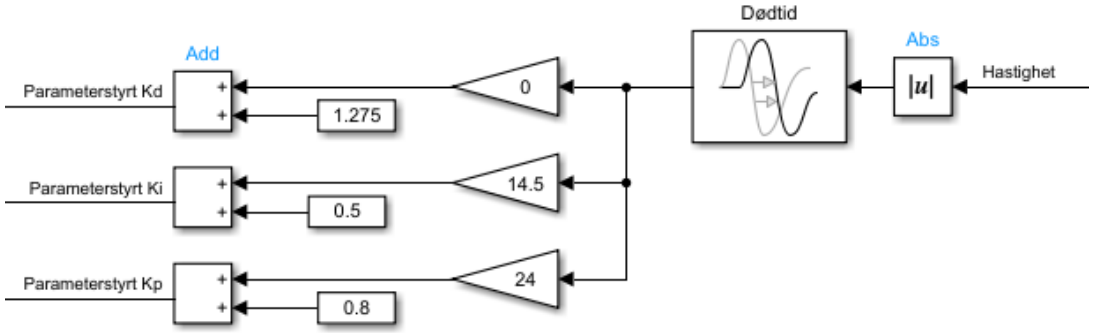
Vi mener at parameterene som gir den beste oppførselen er for arbeidpunktet 0.1 m/s. Selv om oversvinget til denne responsen ikke er det laveste, faller denne fortest mot referansepunktet. For å optimalisere responsen skal parameterene for dette arbeidspunktet tunes. Dette gjøres med utgangspunkt i tabell 8.1, som gir en oppskrift for manuell tuning. Vi har valgt å ikke bruke veldig mye tid på tuning av parameterene i den simulerte modellen, da disse sannsynligvis vil måtte tunes ytterligere på det fysiske systemet. Den tuningen som ble gjennomført, ble gjort med mål om å dempe oversving og responstid så godt det lot seg gjøre. De originale parameterene for arbeidpunktet 0.1 m/s, samt de tunede parameterene for dette arbeidspunktet er vist i tabellen under.

Modell	K_p	K_i	K_d
Ikke tunet	3.23	1.52	1.27
Tunet	3.8	1.3	1.5

I regulatoren med tunede parametre er det i tillegg lagt inn en begrensing på ± 0.3 for integratorleddet. Dette hindrer integratorleddet fra å akkumulere mye avvik, og bli veldig stort ved store sprang. Begrensningen ble lagt til basert på testing av simuleringsmodellen.

I tillegg til å sammenligne den tunede og den ikke-tunede modellen, tas også modellen med

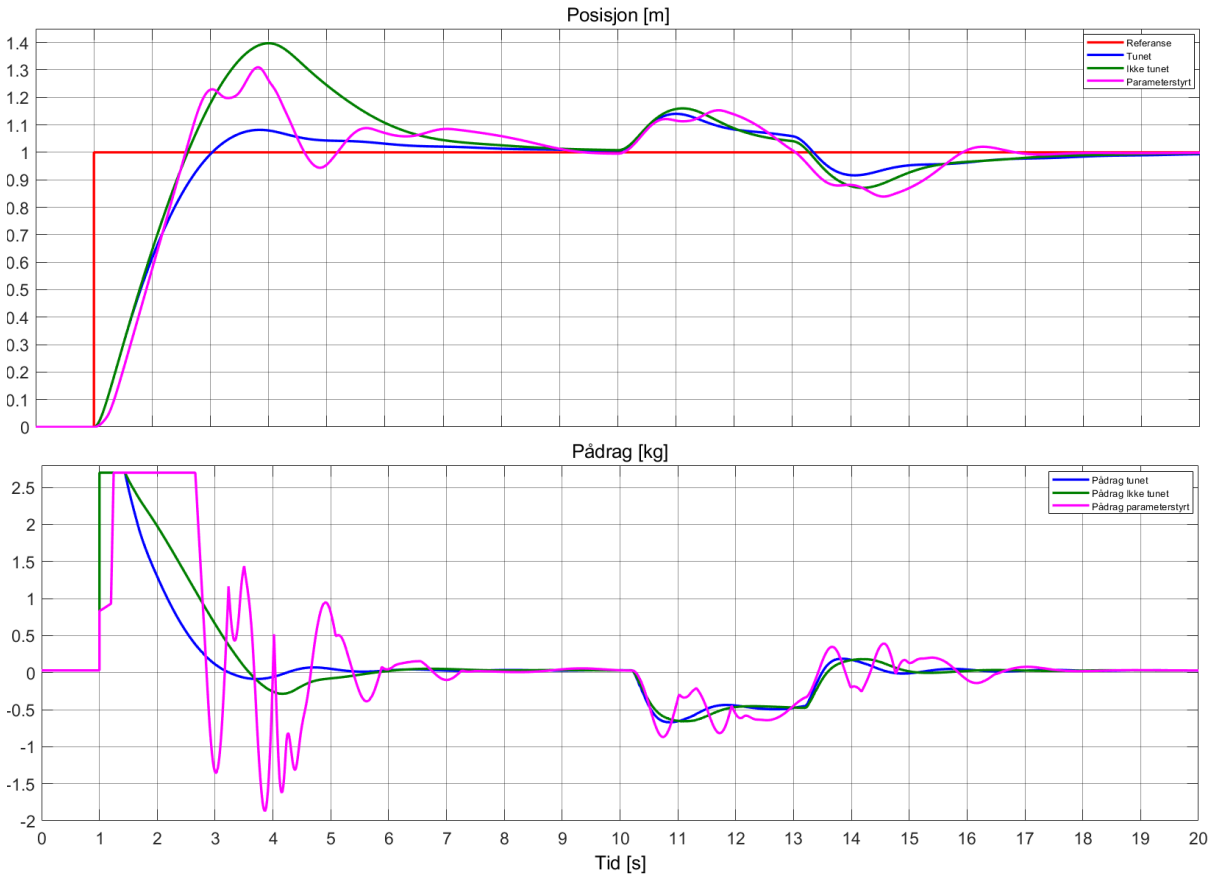
parameterstyring med. For å implementere parameterstyringen endres simulinkmodellen som vist i figur 8.8.



Figur 8.8: Implementering av parameterstyring i simulinkmodellen.

Her brukes den målte hastigheten til å kontinuerlig endre parameterene. For å unngå forskyvning og skjevheter i prosessen, er dødtiden τ også lagt til før endringen av parametrene. Dette sørger for at parametrene endres samtidig som det skal beregnes et nytt pådrag.

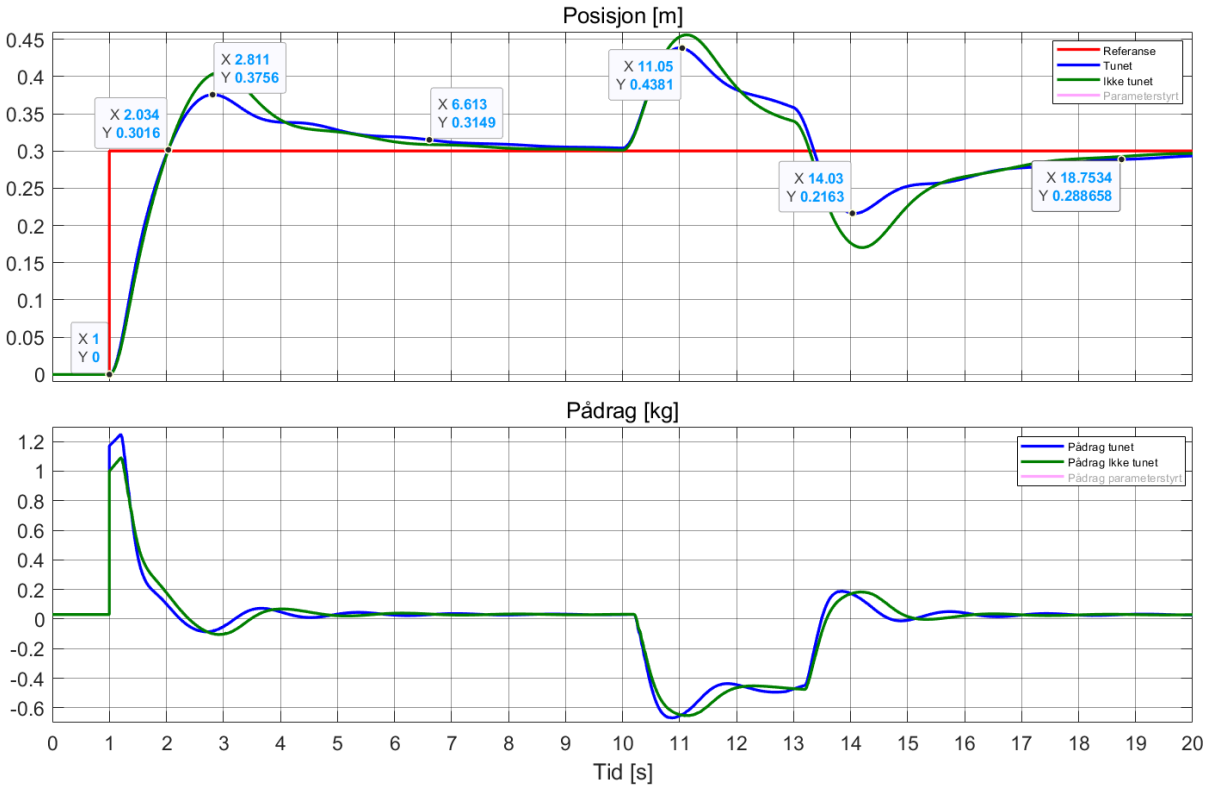
I figur 8.9 vises en simulering av et sprang på 1 m med de tunede parameterene, ikke-tunede parameterne og med parameterstyring. Det er også lagt inn en forstyrrelseskraft fra 10 til 13 sekunder i simuleringen. Denne forstyrrelsen er på 10 N, og er lagt til for å se hvordan regulatorene håndterer tillagt tyngde på ROV-en.



Figur 8.9: Simulering av et sprang på 1 meter. De to første responsene er fra PID-en basert på arbeidspunktet 0.1 m/s, men den ene er tunet i etterkant. Den siste responsen er basert på parameterstyring.

Selv om parameterstyringen gir mindre oversving enn den ikke-tunede regulatoren, kan man se at pådragsbruken er ekstremt aggressiv. Dette er ikke ønskelig da det kan forårsake unødvendig slitasje på thrusterne, og kraftig varierende strømtrekk. Årsaken til den aggressive pådragsbruken kan være at parametrene blir veldig store når hastigheten øker. Ettersom hastigheten er stor i det man nærmer seg settpunktet, gir regulatoren store utslag.

I figur 8.10 vises et sprang på 0.3 m med de tunede parameterene og de ikke-tunede parameterene. Den samme forstyrrelsen som i forrige tilfelle er også lagt til her.



Figur 8.10: Simulering av et sprang på 30 cm. Parameterstyrt respons er fjernet.

Den tunede responsen gir det beste resultatet i både spranget på 1 m og 30 cm. Effekten av begrensningen på integratorleddet er størst i spranget på 1 m, og gir et mye lavere oversving enn den andre regulatoren. Ved et sprang på 30 cm får vi et oversving på ca. 7.56 cm. Dette er et oversving på ca 25 %. Tiden det tar før settpunktet nås første gang er 1 s. Det tar ca. 5.6 s før dybden er innenfor 5% av spranget. Det er vanskelig å simulere en realistisk forstyrrelse, men forsøket som er gjort her viser at den tunede regulatoren gir et mindre utslag på dybden ved forstyrrelsen på 10 N enn den ikke-tunede. Ved påføringen av forstyrrelsen oppstår det et utslag på ca. 13.8 cm. Når forstyrrelsen fjernes oppstår det et utslag i motsatt retning på ca. 8.4 cm. Dybden faller innenfor 5% av utslaget etter ca. 4.7 s. Dette velger vi å se på som en tilfredstillende respons, og de tunede parametrene vil bli testet på det fysiske systemet når ROV-en skal vannsettes.

8.3 Regulering av rull og stamp

I dette delkapittelet presenteres utledningen av overføringsfunksjonen for stamp- og rull-prosessen, samt prosessen for bestemmelse av passende regulatorparametere.

8.3.1 Overføringsfunksjon for systemet

Ut fra ligning 7.20 kan vi sette opp et generelt uttrykk for stamp- og rullrotasjon. I tilfellet for rull og stamp defineres vinkelen og vinkelhastigheten som tilstandsvariabler. Inngangen blir igjen satt som pådraget u gitt i kg. Utgangen, og den verdien som skal reguleres, settes til å

være vinkelposisjonen. Videre brukes ϕ, $\dot{\phi}$ og $\ddot{\phi}$ som variabler for henholdsvis vinkelposisjon, -hastighet og -akselerasjon. Man ender opp med følgende differensialligning:

$$I \cdot \ddot{\phi} = \underbrace{4 \cdot g \cdot r_T \cdot cos(90^o - \theta_T)}_{B_{thruster}} \cdot u - \underbrace{F_B \cdot r_{COM-B}}_{B_{oppretting}} \cdot sin(\phi) \tag{8.17}$$

$$- \underbrace{\frac{1}{2} \cdot \rho \cdot (C_{d,1} \cdot B_1 + C_{d,2} \cdot B_2)}_{B_{vannmotstand}} \cdot \dot{\phi} \cdot |\dot{\phi}| \tag{8.18}$$

For å gjøre utledningen av en overføringfunksjon mer oversiktlig, defineres noen hjelpevariabler for beskrivelse av konstantene som ikke utgjør tilstands- pådrags eller utgangsvariabler. Konstantene $B_{thruster}$, $B_{vannmotstand}$ og I er konstanter som avhenger av om rotasjonen skjer i stamp- eller rull-retning. $B_{oppretting}$ er den samme for både rull og stamp. I ligningen under er de nye konstantene satt inn:

$$I \cdot \ddot{\phi} = B_{thruster} \cdot u - B_{oppretting} \cdot sin(\phi) - B_{vannmotstand} \cdot \dot{\phi} \cdot |\dot{\phi}| \tag{8.19}$$

Ettersom ϕ er dobbelderivert, kan det bekreftes at det er det behov for 2 tilstander. I dette tilfellet tilsvarer disse vinkelposisjon og -hastighet:

$$x_1 = \phi \quad x_2 = \dot{\phi}$$

Videre gjennomføres en tilsvarende linearisering som i delkapittel 8.2.1. Dette gir oss en generell overføringsfunksjon på standardform for arbeidspunktet [x_1, x_2] = [vinkelposisjon, vinkelhastighet]. Merk at overføringsfunksjonen i dette tilfellet er av en annen type, og derfor er standardformen annerledes enn for hiv-prosessen. Overføringsfunksjonen er vist under:

$$H_{rull/stamp}(s) = \frac{\frac{B_{thruster}}{I \cdot B_{oppretting} \cdot cos(\tilde{x}_1)}}{\frac{1}{B_{oppretting} \cdot cos(\tilde{x}_1)} s^2 + \frac{2 \cdot B_{vannmotstand} \cdot |\tilde{x}_2|}{I \cdot B_{oppretting} \cdot cos(\tilde{x}_1)} s + 1} \cdot e^{-0.2} \tag{8.20}$$

Det kan observeres at i tilfellet for rull og stamp, vil begge komponentene av arbeidspunktet ha innvirkning på prosessens overføringsfunksjon. Derfor må begge disse tilegnes verdier når det senere skal utledes overføringsfunksjoner basert på ulike arbeidspunkter av prosessens tilstand.

Videre settes verdiene for $B_{thruster}$, $B_{vannmotstand}$, $B_{oppretting}$ og I inn, og man ender opp med følgende overføringsfunksjoner for stamp- og rull-prosessene.

$$H_{rull}(s) = \frac{\frac{2.90}{cos(\tilde{x}_1)}}{\frac{1}{7.65 \cdot cos(\tilde{x}_1)} s^2 + \frac{0.315 \cdot |\tilde{x}_2|}{cos(\tilde{x}_1} s + 1)} \cdot e^{-0.2} \tag{8.21}$$

$$H_{stamp}(s) = \frac{\frac{2.36}{cos(\tilde{x}_1)}}{\frac{1}{4.79 \cdot cos(\tilde{x}_1)} s^2 + \frac{0.482 \cdot |\tilde{x}_2|}{cos(\tilde{x}_1} s + 1)} \cdot e^{-0.2} \tag{8.22}$$

Overføringsfunksjonene legger grunnlaget for videre parameterbestemmelse for de to regulatorene.

8.3.2 Parameterbestemmelse

Overføringsfunksjonene er av en annen type enn for hiv-prosessen. Dette skaper problemer ved bruk av Skogestads metode. Det kan likevel bestemmes regulatorparametre ut fra overføringsfunksjonene, men gjennom en alternativ metode til Skogestad. Dette vil bli forklart under.

Parametere bestemt ut fra overføringsfunksjonen

Den generelle overføringsfunksjonen for rull og stamp har en form hvor nevneren gir kompleks-konjugerte røtter. Parameterebesetemmelse for denne typen overføringsfunksjoner er ikke definert av Skogestad. Som alternativ kan man bruke en annen metode for parameterbestemmelse, kalt IMC [1], som er definert for denne typen overføringsfunksjoner [38]. IMC er egentlig opphavet til Skogestads metode, men Skogestad har laget sin egen variant av denne hvor formålet var å forbedre regulatorenes kompenseringsevne mot forstyrrelser. En av endringene Skogestad gjorde, var blant annet å sette begrensinger for størrelsen til integratorleddet. Ved store forstyrrelser kan integratorleddet akkumulere store bidrag, og sørge for en hissig oppførsel fra regulatoren.

Parameterene for rull og stamp blir med IMC-metoden bestemt ut fra en overføringsfunksjon på følgende standardform:

$$\frac{K \cdot e^{-\theta s}}{\tau^2 s^2 + 2\zeta\tau s + 1} \tag{8.23}$$

Videre regnes parameterene slik som i tabell 12.1 på side 207 i boken *Process Dynamics and Control* [38]:

$$K_p = \frac{2\zeta\tau}{K(T_c + \theta)} \ , \quad T_i = 2\zeta\tau \ , \quad T_d = \frac{\tau}{2\zeta}$$

Hvor:

- θ er dødtiden i systemet, tidligere definert som 0.2 s.
- T_C er systemets responstid. Velger $T_C = \theta$ som anbefalt av Skogestad.

Ved hjelp av denne metoden kan vi finne parameterverdier for forskjellige arbeidspunkt ved å sette disse inn i overføringsfunksjonene. T_i og T_d må konverteres til K_i og K_d som beskrevet i delkapittel 8.1.1. Overføringsfunkjsonene for ulike arbeidspunkt i rull-prosessen, og tilhørende parametere, er vist i tabellen under. Dødtiden på 0.2 s er lagt inn.

[1] **I**nternal **M**odel **C**ontrol

Arbeidspunkt posisjon [°]	Arbeidspunkt hastighet [°/s]	$H_{rull}(s)$	K_p	K_i	K_d
0	15	$\frac{2.9 \cdot e^{-0.2s}}{0.36^2 \cdot s^2 + 2 \cdot 0.23 \cdot 0.36 \cdot s + 1}$	0.14	0.86	0.11
0	35	$\frac{2.9 \cdot e^{-0.2s}}{0.36^2 \cdot s^2 + 2 \cdot 0.53 \cdot 0.36 \cdot s + 1}$	0.33	0.86	0.11
15	15	$\frac{3.0 \cdot e^{-0.2s}}{0.37^2 \cdot s^2 + 2 \cdot 0.23 \cdot 0.37 \cdot s + 1}$	0.14	0.83	0.11
15	35	$\frac{3.0 \cdot e^{-0.2s}}{0.36^2 \cdot s^2 + 2 \cdot 0.54 \cdot 0.36 \cdot s + 1}$	0.33	0.83	0.11

Tabell 8.2: Parametere for ulike arbeidspunkt i rull-prosessen.

Overføringsfunkjsonene for de samme arbeidspunktene i stamp-prosessen, og tilhørende parametere, er vist i tabellen under. Dødtiden på 0.2 s er lagt inn.

Arbeidspunkt posisjon [°]	Arbeidspunkt hastighet [°/s]	$H_{stamp}(s)$	K_p	K_i	K_d
0	15	$\frac{2.36 \cdot e^{-0.2s}}{0.46^2 \cdot s^2 + 2 \cdot 0.28 \cdot 0.46 \cdot s + 1}$	0.27	1.06	0.22
0	35	$\frac{2.36 \cdot e^{-0.2s}}{0.46^2 \cdot s^2 + 2 \cdot 0.65 \cdot 0.46 \cdot s + 1}$	0.62	1.06	0.22
15	15	$\frac{2.44 \cdot e^{-0.2s}}{0.46^2 \cdot s^2 + 2 \cdot 0.28 \cdot 0.46 \cdot s + 1}$	0.27	1.02	0.22
15	35	$\frac{2.44 \cdot e^{-0.2s}}{0.46^2 \cdot s^2 + 2 \cdot 0.65 \cdot 0.46 \cdot s + 1}$	0.62	1.02	0.22

Tabell 8.3: Parametere for ulike arbeidspunkt i stamp-prosessen.

Disse parameterene vil bli testet under simulering for å se hvilke som gir best regulering av systemet.

Forenklet bruk av Skogestad

For å benytte Skogestads metode, må det gjøres en stor forenkling av systemet. Ettersom målet med reguleringen er å holde ROV-en stabil i vannet, innebærer dette å operere rundt en vinkelhastighet på 0°/s og en vinkelposisjon på 0°. Forenklingen gjøres ved å sette disse verdiene direkte inn i differensialligningen i likning 8.19. Man ender da opp med et lineært utrykk:

$$I \cdot \ddot{\phi} = B_{thruster} \cdot u \tag{8.24}$$

Ved å bruke samme metode for å utlede overføringsfunksjoner basert på tilstandsrommodell ender vi i dette tilfellet opp med enda en ny type overføringsfunksjon:

$$H_{rull}(s) = \frac{22.2}{s^2}\ , \quad H_{stamp}(s) = \frac{11.31}{s^2}$$

Slike overføringsfunksjoner kalles dobbelintegratorer. For en dobbeltintegrerende overføringfunksjonen kan Skogestads metode benyttes, og de resulterende parameterene bestemmes ut fra Skogestads tabell, som ble vist i figur 8.3. Parametrene er listet opp i tabellen under.

Modell	K_p	K_i	K_d
$H_{rull}(s)$	0.14	0.044	0.11
$H_{stamp}(s)$	0.28	0.086	0.22

På grunn av forenklingen gjort med modellen kan disse parametrene være noe usikre, men de tas like vel med for videre vurdering under simuleringen.

Ziegler-Nichols

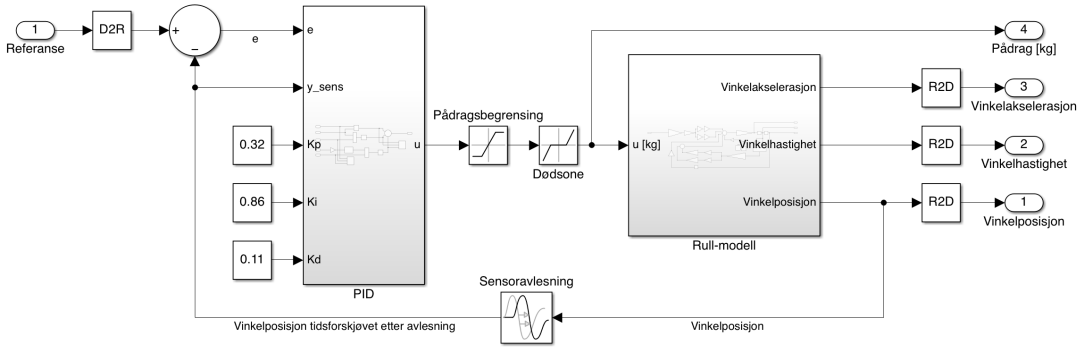
For rull og stamp gjøres det også et forsøk på parameterbestemmelse med metoden til Ziegler og Nichols som beskrevet i kapittel 8.1.1. Parametrene bestemt fra denne metoden er vist i tabellen under.

Prosess	K_{pu}	T_u	K_p	K_i	K_d
Rull	0.66	1.79	0.40	0.44	0.089
Stamp	1.79	2.27	0.47	0.42	0.13

Parameterverdiene er i dette tilfellet funnet ved å følge prosedyren til Ziegler-Nichols i Simulink-modellen for rull og stamp. Parametrene vil sammenlignes med de andre settene som er bestemt.

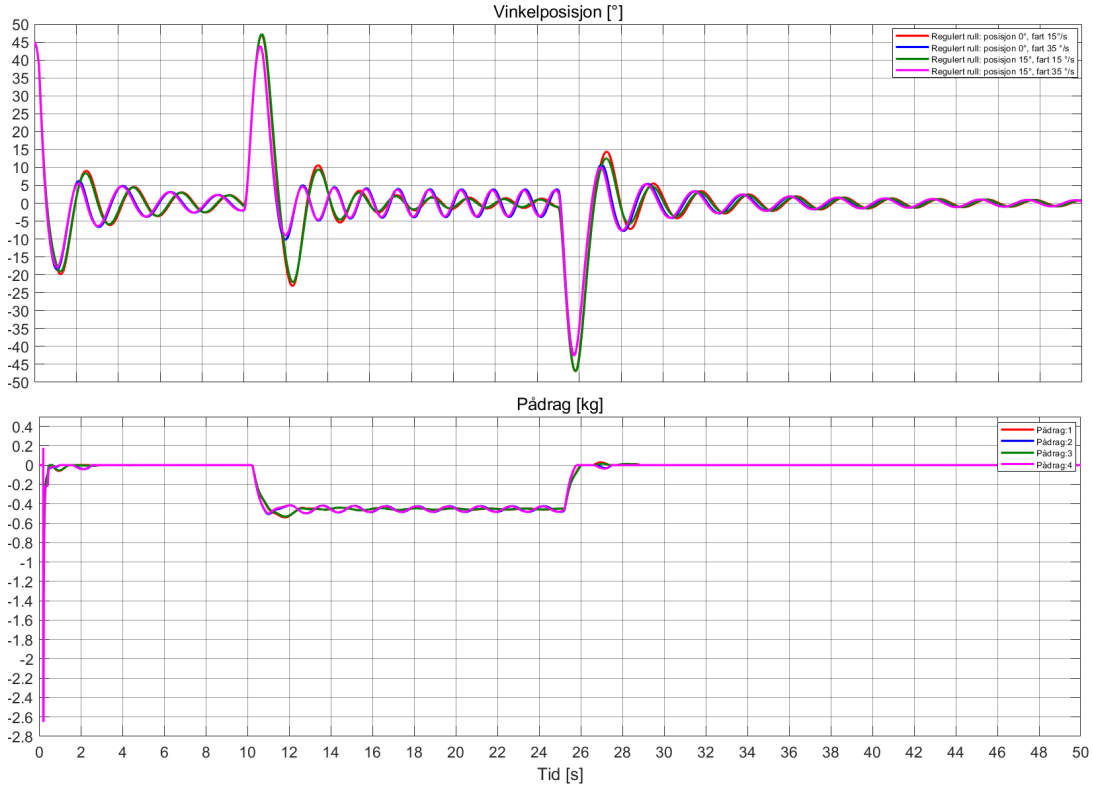
8.3.3 Simulering av rullregulering

Modellen for regulering av rull-prosessen, vist i figur 8.11, tilsvarer modellen for regulering av hiv, men prosessmodellen er erstattet med rull-prosessen utledet i kapittel 7. Det er i tillegg lagt inn blokker med navnet R2D for å konvertere fra radianer til grader slik at verdien som vises i skopet er i grader. For at referansen skal kunne settes i grader er det lagt inn en blokk, D2R, som konverterer grader til radianer.



Figur 8.11: Systemet med regulering av rullmodellen.

Reguleringen simuleres med parameterene fra de fire arbeidspunktene funnet for rull-prosessen i delkapittel 8.3.2. Initiell vinkelposisjon settes til 45°, og det blir lagt til en forstyrrelse i momentsummen mellom 10 og 25 sekunder. Denne forstyrrelsen er på 5 Nm, som tilsvarer 10 N på en 0.5 m lang arm fra massesenter. Dette er den maksimale vekten arrangøren har angitt at vi skal kunne løfte med ROV-en. Resultatet fra simuleringen er vist i skopet i figur 8.12.



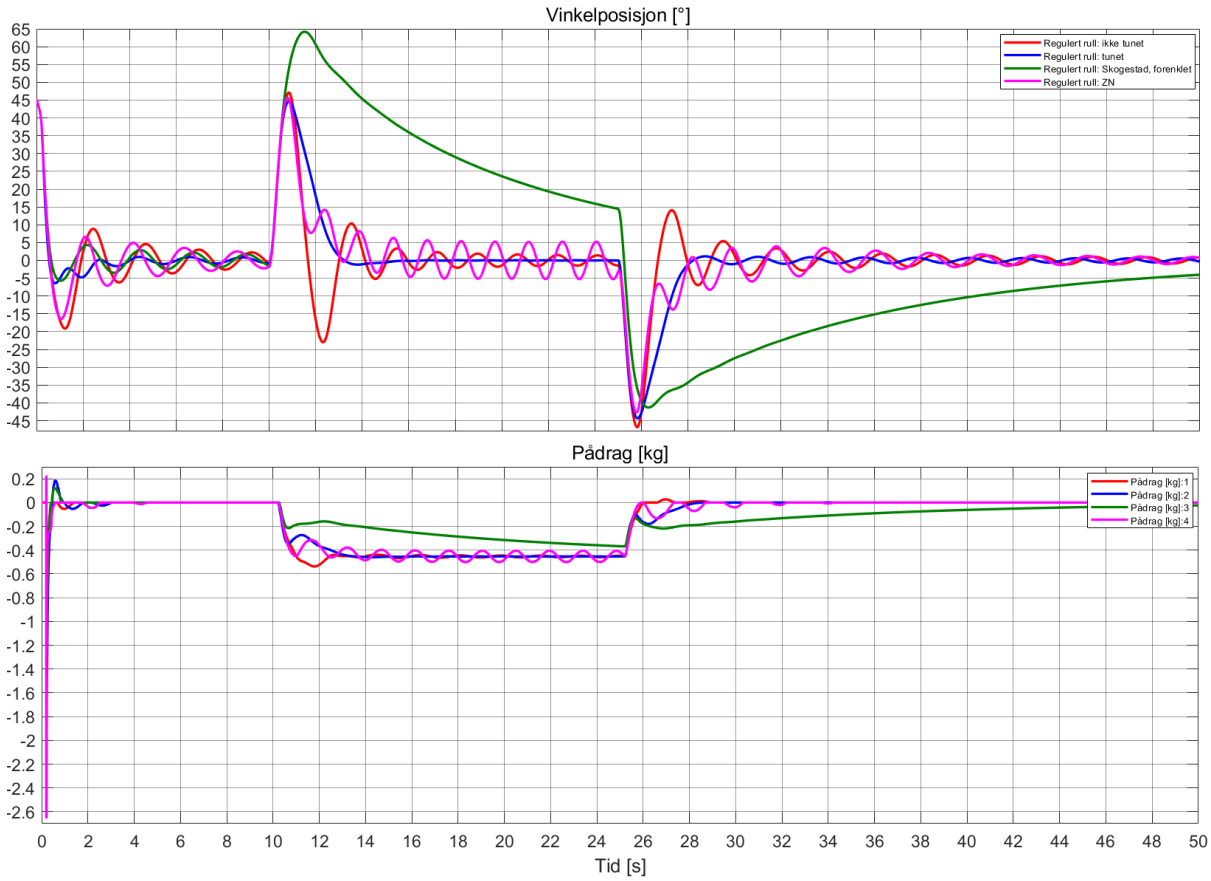
Figur 8.12: Modellen simulert for ulike arbeidspunkt.

Resultatet fra simuleringen viser at parameterene endres minimalt ved å endre vinkelposisjonen i arbeidspunktet. Ved endret vinkelhastighet ser man større utslag. Siden ROV-en helst skal operere rundt 0° i vinkelposisjon, velges denne som første verdi av arbeidspunktet. Videre velges hastigheten basert på hvilke responser som ser ut til å være raskest. Vi velger derfor å tune parameterene fra arbeidspunktet 0° og 15°. Dette arbeidspunktet har en respons med større oversving, men til gjengjeld faller responses raskere til ro. Dette gjelder både for den initielle vinkelposisjonen og for forstyrrelsen.

Parametrene fra dette arbeidspunktet tunes på samme måte som parametrene for hiv-prosessen. De tunede parameterene sammelignes deretter med de originale for det valgte arbeidspunktet, og med parametrene fra Skogestads og Ziegler-Nichols metode. Alle parametersettene er vist i tabellen under.

Modell	K_p	K_i	K_d
Ikke tunet	0.14	0.86	0.11
Tunet	0.14	0.35	0.15
Skogestad	0.14	0.044	0.11
Ziegler-Nichols	0.62	0.76	0.13

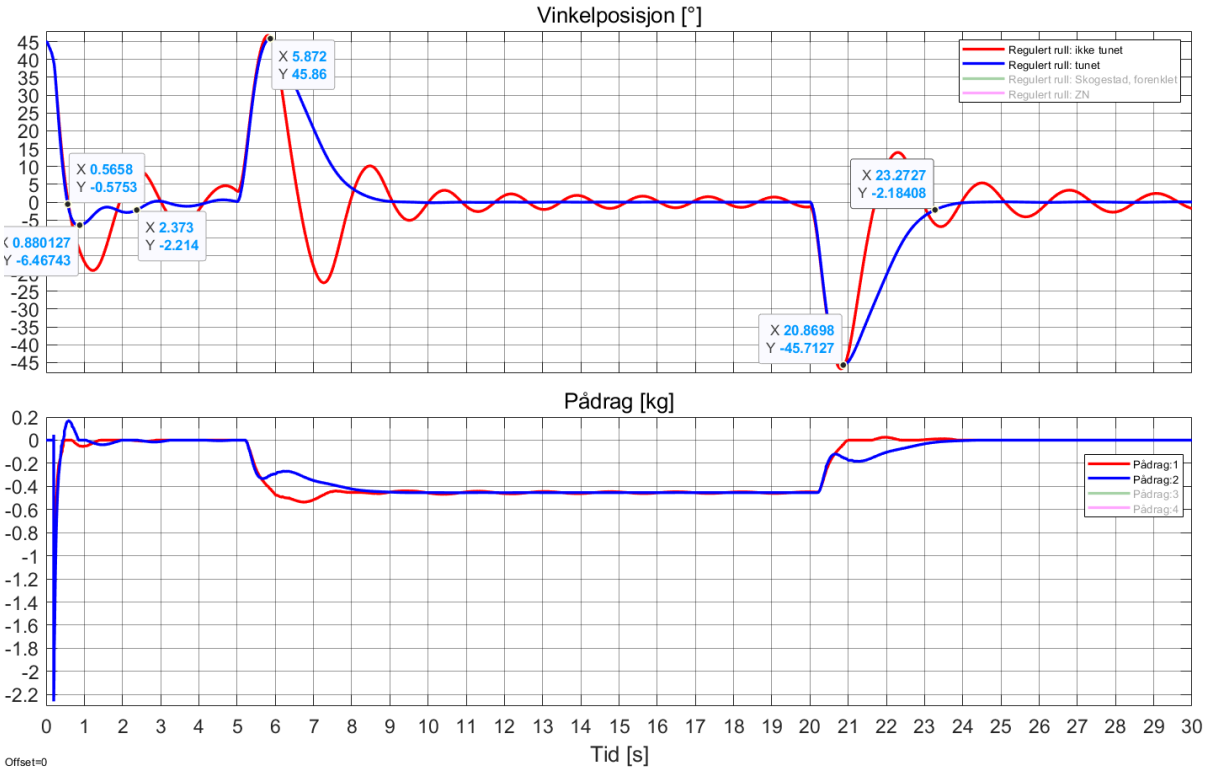
Resultatet fra simuleringen av de fire settene med parametre er vist i figur 8.13.



Figur 8.13: De to første responsene er fra PID-en basert på arbeidspunktet 0° og 15°/s, men den ene er tunet i etterkant. De siste responsene er basert på Skogestad og Ziegler-Nichols. Igjen er det lagt inn momentforstyrrelse mellom 10 og 25 sekunder.

Simuleringen viser at de tunede parameterene gir en mye mer stabil respons. Selv om oversvinget er tilnærmet likt, er oscillasjonene nesten eliminert fra responsen. Resultatet fra Skogestads metode viser en god respons på oppretting av initielt vinkelutslag, men dårlig håndtering av forstyrresle. Dette kommer antakeligvis av forenklingen som er gjort. Metoden til Ziegler-Nichols virker å gi en aggressiv respons, og vil kreve en del tuning for å få oscillasjonene fjernet.

Videre har vi simulert på nytt, og kun evaluert regulatorene med tunede og ikke tunede parametere, funnet ved bruk av IMC-metoden. Forstyrrelsen er lik i størrelse, men er flyttet til å virke mellom 5 og 20 sekunder ut i simuleringen. Resultatet er vist i figur 8.14



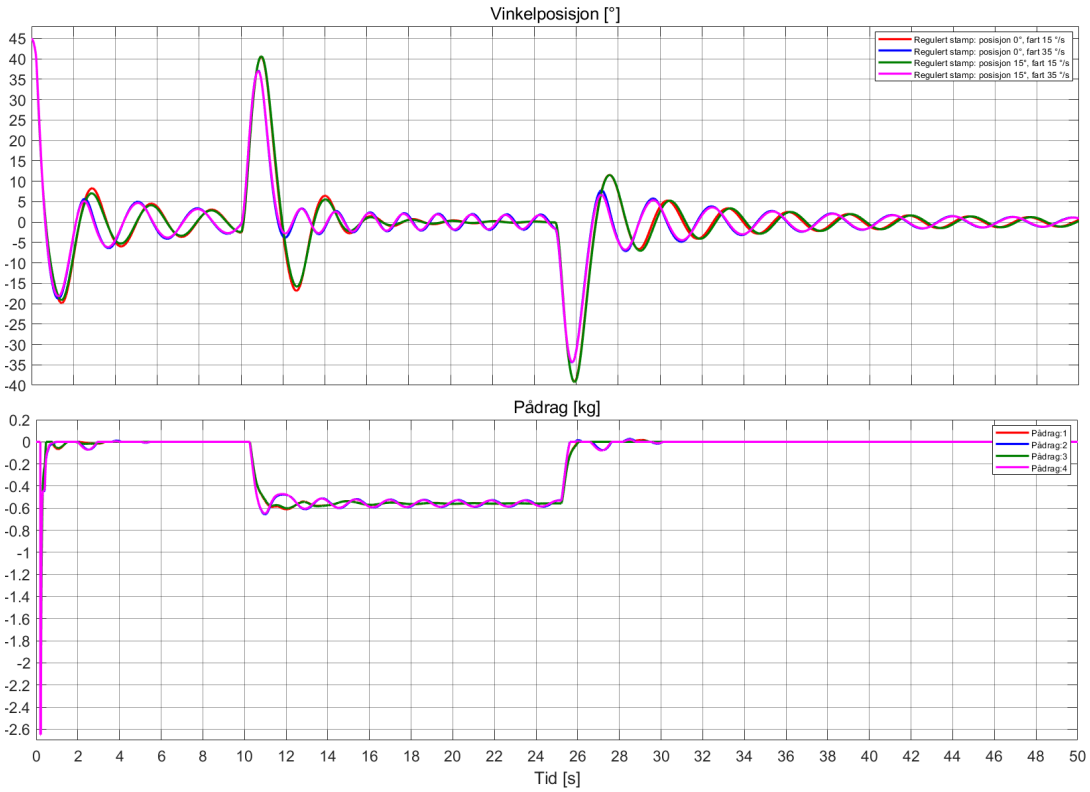
Figur 8.14: Forstyrrelsen er nå lagt til mellom 5 og 20 sekunder. Skogestad og Ziegler-Nichols er fjernet.

Den tunede responsen gir det beste resultatet. Ved et «sprang» fra den initielle vinkelposisjonen på 45°, og til referansen på 0°, viser responsen et undersving på ca. 6.5°. Dette tilsvarer ca. 14 % av det initielle vinkelutslaget. Tiden det tar før settpunktet nås første gang er ca. 0.6 s. Det tar ca. 2.4 s før vinkelposisjonen er innenfor 5% av spranget ned til referansen. Forstyrrelsen gir et kraftig utslag på vinkelen på ca. 45.7°, men denne faller innenfor 5% av utslaget på ca. 2.4 s. Dette regner vi som en tilfredstillende respons, og de tunede parameterene vil senere bli testet på det fysiske systemet.

8.3.4 Simulering av stampregulering

Simulink-modellen for regulering av stamp-prosessen er lik som for rull. Altså tilsvarende figur 8.11. Selve prosess-modellen er erstattet med prosessen for bevegelse i stamp-rotasjon.

Reguleringen simuleres først med parametrene fra de fire arbeidspunktene funnet for stamp-prosessen i delkapittel 8.3.2. Initielt vinkelutslag er igjen 45°, og det er også i dette tilfellet lagt til en forstyrrelse på 5 Nm i momentsummen mellom 10 og 25 sekunder. Resultatet fra simuleringen er vist i figur 8.15.



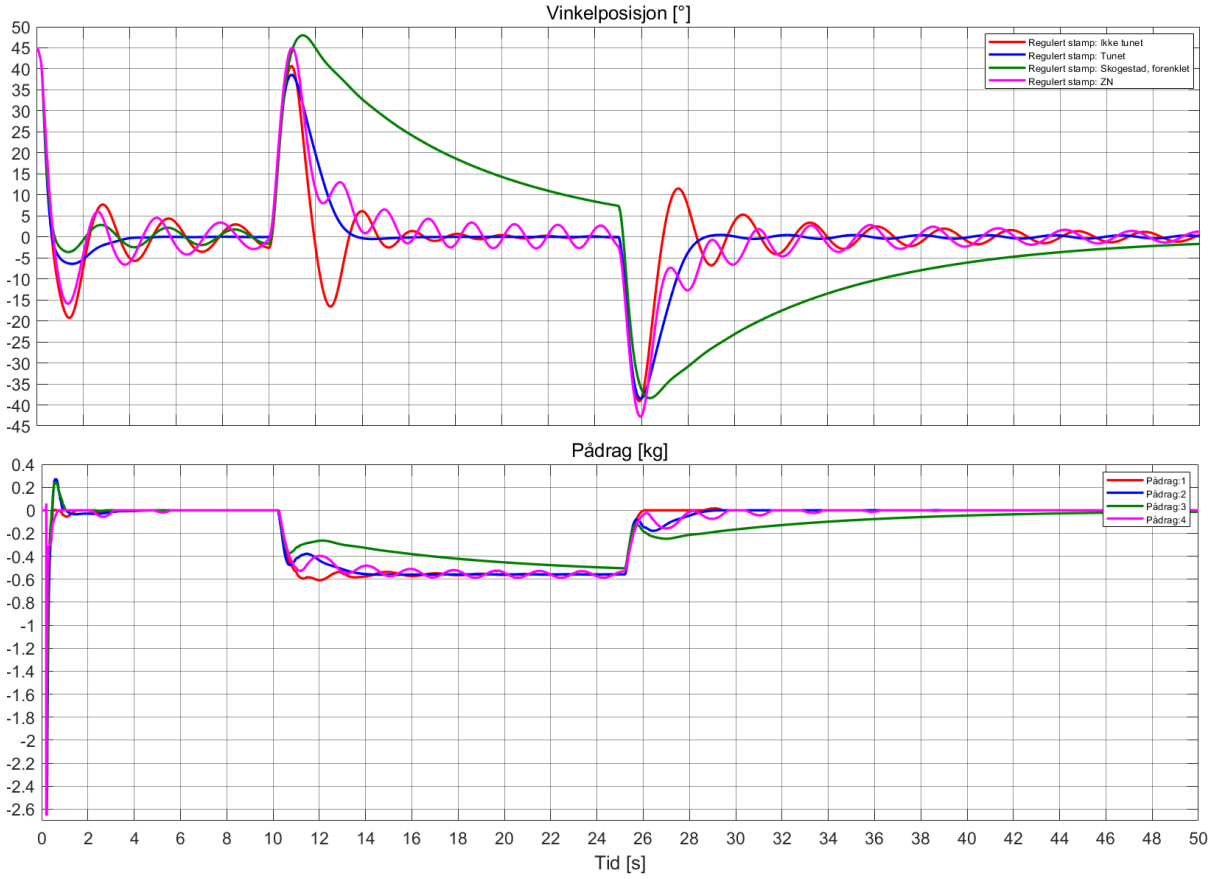
Figur 8.15: Modellen simulert for ulike arbeidspunkt.

Resultatet fra simuleringen er veldig lik resultatet for rull-prosessen. Vi velger derfor av samme grunner som for rull-prosessen å tune parametrene fra arbeidspunktet 0° og 15°/s.

Parametrene blir også tunet manuelt på samme måte som tidligere. De tunede parametrene sammelignes deretter med de originale for dette arbeidspunktet, og med parametrene fra Skogestad og Ziegler-Nichols. De fire parametersettene er vist i tabellen under.

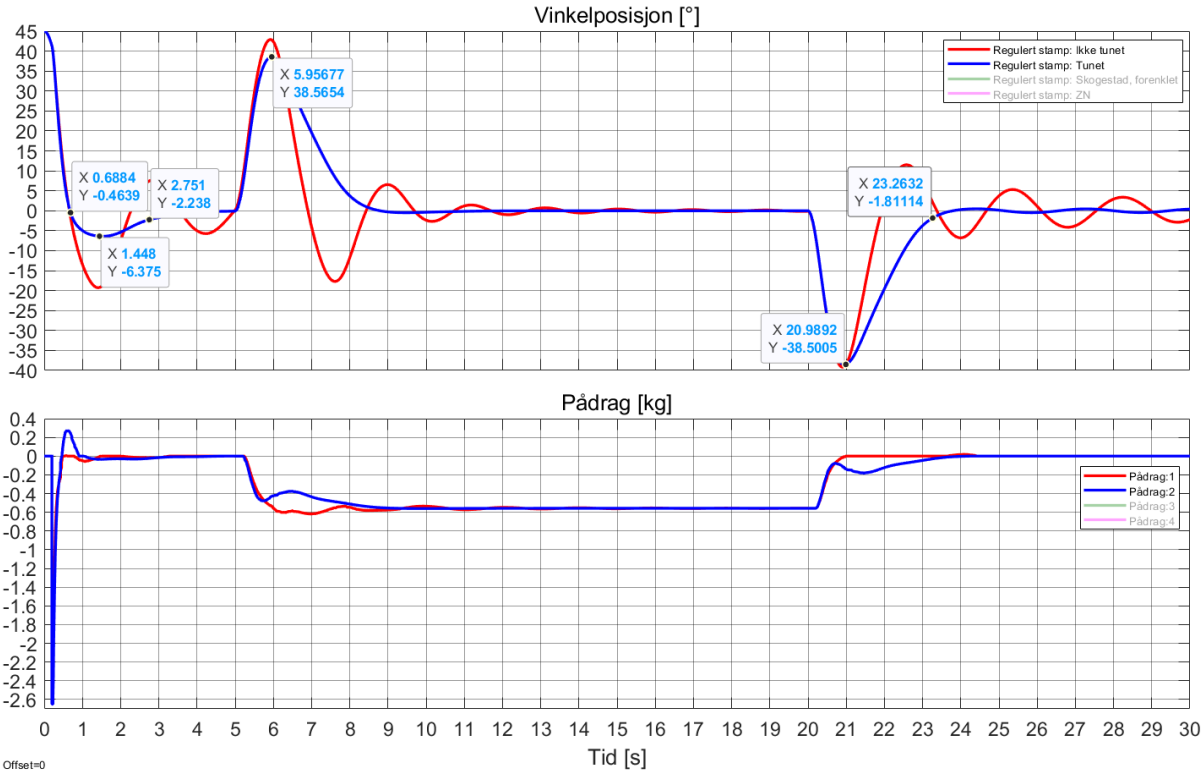
Modell	K_p	K_i	K_d
Ikke tunet	0.27	1.06	0.22
Tunet	0.27	0.5	0.27
Skogestad	0.28	0.086	0.22
Ziegler-Nichols	0.65	0.63	0.17

Resultatet fra simuleringen av de fire settene med parametre er vist i figur 8.16.



Figur 8.16: De to første responsene er fra PID-en basert på arbeidspunktet 0° og 15°/s, men den ene er tunet i etterkant. De siste responsene er basert på Skogestad og Ziegler-Nichols. Igjen er det lagt inn momentforstyrrelse mellom 10 og 25 sekunder.

Igjen gir parameterene fra Skogestads og Ziegler- Nichols metode dårligere respons enn det IMC-metoden gjør. Videre har vi derfor simulert på nytt og kun evaluert regulatorene med tunede og ikke-tunede parametere, funnet ved bruk av IMC-metoden. Forstyrrelsen er også her flyttet til å virke mellom 5 og 20 sekunder. Resultatet er vist i figur 8.17



Figur 8.17: Forstyrrelsen er nå lagt til mellom 5 og 20 sekunder. Skogestad og Ziegler-Nichols er fjernet.

Argumentasjonen for å velge den tunede responsen er lik for stamp-prosessen som for rull-prosessen. Den tunede responsen gir det beste resultatet. Ved et «sprang» fra den initielle vinkelposisjonen på 45°, til referansen på 0°, viser responsen et undersving på ca. 6.4°. Dette tilsvarer ca 14 % av «spranget». Tiden det tar før settpunktet nås første gang, er ca. 0.46 s. Det tar ca 2.8 s før dybden er innenfor 5% av spranget ned til referansen. Forstyrrelsen gir et kraftig utslag på vinkelen på ca 38.5°, men denne faller innenfor 5% av utslaget på ca 2.3 s. Dette regnes som en tilfredstillende respons, og de tunede parametrene vil senere bli testet på det fysiske systemet.

8.4 Implementering i programvare

For å realisere reguleringen i en mikrokontroller, er det nødvendig at alle utregninger kan gjøres diskret. Dette er fordi målte sensordata sendes stegvis, og ikke som en kontinuerlig oppdatering. Bidraget fra P-leddet i integratoren trenger ikke å gjøres noe med for å kunne håndtere diskrete måleverdier. For bidragene fra I- og D-leddet må det utføres henholdsvis humerisk integrasjon, og numerisk derivasjon. I dette delkapittelet vises diskretiseringen av reguleringssystemet, samt implementeringen i programvare.

8.4.1 Numerisk integrasjon

Den numeriske integrasjonen gjøres ved bruk av trapesmetoden. Denne går ut på å summere forrige avvik og det nye avviket, dele på 2 og multiplisere med tidsskrittet. Pådraget fra integralleddet blir dermed:

$$u_i = K_i \cdot \frac{e(k) + e(k-1)}{2} \cdot T_s \tag{8.25}$$

8.4.2 Numerisk derivering og filtrering

Numerisk derivering gjøres ved å finne differansen mellom en ny og en gammel måling, for så å dele denne på tidskrittet mellom målingene. Måleverdien må også filtreres for å unngå uønskede effekter av støy. Pådraget fra derivatleddet blir dermed:

$$u_d = K_d \cdot \frac{y_{m,f}(k) - y_{m,f}(k-1)}{T_s} \tag{8.26}$$

Hvor:

- u_d er pådraget fra derivatleddet.
- K_d er parameteren for D-leddet i regulatoren.
- $y_{m,f}(k)$ er den filtrerte måleverdien.
- $y_{m,f}(k-1)$ er den forrige filtrerte måleverdien.
- T_s er tidsskrittet mellom målingene.

For å sette opp et diskret førsteordens lavpassfilter med tidskonstant T_f, kan det implementeres et IIR-filter på formen:

$$y_{m,f}(k) = \alpha \cdot y_{m,f}(k-1) + (1-\alpha) \cdot y_m(k) \tag{8.27}$$

Hvor

- $\alpha = \frac{T_f}{T_f + T_s}$
- $y_m(k)$ er måleverdien av prosessens utgangsverdi.

En tommelfingerregel er at $T_f >= T_s \cdot 5$. Ettersom målingene gjøres med et en frekvens på 20 Hz blir den laveste anbefalte verdien på T_f lik 0.25 s. Dette gir en minsteverdi av α på 0.83. Merk at tidkonstanten T_f ikke er den samme som blir brukt i lavpassfilteret før D-leddet i Simulink-modellen. På grunn av at målingene gjøres med en frekvens på 20 Hz, og med mye mindre tidsskritt i Simulinkmodellen, blir det feil å bruke samme tidskonstant.

8.4.3 Implementering av regulatoren i programvare

Implementering av regulatoren på mikrokontrolleren gjøres ved å følge en algoritme, eller en oppskrift. Denne reguleringsalgoritmen er vist i følgende pseudokode:

1. Først blir avviket regnet ut:
$$e(k) = y_{ref} - y_m(k). \tag{8.28}$$
Hvor y_{ref} er referanseverdien/settpunktet, og $y_m(k)$ er måleverdien.

2. Det filtrerte målesignalet for derivasjonsleddet blir regnet med filterkoeffisienten α.
$$Y_{m,f} = \alpha \cdot Y_{m,f}(k-1) + (1-\alpha) \cdot Y_m(k); \tag{8.29}$$

3. Det proporsjonale pådraget blir regnet ut:
$$u_p = K_p \cdot e(k) \tag{8.30}$$

4. Det integrerte pådraget blir regnet ut som følger:
$$u_i(k) = u_i(k-1) + K_i \cdot \frac{(e(k) + e(k-1))}{2} \cdot T_s \tag{8.31}$$

5. Det integrerte pådraget har sin egen pådragsbegrensning slik at integratorleddet ikke blir for stort.

6. Pådraget fra derivasjonsleddet blir regnet ut som:
$$u_d = K_d \cdot \frac{(Y_{m,f}(k) - Y_{m,f}(k-1))}{T_s} \tag{8.32}$$

7. Det totale pådraget u er summen av u_p, u_i og u_d:
$$u = u_p + u_i + u_d \tag{8.33}$$

8. Pådraget har en pådragsbegrensning slik at om pådraget er over maksgrensen, eller under minimumsgrensen, blir pådraget satt lik grenseverdien.

9. Pådraget i kg konverteres til slutt om til pådrag i prosent basert på thrusterkarakteristikken utledet i testrapporten i vedlegg B.1. Ved stamp- og rull-rotasjon vil pådraget for to av thrusterene være det motsatte av de to andre slik at alle thrusterene bidrar i samme rotasjonsretning.

Kodeutsnittet under viser hvordan regulatoren for stamp-rotasjon er implementert:

Kode 8.1: Kodeutsnitt av regulatorfunksjonen for regulering av stampvinkel

```
1 void stamp_regulator(void){
2     spid.e = spid.yr - stamp_m; // Avvik
3     spid.fbf = param.stamp_pid.a * spid.fbfs + param.stamp_pid.b*stamp_m;   ...
          // filtrert maaledata (tilbakekobling)
4
5     spid.up = param.stamp_pid.kp*spid.e;    // Proporsjonalbidrag
6     spid.ui = spid.uis + param.stamp_pid.ki * param.stamp_pid.ts * ...
          (spid.e+spid.es)/2;  // integratorbidrag
7     spid.ud = -param.stamp_pid.kd * (spid.fbf-spid.fbfs) / ...
          param.stamp_pid.ts;      // derivatorbidrag
8
9     if (spid.ui > param.stamp_pid.ui_maks){  // Integratorbegrensning
```

```
10          spid.ui = param.stamp_pid.ui_maks;   }
11      else if (spid.ui < param.stamp_pid.ui_min){
12          spid.ui = param.stamp_pid.ui_min;    }
13
14      spid.ut = spid.up+spid.ui+spid.ud;  // Totalbidrag
15      if (spid.ut > param.stamp_pid.ut_maks){     // Totalbidragsbegrensning
16          spid.ut = param.stamp_pid.ut_maks;}
17      else if (spid.ut < param.stamp_pid.ut_min){
18          spid.ut = param.stamp_pid.ut_min;    }
19
20      spid.es = spid.e;   // Oppdaterer forrige avvik
21      spid.fbfs = spid.fbf;   // Oppdaterer forrige filtrerte maaling
22      spid.uis = spid.ui;     // Oppdaterer forrige integratorbidrag
23
24      float prosent_paadrag = kg_til_paadrag(spid.ut);        // Konverterer ...
            kg thrust til prosent paadrag
25      stampbidrag.vhf = prosent_paadrag;
26      stampbidrag.vvf = prosent_paadrag;
27      prosent_paadrag = kg_til_paadrag(-spid.ut);     // Konverterer for ...
            invers rettet thrustere
28      stampbidrag.vhb = prosent_paadrag;
29      stampbidrag.vvb = prosent_paadrag;
30  }
```

I tabellen under vises en oversikt over de forskjellige variablene brukt i koden:

Parameter	Betydning
spid.e	Avvik
spid.yr	Referanse/settpunkt
stamp_m	Målverdi/stampvinkel
spid.fbf	Filtrert måleverdi
spid.fbfs	Forrige filtrerte måleverdi
param.stamp_pid.a	Filterkoeffisient
param.stamp_pid.b	Filterkoeffisient (1-param.stamp_pid.a)
spid.up	Proporsjonalbidrag
param.stamp_pid.kp	K_p
spid.ui	Integratorbidrag
spid.uis	Forrige integratorbidrag
param.stamp_pid.ki	K_i
param.stamp_pid.ts	T_s
spid.es	Forrige avvik
spid.ud	Derivatorbidrag
param.stamp_pid.kd	K_d
param.stamp_pid.ui_maks	Øvre integratorbegrensing
param.stamp_pid.ui_min	Nedre integratorbegrensning
spid.ut	Totalbidrag
param.stamp_pid.ut_maks	Øvre pådragsbegrensing
param.stamp_pid.ut_min	Nedre pådragsbegrensning
prosent_paadrag	Pådrag i prosent
stampbidrag.vhf	Pådrag til Vertikal Høyre Fremme
stampbidrag.vvf	Pådrag til Vertikal Venstre Fremme
stampbidrag.vhb	Pådrag til Vertikal Høyre Bak
stampbidrag.vvf	Pådrag til Vertikal Venstre Bak

I kodeutsnittet under vises funksjonen for å konvertere pådraget i kg til pådrag i prosent. Funksjonen som tar høyde for motorkarakteristikken, og hvilken retning thrusterne skal gi pådrag, eksekveres senere i styringsprosessen. Derfor beregner *kg_til_paadrag()* uskalert pådrag uansett thrusterretning. I selve utregningen brukes derfor absoluttverdien til pådraget i kg. Funksjonen gir likevel pådraget i prosent med samme fortegn som pådraget i kg.

Kode 8.2: Kodeutsnitt av funksjonen for konvertering fra kg til prosent pådrag

```
1 float kg_til_paadrag(float kg_thrust){
2     float kg_float = (float) (fabs(kg_thrust));
3     float paadrag = (24.44*kg_float*kg_float+39.94*kg_float + ...
          0.0)/(kg_float+0.21); //
4     if(kg_thrust<0.0){
5         return -paadrag;    }
6     else {
7         return paadrag;     }
8 }
```

Prosessen som behandler pådragene videre ble beskrevet i programvarekapittelet 5.3.3.

Nå som det er bestemt regulatoparametre teoretisk, og regulatorene er implementert på mikrokontrolleren, er systemet klart til å kunne testes fysisk. Det neste delkapittelet tar for seg den avsluttende testen gjort på ROV-en.

8.5 Avsluttende regulatortest

I dette delkapittelet presenteres fysiske tester av reguleringen i dybde, rullvinkel og stampvinkel, med ROV-en i vann. På grunn av tidsbegrensning er ikke testen like omfattende som vi ønsket. Det ville vært ønskelig å kjøre sprangresponser for de regulerte frihetsgradene for å sammenligne den faktiske oppførselen til ROV-en med den teoretiske oppførselen simulert i Simulink. På grunn av forsinkelser i ferdigstillingen av andre systemer i elektronikkhuset, kunne ikke regulatorene testes før i siste uken av bachelorprosjektet.

8.5.1 Hensikt

Formålet med testen er å få en indikasjon på hvorvidt regulatorene fungerer ut fra det teoretiske grunnlaget som er lagt ned i kapittel 8, og eventuelt gi grunnlag for videre justering av regulatorparametrene.

8.5.2 Metode

Regulatorene er implementert for å sørge for autonom stabilitetsregulering av ROV-en. Testene vil ikke være basert på en spesifikk fremgangsmåte med etablert teoretisk grunnlag. En slik test ville typisk vært å utføre sprangresponser, og kvantisere ROV-ens egenskaper ut fra responsen. Testene som skal gjøres vil heller baseres på visuell observasjon av ROV-ens stabiliseringsegenskaper, og tolkning av logget sensordata i etterkant. Det er derfor ikke et mål å oppnå optimale

parametere, men heller å vise at regulatorene kan justeres grovt, og fortsatt forbedre ROV-ens stabilitetsregulering.

8.5.3 Fremgangsmåte

Først skal de teoretisk utregnede regulatorparameterene testes på ROV-en. Disse parameterene skal vurderes ved å visuelt observere stabiliteten ved stillstand i vannet, og at reguleringen ikke gir uønskede effekter. Ut fra den visuelle vurderingen, skal parameterene justeres slik at ROV-en oppnår en tilfredsstillende stabilitet.

For å verifisere at regulatorene bidrar til bedret oppførsel og stabilitet, skal det utføres tre ulike tester. De individuelle testene utføres først med regulatorene deaktivert, og deretter med alle tre regulatorer aktive samtidig. De tre testene som gjennomføres er listet opp under.

1. ROV-en skal kjøres i en rett linje på tvers av bassenget. Piloten skal sørge for at ROV-en kjører kun i jagretning. Dette innebærer å gi pådrag rett frem, og å kompensere med girbevegelse dersom navlestrengen drar ROV-en til siden. Piloten skal ikke hjelpe ROV-en til å opprettholde rett dybde.
2. Det skal festes en vekt som utgjør ca. 10 N i vann på ROV-ens manipulator. Piloten skal ikke bidra til ROV-ens stabilitet.
3. En person skal dytte kraftig ned på ett av hjørnene til ROV-en for å simulere en forstyrrelse.

8.5.4 Resultat

Innledningsvis plasserte vi ROV-en i bassenget med regulatorer oppsatt med teoretisk utregnede parametre. Det ble fort oppdaget at parameterene funnet fra simulering ga mye oscillasjoner, og veldig aggressiv pådragsbruk. Dette indikerer at den forventede responstiden, T_C, som er satt til 0.2 s, er et urimelig krav til systemet. Vi valgte derfor å beregne nye parametre ut fra en forlenget responstid. I tillegg endret vi integratorforsterkningen c fra 4 til 1.5 for hiv-regulatoren for å dempe aggressiviteten. Utregningene ble foruten disse endringene gjort på samme måte som i kapittel 8. Endringen gav følgende parametre med tilhørende responstid:

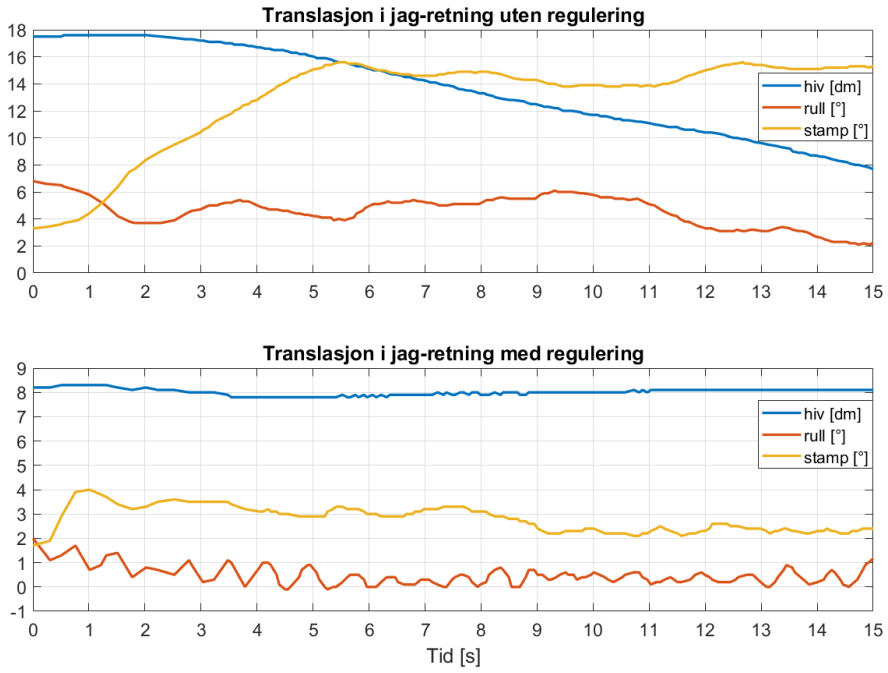
Tabell 8.4: Tabell med nye parametre.

Regulator	T_C [s]	K_p	K_i	K_d
Hiv	1.0	1.049	0.452	0.425
Rull	1.2	0.041	0.246	0.032
Stamp	1.2	0.076	0.303	0.063

Disse parametrene dempet pådragsbruken, men det oppstod fortsatt oscillasjoner i rull- og stamp-prosessen. Fra parametrene i tabell 8.4, kan man se at K_i for disse er flere ganger større enn K_p og K_d. Ettersom ROV-en har et naturlig oppretningsbidrag, skal det ikke være behov for et dominerende I-ledd. På grunn av oscillasjonene valgte vi å justere K_i ytterligere. Ved obesrvasjon under justeringen ble det vurdert at en mest stabile oppførselen ble oppnådd ved å sette $K_i = 0$. Etter en siste visuell observasjon, ble stabiliteten vurdert til å være tilfredsstillende

nok til å gjennomføre de tre testene beskrevet i fremgangsmåten. Resultatet av justeringen ble altså at rull-og stamp-prosessen ble regulert med en PD-regulator.

Første test ble gjennomført ved å gi pådrag, og kjøre ROV-en i omtrent 15 sekunder kun i jag-retning. Sensordataene ble i etterkant hentet ut fra loggen i operatørgrensesnittet, og plottet i MATLAB. Figur 8.18 viser resultatet etter gjennomføringene både med og uten regulator. Positiv hiv-måling tilsvarer dybden ROV-en befinner seg på. Høyere måleverdier tilsvarer alstå at ROV-en befinner seg dypere i bassenget. Positiv rullmåling tilsvarer rullvinkel til høyre, og positiv stampvinkel tilsvarer tipping av fronten oppover, eller steiling.

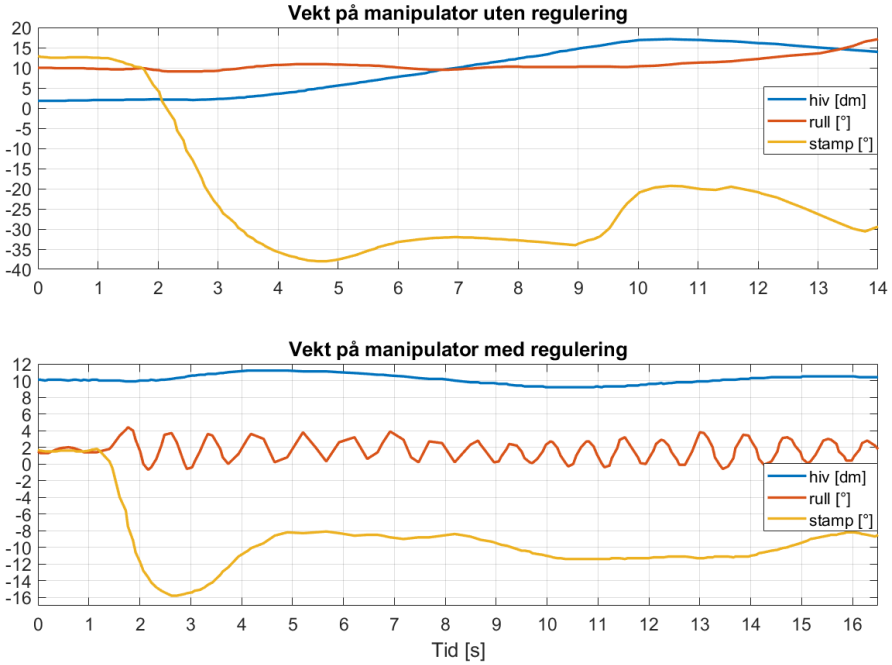


Figur 8.18: Grafer over rullvinkel, stamvinkel og dybde ved translasjon i jag-retning. Øverste plott viser testen utført uten regulering, og nederste plott viser testen med regulering. Dybden er vist frem i desimeter for å ligge i samme skala som gradmålingene fra rull og stamp. Merk også forskjellen i y-aksens skalering i de to plottene.

Man kan fra det øverste plottet se at ROV-en beveger seg oppover mot vannskorpen når den ikke reguleres. Stampvinkel ligger rundt 15°, og dybden måles til en endring på rundt 80 cm mellom 4 og 15 sekunder ut i testen. Dette kommer hovedsaklig av at navlestrengen drar ROV-en nedover i bakenden og resultatet er at jag-bevegelse fører til kjøring oppover i bassenget. Rullvinkelen varierer mellom 2° og 8°, men denne blir påvirket av gir-kompenseringen fra piloten, og kommer ikke utelukkende av forstyrrelser under jag-translasjon.

Det nederste plottet viser at regulatoren holder ROV-en på en noenlunde konstant dybde på 80 cm gjennom hele testen. Rullvinkelen holdes nokså stabil på mellom 0° og 2°, hvor det største utslaget er i det piloten starter å kjøre. Stampvinkelen ligger mellom 2° og 4° under kjøring. Dette kommer fortsatt av at navlestrengen drar på bakenden. Grunnen til at dybden ikke endres selv om ROV-en steiler under kjøring, er at hiv-reglatoren kompenserer med pådrag fra de vertikale thrusterne. Forskjellen mellom kjøring med og uten regulator er også enda større enn plottene indikerer, da y-aksen i det første plottet er skalert etter de store variasjonene.

I andre test ble et lodd festet til ROV-ens manipulator. Figur 8.19 viser resultatet etter gjennomføringen både med og uten regulator.

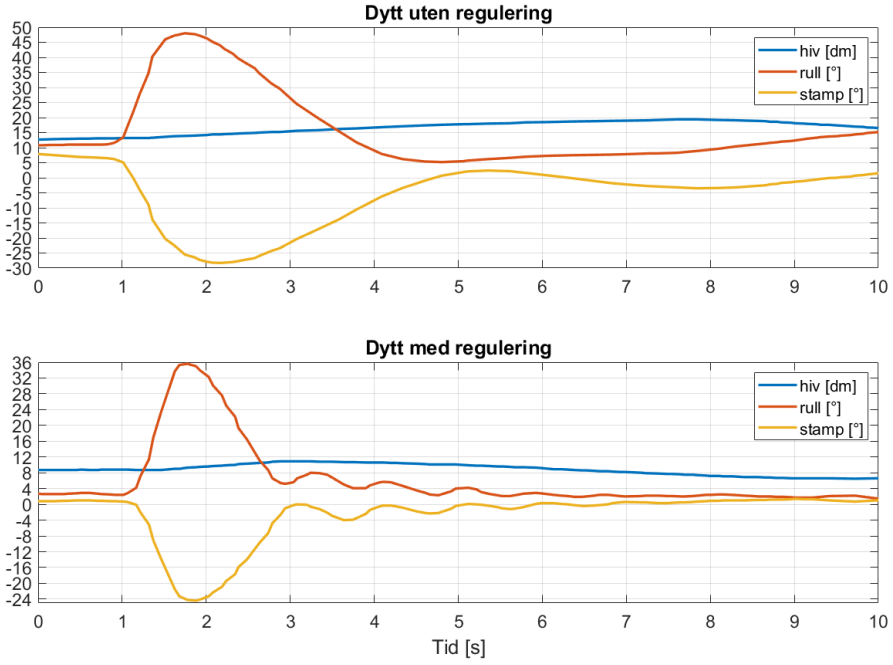


Figur 8.19: Grafer over rullvinkel, stamvinkel og dybde når det festes et lodd på manipulatoren. Øverste plott viser testen utført uten regulering, og nederste plott viser testen med regulering. Positiv hiv-måling tilsvarer dybden ROV-en befinner seg på. Positiv rullmåling tilsvarer rullvinkel til høyre, og positiv stampvinkel tilsvarer tipping av fronten oppover, eller steiling.

Det øverste plottet viser tydelig at når loddet festes til manipulatoren, ca. 1 sekund inn i testen, tipper ROV-en fremover. Den får et maksimalt utslag på ca. 38° nedover rundt 4-5 s ut i testen. Mellom 9 og 11 s ut i testen, ser man at stampvinkelen svinger litt tilbake. Dette kommer av at loddet treffer bunnen av bassenget. Det at stampvinkelen beveger seg mot 0, etter dette kommer av at navlestrengen drar bakenden ned mot bunnen av bassenget igjen. Hiv-målingene viser at ROV-en først ligger rett under vannoverflaten av bassenget. Merk at når loddet festes, tar det ca 1-2 s før man ser utslag på hiv-målingen. Dette kommer av at bakenden av ROV-en, hvor trykksensoren er plassert, tipper oppover i det fronten dras ned. Hiv-målingen viser dybdeendring først når stamvinkelen når et visst utslag, og ROV-en dras nedover i bassenget. Det at dybdemålingen viser at ROV-en synker selv etter 9 s, når loddet treffer bakken, er fordi bakenden av ROV-en dras ytterligere ned av navlestrengen. Målingene av rull-vinkel er ikke like relevant i denne testen.

I det nederste plottet kan man se at loddet festes til manipulatoren rett etter at det har gått 1 sekund. ROV-en tipper fremover til ca. 16°, mellom 2 og 3 s inn i testen, men regulatoren kompenserer, og får den til å rotere tilbake til rundt 8°. Etter dette stabiliserer stamvinkelen seg på mellom 8° og 12°. Grunnen for at den ikke klarer å regulere vinkelen opp til 0°, kommer antakeligvis av at I-leddet i regulatoren er fjernet. Dette gir grunn til å tro at en PD-regulator ikke nødvendigvis er det beste alternativet. Hiv-målingene viser at regulatoren holder ROV-en på en dybde på rundt 1 meter nokså jevnt under hele testen. Rull-målingene svinger veldig. Disse oscillasjonene kommer antageligvis fra at rull- og stamp-regulatorene påvirker hverandre. Dette kan muligens være et resultat av at regulatorparametrene for rull og stamp er bestemt med samme responstid, eller båndbredde. Utslaget fra oscillasjonene er likevel veldig små.

I siste test ble det påført et kraftig dytt i høyre hjørne fremme på ROV-en. Figur 8.20 vises resultatet etter gjennomføringen både med og uten regulator.



Figur 8.20: Grafer over rullvinkel, stamvinkel og dybde når ROV-en dyttes nedover i ett av hjørnene. Øverste plott viser testen utført uten regulering og nederste plott testen med regulering. Positiv hiv-måling tilsvarer dybden ROV-en befinner seg på. Positiv rullmåling tilsvarer rullvinkel til høyre, og positiv stampvinkel tilsvarer tipping av fronten oppover, eller steiling.

Det øverste plottet viser at dyttet, påført etter 1 s, gir store utslag i rull- og stamp-vinkel, med nesten 50° utslag i rullvinkel og 30° i stamvinkel. Man ser effekten av det naturlige opprettingsbidraget ved at vinklene faller tilbake etter responsen også uten regulering. Dette tar rundt 4 s fra dyttet skjer. Det at ROV-en ikke er perfekt balansert, sammen med påvirkningen fra navlestrengen, gjør at vinklene ikke faller til 0°. ROV-en dyttes også fra en dybde på rundt 1.4 m, ved 1 s, til en dybde på rundt 2 m ved 8 s. I dybde dyttes altså ROV-en ned rundt 60cm, før den på grunn av oppdriften flyter oppover igjen.

I det nederste plottet skjer også dyttet rundt 1 s ut i testen. Utslaget i rull-vinkel blir rundt 35°, og i stamp-vinkel rundt 25°. Dette er altså et vesentlig mindre utslag enn når regulatorene ikke er aktive. Likevel kan man ikke fra dette konkludere med at forskjellen i utslag kun stammer fra regulatoren. Kraften i dyttet er mest sannsynlig forskjellig i de to testene, og det er vanskelig å si om endringen i utslag kunne vært mindre, eller enda større. Det tar i dette tilfellet omtrent 2 s før regulatoren har stabilisert ROV-en igjen, som er ca. dobbelt så raskt som uten regulering. I tillegg oppfører ROV-en seg mye mer stabilt også etter at den har svinget seg tilbake til rundt 0°. Dette ser man ved at vinkelmålingene ligger noenlunde jevnt på samme nivå helt til testen avsluttes. Hiv-målingene viser at ROV-en dyttes nedover fra rundt 85 cm, til rundt 110 cm mellom 1 og 3 s inn i testen, før den etter ca. 7 sekunder ut i testen har regulert seg tilbake. Det tar altså nokså lang tid for hiv-regulatoren å kompensere for dybdeendring. I tillegg ser man at etter 7 s beveger ROV-en seg over referansedybden, og det kan minne om et oversving. Det at hiv-reguleringen i denne testen virker å være litt treg, vitner om at regulatoren muligens kunne vært satt opp med lavere responstid.

8.6 Konklusjon

I dette kapittelet har vi utviklet reguleringssystemet for ROV-en. Det er realisert tre separate PID-regulatorer som regulerer hver sin frihetsgrad. Parametererene til disse regulatorene ble først funnet teoretisk ved hjelp av analytiske metoder, samt ved simulering. Metodene som ble tatt i bruk, resulterte i forskjellige sett med teoretiske regulatorparametre. De ulike parametersettene ble simulert opp mot hverandre, og det settet vi vurderte til å gi den beste responsen, ble også manuelt tunet. Under simuleringen av regulatorene med de utvalgte parametrene, gjorde vi noen kvantitative mål, hvor noen av disse er listet opp under:

- Ved et simulert sprang på 30 cm i hiv-retning tar det ca. 5.6 s før dybden er innenfor 5% av spranget. Dette med et oversving på ca 25%.
- Ved å slippe ROV-en fra en rullvinkel på 45° i simuleringen, får vi et oversving på ca. 14%, og det tar ca 2.4 s før vinkelposisjonen er innenfor 5% av «spranget».
- Ved å slippe ROV-en fra en stampvinkel på 45° i simuleringen, får vi et oversving på ca. 14%, og det tar ca 2.8 s før vinkelposisjonen er innenfor 5% av «spranget».

Selv med tilfredstillende respons i de simulerte prosessene, viste det seg at parametrene ga en altfor aggressiv regulering når det fysiske systemet ble testet. Grunnen for dette ble vurdert til å være den lave responstiden vi satte under simuleringen, og det ble derfor funnet nye parametre med en mye høyere T_C. Dette dempet oscillasjonene som oppstod.

Selv om testen ble gjennomført med begrenset tid igjen av prosjektet, fikk den likevel bekreftet at vi har satt opp et reguleringssystem som forbedrer ROV-ens stabilitetsegenskaper. I tillegg fikk vi justert regulatorparametrene etter det fysiske systemet, og ikke bare basert på simuleringer. De er derimot ikke finjustert etter presise og gode tester, slik som vi egentlig skulle ønsket.

Testene utført i dette kapittelet er også filmet. Videoen er tilgjengelig via denne linken:
`https://www.youtube.com/watch?v=KWpERBYcOKY&t=32s&ab_channel=UiSSubsea`

Rådataen fra testene utført i dette kapittelet, samt MATLAB-filen for behandlig og plotting av dataen, kan lastes ned via linkene i vedlegg A.

Kapittel 9

Diskusjon

Kapitteloversikt

Det har gjennom arbeidet med styring- og reguleringssystemet blitt utført mye variert arbeid. I dette kapittelet diskuteres resultatene fra de ulike hoveddelene av prosjektet. I tillegg til arbeidet med prosjektoppgaven, ble Tomas Royal Choat utnevt til prosjektleder, og Otto Nessa Ljosdal utnevnt til sponsoransvarlig. I forbindelse med dette, vil det også diskuteres rundt det overordnede prosjektarbeidet.

9.1 Motorer og motorkontrollere

Valget av motorer for manøvrering av undervannsfarkosten endte på T200 thrustere produsert av *Blue Robotics*. Thrusteren er bygd opp av en ferdig vanntettet BLDC-motor, med propeller og motorhus. Den teoretiske gjennomgangen av ulike typer elektriske motorer viste at en BLDC-motor egner seg godt for akkurat dette formålet. Ingen bevegelige deler er i fysisk kontakt med hverandre, og vedlikeholdsbehov vil derfor være minimalt. Motorene egner seg også godt for høye rotasjonshastigheter, som er et behov for at thrusterne skal produsere et tilstrekkelig kraftbidrag i vann. Thrusterene ble testet i et badekar, og den maksimale thrusten ble målt til å være omtrent 3.5 kg kraft. De fungerte godt etter hensikten sin på den ferdige farkosten, men pådraget måtte nedskaleres for å forsikre jevn styring i alle retninger. Siden propellene viste seg å være mer effektive i en retning, ble pådraget i denne retningen dempet med opp til tyve prosent for å sikre lik styring i begge retninger. Dette er dokumentert i testkapittelet i vedlegg B.1.

Valget av thrustere er vi godt fornøyde med. De fungerte godt, og valget av disse over andre thrustere sørget for rask fremgang i prosjektet. Likevel kunne et annet valg av motorkonfigurasjon sørget for bedre kontroll i samtlige frihetsgrader. Et alternativ kunne være å montere en thruster i hvert av de åtte hjørnene av farkosten, med 45° utslag i forhold til alle de tre sidekantene. Dette ville krevd mer omfattende pådragsberegninger, men muligens bedre grunnlag for styring og regulering i flere retninger. Et annet alternativ vi ville vurdert for fremtidige prosjekt er å plassere alle åtte thrusterne på ett horisontalt plan sammen med massesenteret og volumsenteret. Dette ville gjort thrusterne mer effektive, og samtidig tillatt ROV-en å kunne kjøre fritt i alle retninger, samt å kunne stabilisere seg i en vilkårlig orientering.

De tre manipulatormotorene er av typen *Eaglepower 3508*. Disse er også av typen BLDC, og ble på samme måte som thrusterne valgt basert på fordelen med lite vedlikehold, men i tillegg på grunn av betydelig vektreduksjon relativt til for eksempel en stegmotor. Den høye rotasjonshastigheten er ikke like heldig for en manipulatormotor, men dette kunne blitt kompensert for av manipulatorgruppen gjennom nedgiring. På grunn av kravet om vanntetting ble motorene også støpt inn i termisk epoksy. Foreløpig er motorene kun testet på en begrenset versjon av manipulatoren, men de fungerte bedre enn forventet. Utfordringene vi møtte, var i form av unøyaktigheter i manipulatorens design.

Man kunne selvsagt brukt en stegmotor som alternativ til BLDC-motoren. Denne ville vært enklere å drive på lave rotasjonshastigheter, og i tillegg gitt bedre presisjonskontroll av manipulatoren. Disse egenskapene ble ikke prioritert på grunn av konkurranseoppgavenes kompleksitetsnivå og strenge vektkrav. Dessuten ville også stegmotorer gitt et økt vektbidrag til ROV-en.

Motorkontrolleren av typen *Basic ESC* fra *Blue Robotics* ble valgt til å drive samtlige elleve motorer. Disse ble valgt på grunnlag av at de er designet for T200-thrusteren, og samtidig for enkelhetens skyld ved å bruke de samme for manipulatormotorene. Et ønske i forbindelse med bruken av motorkontrollere, var å kartlegge den elektriske kretsen for å få innblikk i hvordan styresignalet vi sender til motorkontrolleren blir behandlet og omformet til et pådragssignal motoren kan drives på. På grunn av manglende dokumentasjon fra produsenten, tok vi i bruk et multimeter for å undersøke koblingen mellom alle punkter på kontrolleren. Vi klarte på denne måten å kartlegge oppbygningen til motorkontrolleren. Motorkontrolleren består i hovedsak av kretsene listet under.

- **PWM-inngang**: Brukes til å styre prosentvis pådrag i begge rotasjonsretninger.
- **Regulatorkrets**: Sørger for nedskalering til riktig spenningsnivå for mikrokontroller og gate-driver.
- **Transistornettverk**: Tolv MOSFET-er som sørger for kommutering av spenningssignalet mellom de tre fasene inn til motoren.
- **Gate-driver**: Styrer hvilke MOSFET-er som aktiveres.
- ***Bootstrap*-krets**: Sørger for tilstrekkelig spenningsnivå på alle *gate*-terminaler på MOSFET-ene.
- **Posisjonsavlesningskrets**: Brukes til å gi mikrokontrolleren informasjon om når man skal over i neste steg av kommuteringssekvensen.

Kretsene er tegnet opp i et kretsskjema, som ble vist i figur 3.5.

Motorkontrollerene ble testet på både thrusterene og manipulatormotorene, og fikk hver av dem til å kjøre etter ønsket oppførsel. Det ville likevel vært nyttig med en mer avansert motorkontroller for å drive manipulatormotorene. Dette kunne gitt mulighet for å styre motorene med hensyn på dreiemoment, og til og med oppnå et holdemoment med BLDC-motorer. Dersom vi hadde hatt mer tid, skulle vi også gjerne gjort videre analyser på motorkontrolleren. Det ville vært interessant å gå grundig inn på programvaren i mikrokontrolleren, og se hvordan denne behandler signalene den leser av. Dette fikk vi derimot ikke tid til, men det kan stå som et område for videre arbeid etter at oppgaven er levert.

9.2 Maskinvare

For å realisere kommunikasjon mellom de ulike delene av systemet, er det designet og produsert et kretskort. Kortet består av en tranceiverkrets for CAN-kommunikasjon, LED-lys til testing av thrusterpådrag, plugger for PWM-signalet til motorer, samt tilkoblingspunkter for et plattformkort med mikrokontroller. Utlegget for kretskortet ble tegnet i *Altium*, og kortet ble produsert av *JLCPCB*. I *Altium* ble det også lagt inn 3D-modeller av alle komponenter slik at en 3D-modell av kortet kunne plasseres virtuelt i elektronikkhuset. Dette gjorde det enkelt å sørge for at komponentene ikke kom i konflikt med andre ting i elektronikkhuset. Ettersom kortet i hovedsak er utviklet for å realisere kommunikasjon, er kretsene på kortet relativt enkle. Foruten mikrokontrolleren, er det kun en IC i form av tranceiverkretsen for CAN-kommunikasjon på kortet. Det ble verifisert at alle signalbanene var riktig koblet og at kommunikasjonen over CAN-buss virket. Dette er rapportert i testrapporten i vedlegg B.2.

Det er flere utbedringer vi ville gjort om kortet skulle blitt designet på nytt. Den viktigste endringen er å sørge for at spenningstilførselen er koblet til på riktig sted på utviklingskortet. Dette måtte endres ved hjelp av en ledningsbit på kortet. Feilen ble heldigvis oppdaget tidlig nok til at det ikke forårsaket skader på utviklingskortet.

Det er i tillegg flere endringer som ikke er kritiske for virkemåten, men som kan gjøre det enklere å bruke kortet:

- Utviklingskortet ble loddet direkte på kretskortet for å spare plass i høyden. Det ville vært mer hensiktsmessig å plassere kortet et sted som ville tillatt bruken av kontakter som enkelt lar utviklingskortet kunne fjernes. Hvis kretskortet blir skadet, er det nå unødvendig krevende å lodde av hele utviklingskortet.
- Utviklingskortet er plassert slik at det er utfordrende å koble til USB-plugg for å programmere mikrokontrolleren. Dette kunne lett blitt fikset ved å plassere kortet slik at USB-pluggen peker utover.
- Ettersom mikrokontrolleren må programmeres ved hjelp av en USB-plugg direkte på utviklingskortet, kreves det at elektronikkhuset demonteres før dette skal gjøres. Det ville vært bedre om mikrokontrolleren kunne programmeres ved hjelp av mini-PC-en slik at dette kunne blitt gjort fra toppsiden. Dette ville spart oss mye tid under feilsøking og testing.
- *PicoBlade*-pluggene som brukes for å koble motorkontrollerene til kretskortet er veldig små. Dette gir en lav profil, men disse var veldig vanskelige å montere på ledningene fra motorkontrollerene. Pluggene må festes med krympesko på ledningene, som gjør det vanskelig å erstatte en ødelagt motorkontroller med en ny. Det ville vært mer hensiktsmessig

med litt større plugger hvor ledningene kunne blitt skrudd fast. Et annet alternativ kunne vært å koble alle PWM-signalene i én større kontakt, for å gjøre oppkobling enklere.

Det ble i tillegg designet stativ for motorkontrollerene. Stativet gjør det enkelt å montere motorkontrollerene på kraftkortet, slik at ledningene som forsyner disse med 12 V blir kortest mulig. Stativene har også vifter som sørger for at motorkontrollerene kjøles ned. Det er vist gjennom simulering at viftene kan bidra til å fjerne varme fra motorkontrollerene, men dette er vanskelig å teste ettersom thrusterene må være under vann for å gi nok last for å varme opp motorkontrollerene tilstrekkelig.

Selv om det finnes forbedringspotensiale på kretskortet, virker alt av maskinvare som ønsket.

9.3 Programvare

Mikrokontrolleren som brukes i systemet er produsert av *STMicroelectronics*, og sørger for kommunikasjon med resten av elektronikken i ROV-en, samt implementering av styrings- og reguleringssystemet. Konfigurering og programmering er gjort i utviklingsverktøyet *STM32CubeIDE* som er fra samme produsent som mikrokontrolleren. Kommunikasjon er realisert med CAN-buss som er tilkoblet de andre kontrollerne i ROV-en: sensor-, kraftforsynings- og bildebehandlingskontrolleren. De andre modulene som er tatt i bruk, er TIM-moduler som brukes til å generere PWM-signal til de elleve motorkontrollerne.

Det er lagt ned en betydelig arbeidsmengde for å oppnå god programstruktur i systemet. For oversiktens skyld er programmet opprettet og inndelt med følgende struktur:

- **Filinndeling**: Det er opprettet egne filer for variabeldeklarasjoner, bestemmelse av verdier, egendefinerte *structs*, funksjoner, PID-regulatorene, testfunksjoner og avbruddshåndtering.
- **Hovedprogram**: Består av en uendelig løkke som trigges av flagg, og kaller på ønskede funksjoner deretter.
- **Avbruddshåndtering**:
  - **CAN_RX0**: Det er satt opp avbrudd på mottaksregisteret til CAN-modulen. Dette omfatter mottatt styredata, sensordata og effektforbruk, samt oppdatering av parameterverdier.
  - **SysTick**: Et periodisk systemavbrudd som skjer hvert millisekund. Her inkrementeres ulike tellere for å endre tilstander, eller sjekke for endringer i systemet.
- **Pådragshåndtering**: En egen funksjon for å samle opp bidragene til pådrag fra både styrings- og reguleringssystemet, og behandle disse. Funksjonen summerer opp alle bidrag og kontrollerer, behandler og skalerer totalpådraget basert på blant annet effektbruk og motorkarakteristikk.
- **Kommunikasjon**: Det er kartlagt 8 ulike meldinger som kommuniseres gjennom mikrokontrolleren vår. En oversikt over meldingene finnes i tabell 5.3 og 5.4, som inneholder meldingsnavn, ID, hvilke variabler disse inneholder, samt variablenes datatyper og verdiområder.

Bildebehandlingsgruppen har hatt hovedansvaret for kommunikasjon, og gjennomførte nødvendig testing for å verifisere at mikrokontrolleren vår kommuniserte med resten av systemet. Resterende programvare i systemet vårt har vi testet stegvis underveis. Både styrings- og reguleringssystemet er avhengig av fungerende programvare for at disse skal fungere. Derfor har vi for eksempel gjennom måling av PWM-signaler med oscilloskop kunnet verifisere at TIM-modulene er konfigurert som de skal, og pådragshåndtering er utført korrekt. Dette ble verifisert gjennom testen i vedlegg B.3.

Når det gjelder forbedring av programvaren kan det alltid legges ned ekstra arbeid med programstruktur og oversiktlig kode. Likevel har vi tilstrebet å lage programkode som er mest mulig leselig, og ikke minst som er modulært oppbygget. Gjennom å separere de ulike prosessene i programmet inn i moduler, legges det et godt grunnlag for videre utvikling, og ikke minst gjør det enkelt å utføre endringer som kan måtte gjøres i ettertid. Programmet danner også en struktur som lett kan brukes som grunnlag, eller skall, for fremtidige styringsprosesser.

En vesentlig forbedringsmulighet vil være å sette opp funksjoner som er i stand til å lese fra og skrive til *flashminnet*. Dette vil gi muligheten til å permanent oppdatere parametre, tilstander og andre verdier fra toppsiden. Dette er mest sannsynlig ikke funksjoner som er vanskelige å sette seg inn i, men noe vi ikke har funnet tid til å gjøre.

9.4 Styring

Styresystemet som er implementert på mikrokontrolleren tar inn styredata fra toppsiden, og kontrollerer horisontal translasjon, vertikal translasjon og rotasjon om z-aksen manuelt for ROV-en. Systemet er utviklet basert på ROV-ens bevegelse i fire av de seks frihetsgradene[1], hvor rull og stamp er de som ikke styres manuelt, men er regulert. Styring av bevegelse i de ulike frihetsgradene er beskrevet gjennom referansesystemet BFF[2], som er fiksert til ROV-en med origo i rotasjonssenteret. Det ble utført pådragsberegninger basert på styredata fra en kontroller av typen *Xbox One*, som videre sendes til motorene og sørger for bevegelse i ønsket retning.

Styresystemet ble i første omgang testet gjennom observasjon av lysdiodene montert på kretskortet. Disse er programmert for å indikere om pådragssignal sendes til thrusterene, og om pådraget skal være positivt eller negativt. Testen viste at pådragssignalene ble sendt som ønsket. Videre ble det genererte PWM-signalet til alle motorene målt med oscilloskop. Vi verifiserte gjennom denne testen at et gitt utslag på styrestikken gav en ønsket pulsbredde i pådragssignalet. Dette er dokumentert i testrapporten i vedlegg B.3. Den avsluttende testen for reguleringssystemet ble gjennomført i et av universitetets bassenger. Under denne testen fikk vi også verifisert at den manuelle styringen fungerte på det fysiske systemet.

Styresystemet til ROV-en kunne blitt forbedret gjennom eksempelvis å legge til rette for manuell kontroll av alle de seks frihetsgradene. Man kunne da brukt regulatoren for å holde ROV-ens orientering konstant når den ikke mottar styredata, og manuell kontroll for å endre referansepunktet for orientering. ROV-en ville på denne måten hatt flere muligheter for annen type oppgaveløsning, da den ville oppnådd en større grad av mulighet for bevegelse. Dette ville likevel krevd mer komplekse pådragsberegninger, og gjerne en annen motorkonfigurasjon som diskutert tidligere. Et annet konsept som ville sørget for ytterligere kontroll ville vært total posisjonskontroll. Der-

[1] Jag, svai, hiv, rull, stamp og gir
[2] *Body Fixed Frame*

som man hadde styrt posisjonen til ROV-en i et referansesystem som er fiksert til omgivelsene, ville det vært mulig med prediksjon av bevegelse i alle retninger. Dette ville muliggjort styring basert på ønsket kjørelengde i en gitt retning, som videre kunne blitt utviklet til en forbedret autonom styring. Selv om dette ville krevd ekstra sensorer, som for eksempel eksterne sonarer, kunne det absolutt være et spennende konsept å jobbe med.

9.5 Modellering og regulering

Reguleringssystemet er realisert gjennom en PID-regulator for hver av frihetsgradene hiv, rull og stamp. For å legge til rette for reguleringen ble i styringskapittelet referansesystemet NED[3] definert i tillegg til BFF. Ved å bruke to referansesystemer, beskrives ROV-ens posisjon i hiv, rull og stamp gjennom hvordan referansesystemene beveger seg i forhold til hverandre.

For å finne gode regulatorparametere laget vi i kapittel 7 modeller som beskriver hvordan ROV-en oppfører seg ved bevegelse i de ulike frihetsgradene. For å oppnå mest mulig presise modeller ble det brukt digitale verktøy for å bestemme blant annet drag-koeffisienter og treghetsmoment. Verdiene funnet fra de digitale vertøyene ble også sammenlignet med teoretiske utregningsmetoder for verifisering. Modellene er likevel basert på flere forenklinger, hvor en av de største forenklingene er gjort i forbindelse med estimering av vannmotstand under rotasjon. Det er utfordrende å bekrefte nøyaktigheten til modellene ved testing, men simuleringen gav en indikasjon på hvordan ROV-ens bevegelse i vann ville være. Modellene dannet også et godt grunnlag for utvikling av regulatorene.

Selv om kun noen av frihetsgradene reguleres, er det likevel hensiktsmessig å vite hvordan ROV-en oppfører seg i de uregulerte frihetsgradene. Derfor ble det utviklet modeller for alle frihetsgrader. Man kan fra disse gjøre estimater på for eksempel «blindkjøring». På grunn av forsinkelser i kameraoverføringen vil ROV-en, ifølge simuleringer, kunne bevege seg opp til 17.6 cm i jag-retning, 11.8 cm i svai-retning og kunne rotere 49.1° i gir-retning iløpet av forsinkelsestiden på opp mot 200 ms i kameraoverføringen.

Modellene kunne alternativt vært utledet etter at den fysiske ROV-en var produsert. Det ville blant annet vært mulig å estimere dragkraften mer nøyaktig basert på målt topphastighet og kjent pådrag. Dette ville sannsynligvis gitt enda bedre presisjon i beskrivelsen av ROV-ens bevegelser, og følgelig en mer optimalisert regulator.

Videre skulle de utledede modellene brukes for å sette opp regulatorer. På grunn av vannmotstanden er modellene ulineære, og for å kunne regne overføringsfunksjoner måtte det velges et arbeidspunkt som modellene lineariseres rundt. Basert på tilstandsrommodellene, og valgte arbeidspunkter, utledet vi overføringsfunksjoner for de regulerte frihetsgradene. Parametre for hiv-regulator ble funnet gjennom Skogestads metode, og for rull- og stamp-regulator gjennom metoden *IMC*. Det er også funnet regulatorparametere ved testing av det fysiske systemet i et basseng. Tabell 9.1 viser de utregnede overføringsfunksjonene og parametrene, samt parameterene funnet ved testing.

[3] *North East Down*

Tabell 9.1: Tabell over overføringsfunksjoner og oppnådde regulatorparametere for de regulerte frihetsgradene.

Frihetsgrad	Hiv	Rull	Stamp
Valgt arbeidspunkt	$\dot{z} = 0.1$ m/s	$\phi = 0°$, $\dot{\phi} = 15°/s$	$\theta = 0°$, $\dot{\theta} = 15°/s$
Overføringsfunksjon	$\frac{1.03}{(0.52s+1)s} \cdot e^{-0.2s}$	$\frac{2.9 \cdot e^{-0.2s}}{0.36^2 \cdot s^2 + 2 \cdot 0.23 \cdot 0.36 \cdot s + 1}$	$\frac{2.36 \cdot e^{-0.2s}}{0.46^2 \cdot s^2 + 2 \cdot 0.28 \cdot 0.46 \cdot s + 1}$
Parametere fra overføringsfunksjon	$K_p = 3.23$	$K_p = 0.14$	$K_p = 0.27$
	$K_i = 1.52$	$K_i = 0.86$	$K_i = 1.06$
	$K_d = 1.27$	$K_d = 0.11$	$K_d = 0.22$
Parametere tunet fra simulering	$K_p = 3.8$	$K_p = 0.14$	$K_p = 0.27$
	$K_i = 1.3$	$K_i = 0.35$	$K_i = 0.15$
	$K_d = 1.5$	$K_d = 0.15$	$K_d = 0.27$
Parametere tunet fra fysisk test	$K_p = 1.049$	$K_p = 0.041$	$K_p = 0.076$
	$K_i = 0.452$	$K_i = 0$	$K_i = 0$
	$K_d = 0.425$	$K_d = 0.032$	$K_d = 0.063$

Regulatorene med de teoretisk tunede parametrene ble testet i *MATLAB Simulink*. For hiv ble følgeegenskapene for et sprang på 30 cm testet, og for rull og stamp ble de testet ved å sette et initielt vinkelutslag på 45°. Resultatene etter testen er vist i tabell 9.2.

Tabell 9.2: Simulert regulatorprestasjon ved sprang i referanse.

Frihetsgrad	Hiv	Rull	Stamp
Sprang	0.3 m	45 °	45 °
Oversving	7.56 cm	6.5 °	6.4 °
Oversving %	25 %	14 %	14 %
Stigningstid	1 s	0.6 s	2.4 s
5 % Settling time	5.6 s	0.46 s	2.2 s

Da arrangøren av oppgaven oppga at ROV-en kunne måtte løfte opptil 10 N i vann, testet vi også regulatorene for forstyrrelser under simulering. For hiv ble det lagt til 10 N i kraftsummen, og for rull og stamp 5 Nm, eller 10 N fungerende 0.5 m fra rotasjonssenteret, som representasjon av en forstyrrelse. Resultatene fra forrstyrrelsestesten er vist i tabell 9.3.

Tabell 9.3: Simulert regulatorprestasjon ved forstyrrelse.

Frihetsgrad	Hiv	Rull	Stamp
Forstyrrelse	10 N	5 Nm	5 Nm
Utslag	13.8 cm	45.7°	38.5°
5 % Settling time	4.7 s	2.4 s	2.3 s

Som vist i tabell 9.1 ble parameterene funnet ved testing på det fysiske systemet mye lavere enn de teoretiske parametrene. Årsaken til dette er at kravet om en responstid på 0.2 s for systemet var for stort, og systemet begynte å oscillere som følge av en aggressiv regulator. Parametrene som videre ble testet ble derfor funnet med en forventet responstid på 1 s for hiv, og 1.2 s for rull og stamp. Disse responstidene ble valgt basert på testing, og visuell observasjon av ROV-en.

Selv med en økt responstid, oppstod det oscillasjoner for rull- og stamp-prosessen. Dette ble vurdert til å være på grunn av en veldig stor K_i. Ved å sette denne til 0 for disse prosessene

ble det observert en bedret oppførsel. Vi endte altså opp med å regulere rull og stamp med en PD-regulator. Grunnlaget for dette valget, var kun basert på visuell observasjon av ROV-ens oppførsel. Testene utført på systemet viste at regulatorene bidro til bedre stabilitet enn uten regulering. Disse testene ble gjennomgått i delkapittel 8.5. Det ble på grunn av begrenset tid, ikke gjort noen kvantitative mål regulatorenes egenskaper.

Selv om testene viste at regulatorene bidro til bedre stabilitet, viste datasettene etter kjøringen at det er rom for forbedring. Uten I-ledd oppstår det et kontinuerlig avvik i rull- og stamp-vinkel fra referanseverdien på 0°, selv med et naturlige opprettingsbidraget. Det kan derfor tenkes at en PD-regulator ikke var det beste alternativet, og at det er nødvendig med et I-ledd, bare med mye mindre K_i enn det IMC-metoden gir. Ettersom det også fortsatt oppstår små oscillasjoner i rull- og stampvinkel under regulering, er det mulig at en enda høyere T_C for disse prosessene er nødvendig. Oscillasjonene kan også dempes ved å bruke forskjellig T_C for rull- og stamp-prosessen. Dette vil gjøre at de to overføringsfunksjonene får ulik båndbredde, som igjen vil gjøre at reguleringen av dem vil påvirke hverandre i mindre grad. Dette er noe vi ønsker å teste etter innlevering av oppgaven.

Med mer testing kunne parametrene blitt optimalisert. Det er viktig å presisere at testene som er utført, er basert på hva som oppfattes som bra visuelt sett. Testene er altså mer kvalitative enn kvantitative, men viser likevel at regulatorene er i stand til å forbedre stabiliteten til systemet.

For rull og stamp kunne det som en tilleggsoppgave blitt implementert regulering etter ønsket referansevinkel i stedet for stabilitetskontroll med konstant referanse på 0°. Ved å sette en referansevinkel kunne man fullført flere oppgaver mer uavhengig av en kompleks manipulatorarm, og det ville økt rekkevidden for gripemuligheter betraktelig. En slik regulator ville krevd pådragsberegninger for å motvirke drift, da ROV-en ville opplevd kraftkomponenter fra thrusterne i horisontal retning. Dette var noe vi ikke fikk tid til, men vil kunne stå som en mulighet for videre utvikling.

9.6 Overordnet prosjektarbeid

I forbindelse med deltakelse i prosjektet, satte vi i starten opp en fremdriftsplan med tidsanslag på de ulike arbeidsoppgavene. I figur 9.1 vises fremdriftsplanen.

Fremdriftsplan

Uke	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	Tidsanslag	Tidsbruk	Progresjon	Ansvarlig
Overordnet design																							
Valg av motorer																				20	19	95,0%	
Kretskortdesign																				30	28	93,3%	
Research/teori																				250	164	65,6%	
Møter/Planlegging																				300	299	99,7%	
Thrustere																							
Elektronikkdesign																				40	47	117,5%	
Styring av thrustere																				50	61	122,0%	
Matematisk modellering																				120	115	95,8%	
Regulering av thrustere																				200	125	62,5%	
Programmering																				80	76	95,0%	
Manipulator																							
Elektronikkdesign																				40	71	177,5%	
Styring av manipulator																				60	24	40,0%	
Programmering																				60	35	58,3%	
Testing																							
Test av Kretskort																				40	30	75,0%	
Test av Thrustere																				60	74	123,3%	
Test av Manipulator																				80	20	25,0%	
Test av Regulering																				120	159	132,5%	
Dokumentasjon																							
Rapportskriving																				450	536	119,1%	
Sum																				2000	1883		

Fargekoder: Avsatt tid Progresjon Fullført modul

Figur 9.1: Fremdriftsplan. Rutene markert med blått er tid som er avsatt til den aktuelle aktiviteten men som ikke har blitt benyttet. Rutene markert med lilla viser progresjonen, og rutene markert med grønt viser når modulen ble fullført.

For det meste har målene satt i fremdriftplanen blitt overholdt. Tidsbruken avviker ikke mye fra tidsanslaget, bortsett fra testing av manipulatoren. Det ble brukt en del mindre tid på styringen av manipulatoren ettersom det ble nødvendig å ta i bruk en forenklet manipulator, som ble laget som en reserveløsning. På grunn av forsinkelser med realiseringen av hovedmanipulatoren, var det lite testarbeid som var mulig å gjennomføre. Test av regulering er heller ikke markert som fullført ettersom det fortsatt er behov for en del testing og tuning av regulatorene.

Delkapittelet vil videre ta for seg noen av aspektene vi har vært borti i forbindelse med det overordnede prosjektarbeidet.

9.6.1 Prosjektledelse

Bachelorprosjektet Fenris og Frøya er til sammen en prosjektgruppe av 21 studenter, som jobber med 9 forskjellige bacheloroppgaver. Alle underprosjektene er avhengige av samarbeid på tvers av gruppene, om enn noen mer enn andre. For å forsikre at prosjektet går fremover på rett måte, hvor alle drar i samme retning, er det helt nødvendig å ha en sentral ledelse. Denne ledergruppen må stå i spissen for at det er en felles visjon og plan som alle kan, og ønsker, å følge. Prosjektledelsen for prosjektet er som nevnt i kapittel 1.1:

Prosjektleder: Tomas Royal Choat

Teknisk ansvarlig programvare: Christoffer Næss

Teknisk ansvarlig maskinvare: Mats Røste

Oppgavene for prosjektledelsen har variert en del i de forskjellige fasene av prosjektet. Det som var innlysende fra starten var at hvordan tiden blir forvaltet gjennom hele perioden, er kritisk avgjørende for å komme i mål i tide. Prosjektet har kun fire måneder fra start til slutt, fra

tidligste planleggingsfase til det skal produseres videobevis av en operativ ROV i basseng som kvalifiserer den til MATE ROV Competition. UiS Subsea har mange års erfaring med å utvikle ROV-er, men det er over 4 år siden sist en ROV fra bachelorprosjektet har vært ferdigstilt innen oppgaven skal leveres. Denne informasjonen satte alvoret i prosjektledelsen, slik at det ble lagt vekt på en grundig planleggingsfase i starten av prosjektet. Tidligere i utdanningsløpet har vi gjennomført prosjekt med innslag av systemteknikk, og det har derfor vært fundamentet som prosjektledelsen for Fenris og Frøya har vært bygget på.

Planleggingsfasen:

Det viktigste i starten av et prosjekt er å kartlegge behov. Hva skal realiseres, hvilke problemer skal løses og hvilke begrensninger er satt. Disse punktene er sentrale både for prosjektet i sin helhet, men også for de enkelte prosjektgruppene. I samme grad er det avgjørende å identifisere når forskjellige milepæler i prosjektet må være ferdigstilte. Dette er også sentralt både overordnet i prosjektet, men også innad i hver gruppe. Umiddelbart var det én tidsfrist som sto ut; innleveringsdato for både bacheloroppgavene óg kvalifiseringsfrist for konkurransen, 15. mai. Ut av respekt for innspurten mot levering av en bacheloroppgave, ble de siste to ukene satt av til rapportskriving. I tillegg anslo vi at det var behov for en minimum to ukers periode for å teste ROV-en i basseng. Fristen for å ha en ROV klar til testing ble derfor 15. april. Med andre ord hadde gruppene tre måneder på å planlegge, designe og produsere sine deler. Med en noe flytende overgang mellom fasene, hadde gruppene omlag en måneds planlegging og to måneder på å designe og produsere sin oppgave.

Første del av planleggingen er, som nevnt over, behovsanalyse. I vårt prosjekt var det naturlig at neste steg var å analysere erfaringene fra tidligere års prosjekt. Hvilke løsninger som har fungert bra, og prøve å identifisere hva som er årsaken når prosjektene møter kritiske utfordringer. Deretter er neste steg å planlegge hvordan den mest kritiske ressursen skal forvaltes; tid. Alle prosjektgruppene måtte legge opp en aktivitetsplan som beskriver når og hvor mye tid som skal brukes på forskjellige arbeidsoppgaver i prosjektet.

I praksis resulterte planleggingsfasen i ukentlige fellesmøter med representanter fra alle gruppene, i tillegg til daglige møter med en eller flere grupper. Så snart behovsanalysen var i boks, gikk det over i kartlegging av grensesnitt mellom prosjektoppgavene. Med et sammensatt og tverrfaglig prosjekt som Fenris og Frøya, er det også en mengde arbeidsoppgaver som faller utenfor, eller imellom de definerte oppgavene i forhold til bacheloroppgaven. Det faller også på prosjektledelsen å ha oversikt over disse oppgavene, og ta ansvar for eller delegere de ut der det er ledige ressurser.

Utviklingsfasen:

I perioden hvor gruppene jobber med design og produksjon av sin oppgave, skifter fokuset av lederjobben til en mer individuell oppfølging. Etter en vellykket planleggingsfase er gruppene allerede ensrettet, og det er ikke like stort umiddelbart behov for samarbeid mellom gruppene. Ukentlige fellesmøter fortsatte som før, men agendaen her var sentrert rundt fremdrift og komplikasjoner:

- Hva har blitt gjort den siste uken?
- Hva skal gjennomføres frem til neste møte?
- Har gruppen møtt på uforutsette utfordringer?
- Hva trenger gruppen for å løse utfordringen?

- Hvordan ligger gruppen an i forhold til aktivitetsplanen?
- Informasjon om fellesbestillinger, leveringstid av komponenter og diverse økonomi i denne sammenhengen.

Også i denne delen av prosjektet var det mange tilleggsoppgaver som måtte utføres, og sørge for at nødvendige fasiliteter som maskinverksted, kjemilaboratorium, elektrolaboratorium, 3D-printere og basseng er tilgjengelige for gruppene som har behov.

Sluttfasen:

Opplevelsen av sluttfasen er som regel et resultat av den akkumulerte innsatsen lagt inn i de foregående fasene av prosjektet. Det er likevel et velkjent fenomen at utviklingsprosjekter alltid tar vesentlig lenger tid enn først anslått. Ukemøtene møtte igjen et skift i innholdet. Fokuset ble nå ført tilbake til samarbeid på tvers av gruppene. I første omgang et fokus på ferdigstilling og testing av enkeltmodulene, og så oppkobling og testing med nabomoduler. Det blir raskt klart at de gruppene som la i innsats i å planlegge og ta fremdriftsplanen på alvor, hadde et mye bedre utgangspunkt for ferdigstillelsen av det sammensatte systemet. Alle gruppene møtte på feil med sine enheter, enten i maskinvaren eller programvaren, men alvorlighetsgraden av manglene gjenspeiler tidligere innsats og grundighet i prosjektet.

Etter et par uker testing med flere alvorlige komplikasjoner, ble det klart at det var nødvendig med ekstraordinære tiltak for å feilsøke og fikse problemene i tide. Det ble innkalt til et krisemøte den 28. april for å diskutere løsninger. En uke med lange kvelder og helgearbeid, og en berg-og-dalbane av motivasjon og følelser, skulle til før prosjektet endelig kom i mål.

Erfaringer:
Overordnet sett har prosjektet vært en solid suksess, en imponerende bragd å få gjennomført på så kort tid. Spesielt tatt i betraktning at ingen i prosjektgruppen hadde noen som helst forkunnskap eller erfaring i forhold til ROV-er eller generell *subsea*-teknologi. Det har likevel vært noen frustrerende utfordringer som kunne, og burde, vært unngått. Selv om det er tildelt lederstillinger, så er vi alle medstudenter ved dagens ende. Krav til innsats og resultater fra gruppene er basert på en gjengjeldt respekt og forståelse at alle ønsker at prosjektet skal lykkes. Dette er ikke alltid nok for at alle skal holde motivasjonen oppe, og svikt i et ledd på denne måten blir mer tydelig ettersom prosjektet utfolder seg. Til tross for at det blir oppdaget tidlig i prosjektet, er det begrenset hvor mye som kan gjøres for å kompensere for forsinkelsene og manglene. Det er også tydelig at gruppene motiverer og hjelper hverandre når de velger å oppholde seg i nærheten med de andre prosjektgruppene. Grupper som i større grad isolerer seg, faller også utenfor arbeidsmiljøet og den informasjonsflyten som er tilgjengelig ved å oppholde seg sammen med resten av prosjektgruppene. Dessverre er en svært sentral utfordring i denne sammenhengen at arbeidsrommet til UiS Subsea ikke på langt nær er stort nok til å akkommodere alle medlemmene. Et tiltak for å minimere effekten av å måtte dele opp gruppene, har vært å prioritere at alle representerte linjer har kontorplass, slik at alle ingeniørlinjene er representert. I tillegg har det vært målet å skape et miljø hvor alle føler seg velkommen og hvor alle kan trives. For å motivere gruppene til å jevnlig besøke kontoret, er det blant annet alltid gratis kaffe og noe søtt til. Det er også kjøleskap og minikjøkken slik at det innbyder til å samles til lunsj.

En annen situasjon som hadde betydelig påvirkning på prosjektet var at to grupper hadde medlemmer som, for helt gode legitime grunner, ikke kunne bli med i arbeidet den første måneden. I tillegg til arbeidstiden som var tapt, gikk de også glipp av mesteparten av planleggingsfasen, som førte til ytterligere forsinkelser.

ROV-prosjektet er i utgangspunktet et særlig krevende prosjekt, og innebærer mye ekstra arbeid og innsats utover den tilknyttede bacheloroppgaven. Utkommet av prosjektet avhenger at mange er med i den ekstra innsatsen det krever, og her er et av områdene vi ville prioritert å gjøre forandringer for fremtidige prosjekt. For mye av det ekstra arbeidet falt på prosjektledelsen, og burde heller være delegert i større grad. Det var passelig med tre medlemmer i selve prosjektledelsen, men flere av de resterende medlemmene burde være innstilt på å ta ansvar for diverse oppgaver.

I sin helhet er konklusjonen at selv om det er rom for forbedring, så har det vært en fenomenal innsats av bortimot alle prosjektgruppene. Samarbeidsvillighet har aldri vært noe problem, og det har oppstått langt færre interne konflikter enn kan forventes i et slikt omfattende prosjekt. I mangel på et mer beskrivende ord, har prosjektet vært en fremragende suksess.

Link til video av gjennomført kvalifiseringsoppgaver:
`https://www.youtube.com/watch?v=vSQ23oaLuho`
Merk at oppgavene i videoen ble gjennomført før reguleringen kunne implementeres fullstendig, på grunn av manglende sensordata.

9.6.2 Økonomi og miljø

For å gjennomføre et prosjekt av denne størrelsen er det behov for et godt økonomisk grunnlag. Universitetet i Stavanger er den største bidragsgiveren for organisasjonen, men UiS Subsea blir også støttet av flere sponsorer fra næringslivet, som bidrar med kompetanse, komponenter og økonomisk støtte.

Otto Nessa Ljosdal ble utnevnt til å være en av de sponsoransvarlige, og har jobbet aktivt opp mot universitetet og bedrifter fra næringslivet. Erfaringen fra dette er at UiS Subsea er en attraktiv organisasjon både som sponsorobjekt, men også som kilde til nye ansatte. Mange av bedriftene som har blitt kontaktet har stilt med økonomiske midler og komponenter etter behov. Det ble for første gang på flere år arrangert en «Subsea-Dag» hvor flere av sponsorbedriftene og andre subsea-relaterte bedrifter stod på stand. Dette ga bedriftene muligheten til å treffe UiS Subsea-teamet, samt studenter fra hele universitetet.

Økonomiansvarlig har utviklet et budsjett for organisasjonen for året 2022. Et utsnitt av dette budsjettet er vist i figur 9.2.

Rov:		
ROV og float-desgin (mas)	kr	10 000,00
Manipulatorer (mas)	kr	7 000,00
Operatørgrensesnitt (data)	kr	5 000,00
Sesnsorkort og internkommunikasjon (el)	kr	7 000,00
Strømregulering (el)	kr	10 000,00
Styring og regulering (el)	kr	3 000,00
Bildebehandling (el)	kr	10 000,00
Float (el)	kr	8 000,00

Figur 9.2: Utsnitt fra budsjettet for UiS Subsea, 2022

Som vist i budsjettet er det satt av 3000,- til vår gruppe. Dette er fordi selve oppgaven krever lite fysiske komponenter. Thrustere skrives på en felles post ettersom disse utgjør en stor kostnad, og er relevant for flere av gruppene. I tabell 9.4 vises utgiftene spesifikt for vår gruppe.

Tabell 9.4: Tabell over utgifter for gruppen

Komponent	Pris
Kretskort	Ca. 400,-
PicobBlade	323,-
Div. komponenter	Ca. 200,-
Epoksy	1640
Motorer til manipulator	1208
SUM	3771

Her er utgiftene for motorer til manipulator, og epoksy for å tette disse, også ført opp. Dette fordi disse har blitt kjøpt inn og vanntettet av vår gruppe. Dette er strengt tatt deler til manipulatoren og skal ikke dekkes av vårt budsjett. Hvis man regner bort kostnaden av disse har gruppen brukt under 1000,- totalt.

Ettersom formålet med prosjektet er å delta på årets *MATE ROV Competition*, har prosjektet et miljøbasert grunnlag. Konkurransen fokuserer på å bruke subsea-teknologi til bærekraftige formål. Alle oppgaver tar for seg noen av FNs bærekraftsmål og hvordan en ROV kan brukes til å løse reelle utfordringer innenfor disse temaene. Det har også vært et mål å gjenbruke utstyr som allerede er kjøpt inn. Mikrokontrolleren og flere komponenter, som motstander og kondensatorer, er tatt av UiS Subsea sin lagerbeholdning. Det ble gjort et godt arbeid med å kartlegge hva som allerede var tilgjengelig før det ble bestilt inn nye deler. Noen komponenter, som for eksempel kretskortet, var nødvendig å bestille fra utlandet. Dette har en negativ påvirkning på miljøet ettersom disse komponentene må transporteres over store avstander. Der det har vært mulig har vi brukt lokale forhandlere til å kjøpe inn komponenter.

9.6.3 Tverrfaglig samarbeid

Deltakelse i ROV-prosjektet har medført god lærdom i forbindelse med samarbeid på tvers av ulike fagretninger, og det å ha erfaring med å være en del av et større prosjekt kan gi gode inngangsverdier før vi etter endt utdanning skal ut i arbeidslivet. Å jobbe med en bacheloroppgave som også avhenger av arbeidet til andre grupper, har gitt oss et innblikk i hva som kan være de største fordelene, men også utfordringene, med slikt samarbeid.

Allerede i startfasen av prosjektet merket vi effekten av at prosjektarbeid sammen med en stor gruppe bidrar med økt motivasjon for fremdrift. Man er til en hver tid avhengig av at alle grupper fullfører sitt planlagte arbeid. Dette har medført stort engasjement hos alle gruppemedlemmer, og vi har selv merket følelsen av stor tilhørighet til prosjektet. En annen stor fordel med et stort prosjekt som dette, er muligheten man får til å utvikle et komplett produkt. Dersom vi hadde jobbet med et eget prosjekt uten deltakelse fra andre bachelorgrupper, hadde vi ikke klart å realisere et like stort system.

I forbindelse med prosjektarbeidet fikk vi virkelig erfare hvor viktig det er med god planlegging og kommunikasjon underveis. For å kunne teste de ulike delsystemene opp mot hverandre, har det vært viktig å sette tidsfrister på når de ulike delene skal være ferdig utviklet. Utover ansvars-

området til hver bachelorgruppe, skulle det også løses en rekke oppgaver som var nødvendige for å få ferdig det komplette systemet. For at alle gruppene skulle kunne jobbe med sitt tildelte ansvarsområde, samtidig som det komplette systemet ble bundet sammen, har det krevd god koordinering.

Selv om det å jobbe sammen med andre grupper har medført mange gode erfaringer, har vi også hatt utfordringer underveis. Det å utvikle et styrings- og reguleringssystem krever at ROV-en kan vannsettes før systemet kan testes fysisk. Vi har fått gjennomført tester på at styresignalene blir behandlet riktig, men reguleringssystemet krever et fysisk system som kan reguleres. Mot slutten av prosjektet støtte flere av gruppene på små problemer som gjorde at ROV-en ikke var fullstendig klar for testing før det kun gjenstod noen dager til innleveringsfrist av bacheloren. Derfor ble testen av reguleringssystemet mindre omfattende enn vi hadde håpet. Dette er et godt eksempel på hvor avhengig man er av at alle deler av prosjektet ferdigstilles for at man skal få teste sitt eget arbeid.

9.7 Videre arbeid

Etter innlevering av bacheloroppgaven gjenstår det fortsatt litt arbeid med prosjektet. Hovedfokuset vil være å gjøre ROV-en så operativ som mulig før MATE-konkurransen. Dette innebærer fellesoppgaver som montering av lys og justering av vektbalanse, og i tillegg ønsker vi å forbedre reguleringssystemet.

Reguleringssystemet vil justeres etter kvantitative tester. Tanken er å kjøre sprangresponser og påføre kraftforstyrrelser, slik at vi kan justere regulatorparametrene for å oppnå en best mulig stabilitetskontroll. I tillegg har vi i løpet av sluttfasen sett fordelen ved å kunne gjøre justeringer fra operatørgrensesnittet. Derfor ønsker vi å opprette tilleggsfunksjoner som lagrer nye regulatorparametre som sendes til mikrokontrolleren fra toppsiden. På denne måten slipper vi arbeid med demontering av elektronikkhus for hver gang vi ønsker å gjøre permanente endringer.

Dersom vi rekker å realisere de overnevnte forbedringene før MATE-konkurransen, har vi også hatt et håp om å kunne regulere stampvinkelen etter et ønsket vinkelutslag. Dette vil kunne gjøre ROV-ens rekkevidde med manipulatoren bedre, ved at den kan gripe gjenstander ovenfra eller nedenfra. Selv om dette vil være mer utfordrende enn å regulere hiv etter ønsket referanse, vil det kunne være mulig å realisere gjennom nye beregninger og mer testing.

Kapittel 10

Konklusjon

De siste fire månedene har vi jobbet med å utvikle et styrings- og reguleringssystem til ROV-en Fenris. Målet har vært at ROV-en skal fjernstyres av en pilot, og at reguleringssystemet skal assistere ved å holde farkostens orientering stabilt i vannet. Oppgavene har i hovedsak bestått av følgende:

- Gjøre utvalg av motorer og motorkontrollere som bidrar til bevegelse av ROV-en. Valget falt på T200 thrustere og Basic ESC motorkontrollere, begge produsert av Blue Robotics, samt BLDC-motorene 3508 fra Eaglepower for manipulatorstyringen. Valget av motorer og motorkontrollere har vist seg å være velegnet for alle sine oppgaver.
- Utvikle og produsere kretskort for kommunikasjon mellom motorkontrollerne, mikrokontrolleren og det resterende systemet. Kretskortet har vist seg å fungere som ønsket og gir god kommunikasjon mellom de forskjellige kretsene.
- Utvikle programvare for håndtering av pilotinstruksjoner for navigasjon og manipulatorstyring. Programvaren har fungert raskt og presist uten komplikasjoner.
- Utvikle programvare for håndtering av sensordata og regulering av frihetgradene rull, stamp og hiv. Regulatorene fungerer også som ønsket, helt uten kjente logiske eller semantiske feil.
- Matematisk modellering av ROV-ens oppførsel og dynamiske tilstand under vann. Utviklingen av modellene har gitt god forståelse av dynamikken involvert i ROV-ens oppførsel. Modellene har vist seg å være gode verktøy i beregningen av regulatorparametre.
- Realisering av PID-regulatorer og beregning av regulatorparametre basert på de matematiske modellene. De teoretiske verdiene viste seg å gi uønskede effekter på reguleringen. Det har ved testing blitt funnet parametre som forbedret oppførselen til ROV-en, men mangel på tid har ført til at parametrene foreløpig ikke har blitt tilfredsstillende tunet.

Utover å utvikle et velfungerende system, var det også fra starten et mål om å utføre prosjektet så modulært og gjenbrukelig som mulig. Fra vår oppgave er det mulig å gjenbruke både maskinvare og programvare. Motorer og motorkontrollere er beregnet å ha flere års levetid. Grensesnittet for elektronikken i elektronikkhuset er standardisert slik at det er enkelt å gjenbruke eller erstatte samtlige kretskort. Programvaren er spesifikt utviklet med en særlig modulær struktur, slik at den kan enten gjenbrukes eller benyttes som et skall for lignende fremtidige prosjekt.

De mest signifikante delene av prosjektet vi ikke har realisert innen innleveringsfristen for bacheloroppgaven, er å gjennomføre grundige kvantitative tester av både reguleringen og manipulatorstyring. Dessverre har det blitt noe knapt med tid, og eksterne forsinkelser har forårsaket at det ikke har latt seg gjøre. Det er likevel planlagt å fortsette å forbedre de utviklede systemene i tiden frem til MATE ROV Competition, slik at vi kan gi Fenris best mulig vinnermulighet.

Prosjektet i sin helhet har vært en suksess. Utviklingen av ROV-en ble fullført i stor nok grad til å utføre oppgavene nødvendig for å kvalifisere til konkurransen i USA. Selv uten optimale regulatorparametre var piloten i stand til manøvrere ROV-en godt nok til å utføre alle oppgavene godt innenfor tidsfristen på 15 minutter.

Videoen brukt for kvalifiseringen til konkurransen i USA er tilgjengelig via denne linken:
`https://www.youtube.com/watch?v=vSQ23oaLuho`

Testene av reguleringen utført i kapittel 8 er også filmet. Videoen er tilgjengelig via denne linken:
`https://www.youtube.com/watch?v=KWpERBYcOKY&t=32s&ab_channel=UiSSubsea`

Bibliografi

[1] Khan Academy. Torque. https://www.khanacademy.org/science/physics/torque-angular-momentum/torque-tutorial/a/torque, 2022. Hentet: 22.03.2022.

[2] Khan Academy. What is buoyant force? https://www.khanacademy.org/science/physics/fluids/buoyant-force-and-archimedes-principle/a/buoyant-force-and-archimedes-principle-article, 2022. Hentet: 17.03.2022.

[3] Britannica. *Hydraulic Power*. https://www.britannica.com/science/binding-energy. Hentet: 18.01.2022.

[4] MG Chemicals. 832tc. https://www.mgchemicals.com/downloads/tds/tds-832tc-2parts.pdf, 2021. Hentet: 23.02.2022, Ver. 4.0.

[5] Quy Truong Daniel Andre Kommedal Hille. *UiS Subsea Motor, fremdrift og manipulator*. Universitetet i Stavanger, 2018.

[6] Dejan. How brushless motor and esc work. https://howtomechatronics.com/how-it-works/how-brushless-motor-and-esc-work/, 2019. Hentet: 09.02.2022.

[7] Energy Education. *Electromagnetic force*. https://energyeducation.ca/encyclopedia/Electromagnetic_force. Hentet: 18.01.2022.

[8] Würth Elektronik. Wl-smcw smt mono-color chip led waterclear. https://www.we-online.com/catalog/datasheet/150060AS75000.pdf, 2019. Hentet: 17.03.2022: REV: 001.001.

[9] Thomas Smith Eide Erik Sæland Hafnor. *Utvikling og analyse av motor-og motorstyringssystem for fjernstyrt undervannsfartøy*. Universitetet i Stavanger, 2016.

[10] Didrik Efjestad Fjereide. *ELE310 - Styringsteknikk Lab 8: Motorstyring*. Universitetet i Stavanger, 2021. Hentet: 19.01.2022.

[11] Fortior. FD6288T-Q 3-PHASE BRIDGE DRIVER. https://static.qingshow.net/fortiortech/file/1597746029372.pdf. Hentet: 01.02.2022, REV_1.3.

[12] Thor I. Fossen. Marine control systems, 2002. Hentet: 07.04.2022.

[13] Yueh-Jaw Lin Francis Nickols. *Creating Precision Robots*. Elsevier Science & Technology, 2018. Hentet: 19.01.2022.

[14] GalcoTV. Dc motors explained: Advantages of brushed & brushless motors. https://www.youtube.com/watch?v=JU08mR_isaw. Hentet: 18.01.2022.

[15] GO-BGC. Float technology. https://www.go-bgc.org/floats. Hentet: 01.04.2022.

[16] Nancy Hall. Shape effects on drag. https://www.grc.nasa.gov/WWW/k-12/airplane/shaped.html, 2021. Hentet: 20.04.2022.

[17] Finn Haugen. *PID Control*. Tapir Academic Press, 2004.

[18] Finn Haugen. Model-based pid tuning with skogestad's method, 2009. Hentet: 08.04.2022.

[19] Infineon. IRFH5301PbF Power MOSFET. https://www.infineon.com/dgdl/Infineon-IRFH5301-DataSheet-v01_01-EN.pdf?fileId=5546d462533600a40153561b477b1ebe, 2015. Hentet: 01.02.2022.

[20] LearningRC. Brushless motor kv constant explained. http://learningrc.com/motor-kv/. Hentet: 17.02.2022.

[21] Edmond Baloku Vigre Markus Lægreid Haldorsen. *UiS Subsea – Motorstyrings- og reguleringssystem*. Universitetet i Stavanger, 2021.

[22] MATE. Hjemmeside mate center. https://www.marinetech.org/. Hentet: 30.01.2022.

[23] MATE. Hjemmeside mate rov competition. https://materovcompetition.org/. Hentet: 30.01.2022.

[24] MATE. *2022 Explorer Competition Manual*, 2021. Konkurransemanual fra MATE.

[25] MATE. Explorer product demonstration prop building instructions. https://files.materovcompetition.org/2022/EXPLORER_prop_building_instructions_2022.pdf, 2022. Hentet: 01.04.2022.

[26] MATE. Mission_flythrough_2022_explorer. https://vimeo.com/679543161, 2022. Hentet: 01.04.2022.

[27] MATLAB. What is gain scheduling? | control systems in practice, part 2. https://www.youtube.com/watch?v=YiUjAV1bhKs&ab_channel=MATLAB, 2018.

[28] Mbari. Float profil. https://www.mbari.org/go-bgc-release/, 2020. Hentet: 31.01.2022.

[29] Mechatronics. Mr3010 series dc axial fan. https://www.mechatronics.com/pdf/MR3010.pdf, 2021. Hentet: 14.02.2022.

[30] MOLEX. Product specification. https://www.molex.com/pdm_docs/ps/PS-51021-025-001.pdf, 2015. Hentet: 17.03.2022: EN-127(2015-11 rev.1).

[31] NXT. Tja1044 high-speed can transceiver with standby mode. https://www.nxp.com/docs/en/data-sheet/TJA1044.pdf, 2017. Hentet: 19.02.2022: Rev 6.

[32] RC-Wing. Eaglepower 3508 390kv. https://www.rc-wing.com/eaglepower-3508-390kv-4-6s-high-quality-multicopter-brushless-motor.html. Hentet: 16.02.2022.

[33] RCInfo. Brushless inrunner vs outrunner motor? https://www.radiocontrolinfo.com/brushless-inrunner-vs-outrunner-motor/. Hentet: 17.02.2022.

[34] RCLab. http://www.rclab.info/2014/01/the-basics-of-electric-power-brushless.html. Hentet: 19.01.2022.

[35] Blue Robotics. https://bluerobotics.com/store/thrusters/speed-controllers/besc30-r3/. Hentet: 10.02.2022.

[36] Blue Robotics. T200 thruster. https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-thruster-r2-rp/. Hentet: 20.01.2022.

[37] Robu. What is oneshot and multishot in esc? https://robu.in/what-is-oneshot-and-multishot-in-esc-difference-between-oneshot-and-multishot-esc-esc-calibration-protocol/, 2019. Hentet: 16.02.2022.

[38] Mellichamp Doyle Seborg Edgar. *Process Dynamics and Control 4th Edition*. John Wiley & Sons, Inc, 2017.

[39] NXP Semiconductors. 3phase sensorless bldc motor control kit with s32k144. https://www.nxp.com/docs/en/application-note/AN12435.pdf, 2020. Hentet: 14.02.2022.

[40] SiLabs. Efm8 busy bee family, efm8bb2 data sheet. https://www.silabs.com/documents/public/data-sheets/efm8bb2-datasheet.pdf, 2017. Hentet: 01.02.2022 REV_1.4.

[41] Britannica. Gordon R. Slemon. *Electric Motor*. https://www.britannica.com/technology/electric-motor. Hentet: 18.01.2022.

[42] CADENCE PCB SOLUTIONS. Termination resistors: Their function and necessity on pcbs. https://resources.pcb.cadence.com/blog/2019-termination-resistors-their-function-and-necessity-on-pcbs, 2019. Hentet: 11.05.2022.

[43] ST. Discovery kit with stm32f407vg mcu, 2020. Hentet: 17.03.2022: UM1472 - Rev 7.

[44] ST. Stm32f405xx stm32f407xx, 2020. Hentet: 17.03.2022: DS8626 - Rev 9.

[45] StackExchange. Skisse børstemotor. https://i.stack.imgur.com/2aDap.jpg. Hentet: 18.01.2022.

[46] Stepperonline. Nema 23 bipolar. https://www.omc-stepperonline.com/nema-23-bipolar-0-9deg-1-26nm-178-4oz-in-2-8a-2-5v-57x57x56mm-4-wires.html. Hentet: 16.02.2022.

[47] PCBONLINE Team. Solved - how to determine pcb trace width and current. https://www.pcbonline.com/blog/how-to-calculate-PCB-trace-width-current-impedance.html, 2021. Hentet: 25.03.2022.

[48] Morten Tengesdal. Litt om utrekning av motorpådrag for ein rov, 2015. Hentet: 19.04.2022.

[49] ThrustMe. *ThrustMe Specifications 2020*. Tilsendt pdf-dokument fra ThrustMe via e-post.

[50] Dr Ken Tindell. Can-bus rammeformat. https://kentindell.github.io/2020/01/03/canframe_py_tool/. Hentet: 06.04.2022.

[51] Jürgen Wagenbach. Can bus topology and bus termination. https://support.maxongroup.com/hc/en-us/articles/360009241840-CAN-bus-topology-and-bus-termination, 2021. Hentet: 17.03.2022.

[52] Wikipedia. Can bus. https://en.wikipedia.org/wiki/CAN_bus. Hentet: 06.04.2022.

[53] Wikipedia. Electronic speed control. https://en.wikipedia.org/wiki/Electronic_speed_control. Hentet: 09.02.2022.

[54] Wikipedia. Faraday's law of induction. https://en.wikipedia.org/wiki/Faraday%27s_law_of_induction. Hentet: 14.02.2022.

[55] Wikipedia. *Brushed DC electric motor*. https://en.wikipedia.org/wiki/Brushed_DC_electric_motor. Hentet: 18.01.2022.

[56] Wikipedia. *Brushless DC electric motor*. https://en.wikipedia.org/wiki/Brushless_DC_electric_motor. Hentet: 19.01.2022.

[57] Wikipedia. Torque. https://en.wikipedia.org/wiki/Torque. Hentet: 17.02.2022.

[58] wikipedia. Ziegler–nichols method. https://en.wikipedia.org/wiki/Ziegler-Nichols_method, 2021. Hentet: 12.05.2022.

[59] Wikipedia. Drag (physics). https://en.wikipedia.org/wiki/Drag_(physics), 2022. Hentet: 17.03.2022.

[60] Wikipedia. List of moments of inertia. https://en.wikipedia.org/wiki/List_of_moments_of_inertia, 2022. Hentet: 09.03.2022.

[61] Wikipedia. Remotely operated underwater vehicle. https://en.wikipedia.org/wiki/Remotely_operated_underwater_vehicle#Educational_outreach, 2022. Hentet: 31.01.2022.

[62] Jinghua Zhong. Pid controller tuning: A short tutorial. https://engineerscommunity.com/uploads/short-url/ueWO4XlcvlQtVgGmhqSFNkkiSzu.pdf, 2021. Hentet: 12.05.2022.

Vedlegg A

Programlisting

Gerberfilene for kretskortet kan lastes ned som .zip-fil via denne linken:
`https://github.com/OttoLjos/Styring-og-regulering/blob/main/Sensor_Reguleringskort.zip`

Programvaren for mikrokontrolleren kan lastes ned som .zip-fil via denne linken:
`https://github.com/OttoLjos/Styring-og-regulering/blob/main/Reg.zip`

Simuleringsfilene kan lastes ned som .zip-fil via denne linken:
`https://github.com/OttoLjos/Styring-og-regulering/blob/main/Simuleringsfiler.zip`

Rådataen fra testen utført på regulatorene kan lastes ned som .txt-fil via denne linken:
`https://github.com/OttoLjos/Styring-og-regulering/blob/main/reg_test_gyro.txt`

Koden for behandling og plotting av rådataen fra testen utført på regulatorene kan lastes ned som .m-fil via denne linken:
`https://github.com/OttoLjos/Styring-og-regulering/blob/main/reg_test_data.m`

Vedlegg B

Tester

B.1 Kraftbidrag fra T200

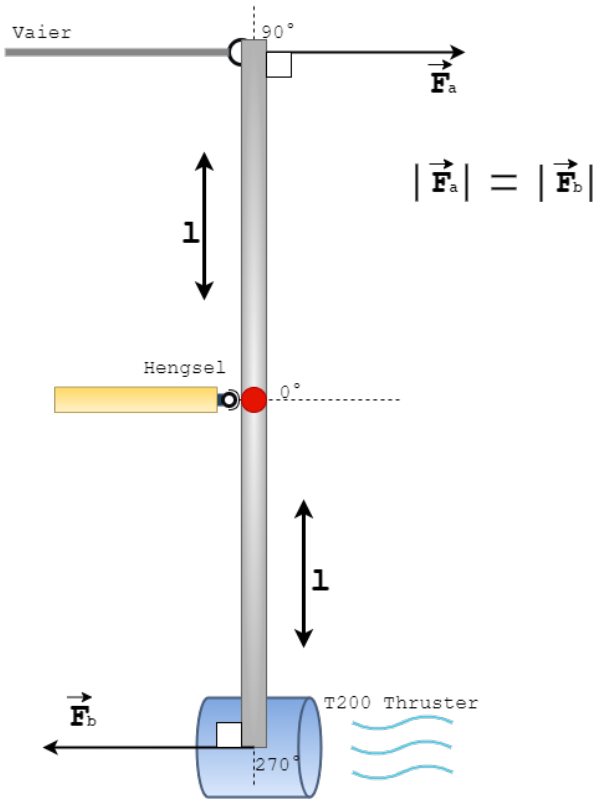
I dette kapittelet presenters det tester på thrusterne T200.

B.1.1 Hensikt

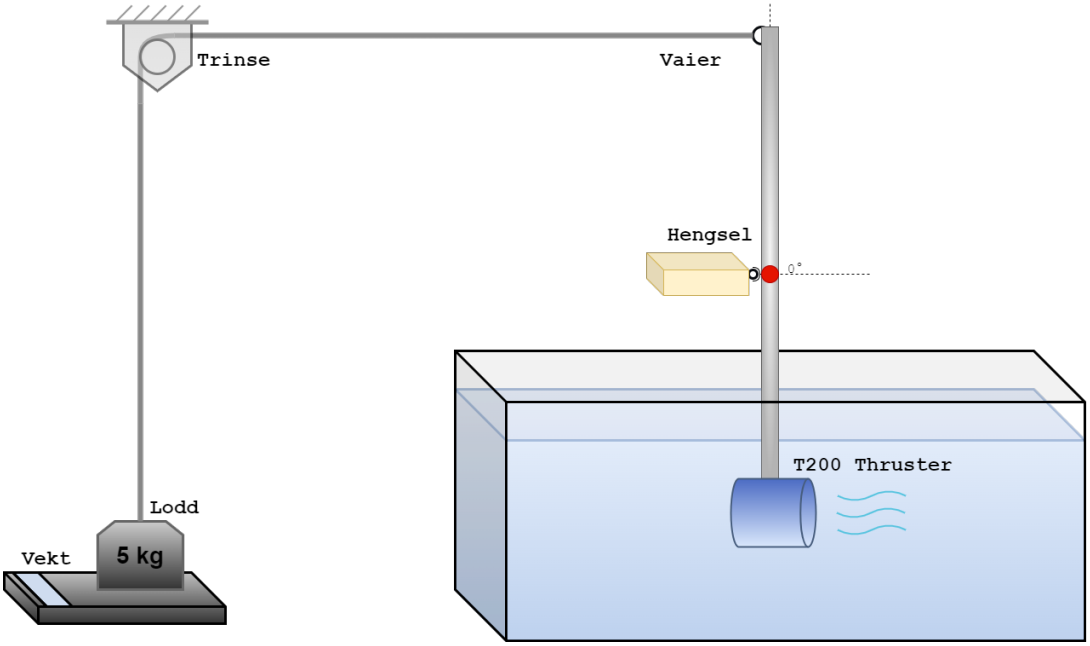
Testen skal kartlegge kraftbidrag og strømtrekk ved faste pådragsnivå. Kraftkarakteristikken skal brukes til å sammenligne kraftbidrag forover og bakover, og lage en matematisk modell for denne sammenhengen. Sammenhengen i motorkarakteristikken er helt nødvendig for å beregne riktig pådragsnivå i styringsprogrammet.

B.1.2 Metode

Måler effektiv kraft gitt av thruster i vann ved å overføre kraften med hjelp av vaier, se figur B.1. Trekket i vaieren blir overført fra thrusteren med hjelp av en symmetrisk vippepinne. Snorkraften i vaieren anses som tilnærmet lik i begge ender, slik at målt vekt i enden av vaieren tilsvarer kraften thrusteren trekker i motsatt ende. En komplett modell av testkonstruksjonen er vist i figur B.2.



Figur B.1: Modell av kraftoverføring fra thruster til vaier.



Figur B.2: Modell av testkonstruksjonen.

B.1.3 Utstyr

- Badekar
- Lodd 5 kg
- Trinse
- Vaier
- Ohaus Ranger 3000 elektronisk vekt.
- T200 Thruster fra Blue Robotics
- Basic ESC motorkontroller
- Keysight InfiniVision MSO-X 3012T
- AimTTi QPX1200SP 1200W DC Power Supply
- Arduino Uno
- Diverse byggematerialer

B.1.4 Fremgangsmåte

Sender PWM-signal fra Arduino til motorkontrolleren. Starter på 1500 μs pulsbredde, og øker med 50 μs for hvert steg. Noterer ned pulsbredde, avlest vekt og strømtrekk på kraftforsyningen. Etter maks pådrag på 1900 μs monteres thusteren motsatt retning, og prosedyren gjentas. Bruker nå motsatt retning på stegene; starter på 1500 μs og korter ned pulsbredden for hvert steg.

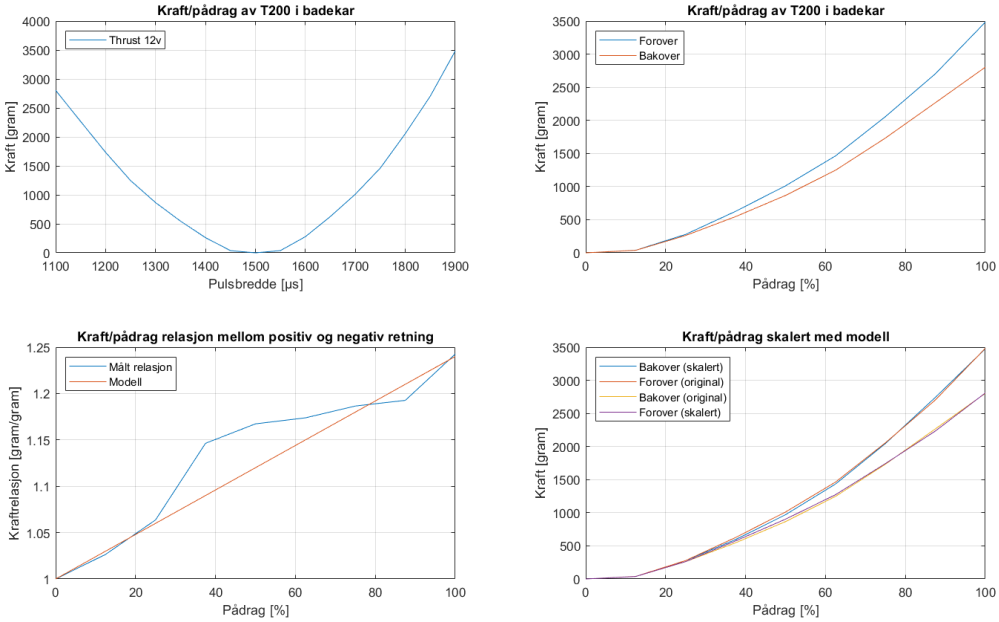
Benytter MATLAB til å plotte motorkarakteristikken og finne modell som beskriver sammenhengen mellom kraften forover og bakover, samt pådrag og kraft.

I figur B.3 vises teskonstruksjonen slik den ble satt opp på verkstedet.

Figur B.3: Den faktiske testkonstruksjonen.

B.1.5 Resultat

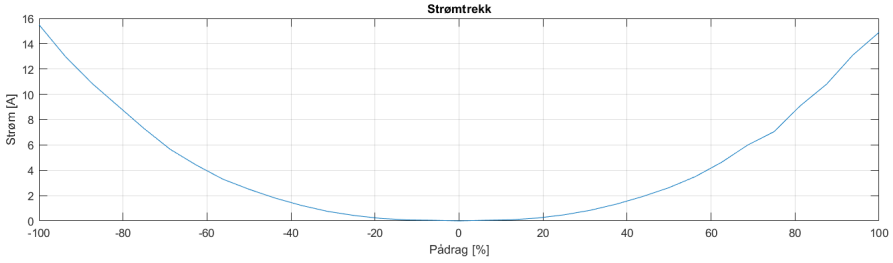
Målingene viser at kraften i positiv kjøreretning er opp til 25% høyere enn i negativ retning, ved tilsvarende pådrag, se figur B.4. Forholdet mellom kraftbidraget i hver retning er grunnlaget for konverteringsfunksjonen i ligning B.2. Denne funksjonen er benyttet i plottet nederst til høyre i figur B.4. Plottet viser grafer for standard pådragskarakteristikk, og to grafer som viser hvordan pådragsskaleringen tilpasser pådraget for motsatt karakteristikk.



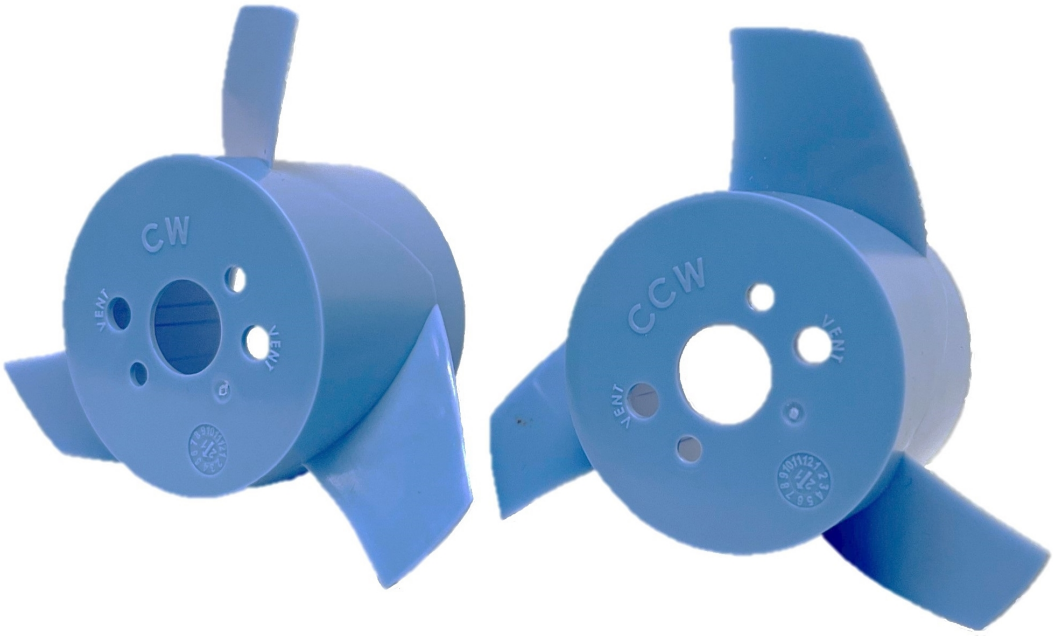
Figur B.4: Øverst venstre: Antall gram absolutt kraft ved forskjellig pulsebreddesignal. Øverst høyre: Antall gram kraft forover og bakover ved prosentvis pådrag. Nederst venstre: Forholdet mellom absolutt kraft forover og bakover, og modellen som brukes som en tilnærming. Nederst høyre: To grafer (gul og rød) viser originalt kraftbidrag, og to viser resultatet av å skalere kraftkarakteristikken med skaleringsfunksjonen (blå og lilla).

Ved å skalere kraftbidragene er det mulig å vite med relativ sikkerhet hvilket pådrag som skal til for å tilsvare kraften gitt av et pådrag i motsatt retning. En thruster montert i motsatt retning kan derfor styres som vanlig, ved å inversere karakteristikken i styringen.

Målingene av strømtrekket viste seg å være svært like i begge kjøreretninger, se figur B.5. Dette bekrefter at det er utformingen av rotorbladene som forårsaker høyere kraft i en gitt kjøreretning. Karakteristikken til rotoren gir ikke bare høyere kraft, men beholder samme strømtrekk, slik at den er mer effektiv. T200 leveres med alternativ propell, som inverserer retningen vannet blir dyttet av rotasjon, se figur B.6.



Figur B.5: Strømtrekk.



Figur B.6: Propeller for T200. Begge er mest effektiv i den samme retningen, men de dytter i motsatt retning på vannet ved samme rotasjonsretning.

Relasjon mellom kraftpådragene som skaleringsfunksjon:

$$\textbf{Nedskalering:} \quad \text{Pådrag}_F = \frac{\text{Pådrag}_B}{1 + 0.0024 \cdot \text{Pådrag}_B} \tag{B.1}$$

$$\textbf{Oppskalering:} \quad \text{Pådrag}_B = (1 + 0.0024 \cdot \text{Pådrag}_F) \cdot \text{Pådrag}_F \tag{B.2}$$

Skaleringsfunksjonen er basert på verktøyet *Curve fitting tool* i MATLAB. Ut fra måledataen som ble hentet ut, ble en lineær modell vurdert som presis nok. Modellen ble også manuelt justert for å passe bedre for det aktuelle formålet.

Konvertering fra kraft til pådrag:

$$f(x) = \frac{24.44 \cdot x^2 + 39.94 \cdot x + 0.52}{x + 0.21} \tag{B.3}$$

Hvor x og f(x) symboliserer:

- x = kraft i kg
- f(x) = prosent pådrag

MATLAB gir kvalitetsmålet $R^2 = 0.996$ av maks 1.0 for hvor godt funksjonen beskriver forholdet mellom kraft og pådrag. Denne funksjonen skal benyttes av PID-regulatorene for å konvertere beregnet pådragskraft i kg til prosent pådrag for thrusterne. Funksjonen ble derfor justert til å fjerne konstantleddet i telleren, 0.52, slik at funksjonen ikke gir ut et konstantpådrag ved 0 kg kraft.

B.1.6 Konklusjon

Målingene og resultatene gir en god indikasjon på den faktiske karakteristikken til T200 thrusteren. Det er mange forstyrrelseselementer og unøyaktigheter med testmodellen som ble benyttet. For å nevne noen av de viktigste kildene til avvik må selve konstruksjonen nevnes. For at kraften skal måles riktig, kreves det at kraftoverføringen i vaieren er nøyaktig 90° på vippepinnen, og at loddet henger helt loddrett ned på vekten. I tillegg er det flere kilder til friksjon. Den største variabelen er antageligvis likevel oppførselen til vannet i badekaret. Oppførselen til vannet endrer seg når det er i bevegelse, og ved kontinuerlig pådrag skapes det vannstrømninger som sirkulerer gjennom thrusteren. Dette fører til at den relative hastigheten mellom vannet og thrusteren endrer seg, men hvor thrusteren står i ro. Det ble observert at når vannstrømmene hadde etablert seg i badekaret ville kraften synke, og strømtrekket synke betraktelig. For å forårsake så realistiske målinger som mulig, og like vilkår for alle målinger, ble alle målingene tatt kort tid etter pådraget var satt. Det er forøvrig unøyaktigheter i målingsapparater, men disse ses som marginale sammenlignet med de ovennevnte årsakene.

Selv om tallene ikke 100% representerer reelle verdier, så gir de et realistisk bilde av thrusterens karakteristikk og trenden i effektivitet forover i forhold til bakover. Derfor vil skaleringsmodellen holde selv om de reelle kraftverdiene kan være noe høyere eller lavere.

B.2 Kretskort: Kommunikasjon

I dette teskapittelet presenteres tester på CAN-bussen samt tidsbruk for utregning av pådrag i mikrokontrolleren.

B.2.1 Hensikt

Formålet med testen er å verifisere at mikrokontrolleren mottar og sender meldinger på CAN-bussen, og dermed teste at tranceiverkretsen på kretskortet fungerer. Testen vil også vise om avbruddshåndteringen på kortet er riktig, slik at mikrokontrolleren reagerer på den mottatte dataen. Tidsbruken for utregning av pådrag basert på styremeldingene og reguleringen skal også måles.

B.2.2 Metode

Det skal verifiseres at mikrokontrolleren mottar styremeldingen med ID 70, heksadesimal: 0x46. Når mikrokontrolleren har mottatt denne meldingen, som innholder styredataen, skal pådragene regnes ut og PWM-signalene oppdateres. Dersom reguleringen er aktiv, skal også disse pådragene regnes ut med den siste mottatte sensordataen. Når pådragene er regnet ut og PWM-signalene satt, skal mikrokontrolleren sende det prosentvise pådraget for alle thrusterne over CAN-bussen slik at dette kan vises i operatørgrensesnittet. Denne pakken har ID 170, heksadesimal: 0xAA. Ved å måle tiden det tar fra styremeldingen er mottatt, til mikrokontrolleren begynner å sende ut pådragsmeldingen, kan man se hvor lang tid behandlingen av dataen tar.

B.2.3 Utstyr

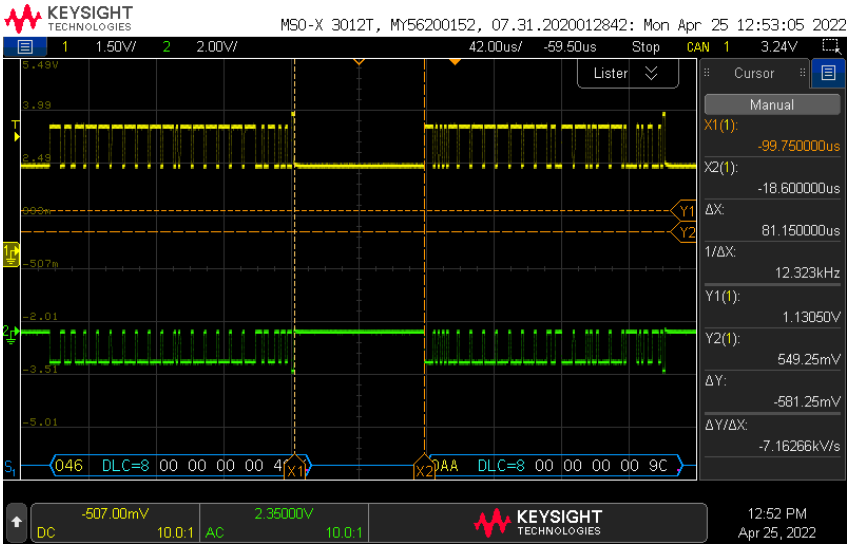
- Oscilloskop av typen Keysight InfiniVision DSO-X 3054T
- GUI laget av UiS Subsea
- Kraftforsyningskortet i Fenris

B.2.4 Fremgangsmåte

Elektronikkhuset kobles opp og spenningssettes. Oscilloskopet kobles opp til CAN-buss-linjene. Probe 1 (gul) kobles til CANH, som er den positive CAN-buss-linjen. Probe 2 (grønn) kobles til CANL, som er den negative CAN-buss-linjen. På oscilloskopet brukes en funksjon som tolker dataen sendt på CAN-bussen og viser dette i et heksadesimalt format. Ved å sette triggeren til å finne styremeldingen, pakken med ID 70, kan man ved hjelp av linjalene måle tiden mellom denne pakken og pakken med ID 170.

B.2.5 Resultat

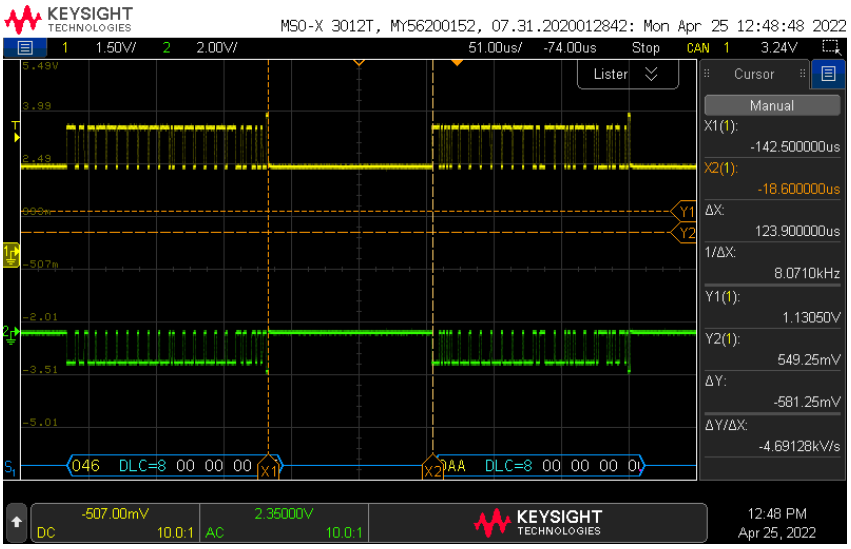
I figur B.7 vises CAN-buss-linjene når reguleringen ikke er aktiv.



Figur B.7: Pakken med ID 70 (0x46) og pakken med ID 170(0xAA) på CAN-bussen, uten aktiv regulering.

Som vist i figur B.7 blir styrepakken med ID 70(0x46) sendt på CAN-bussen. Ca. 81 μs etter at pakken er mottatt, sender mikrokontrolleren pakken med det prosentvise pådraget for alle thrusterne, ID 170 (0xAA). Det kan konkluderes med at mikrokontrolleren mottar pakken med styredata, ettersom den ikke sender ut sin pakke før styrepakken er mottatt. Styrepakkene blir sendt med en frekvens på 20 Hz, og det er derfor viktig at mikrokontrolleren blir ferdig med utregningen før en ny styrepakke kommer. Tiden på 81 μs er raskere enn det tar å sende én pakke over CAN-bussen, og mer en raskt nok til å ikke skape problemer.

I figur B.8 vises CAN-buss-linjene når reguleringen er aktiv.



Figur B.8: Pakken med ID 70 (0x46) og pakken med ID 170(0xAA) på CAN-bussen, med aktiv regulering.

Med aktiv regulering øker tiden med ca. 43 μs. Den totale tiden blir dermed ca. 124 μs. Dette tilsvarer ca 0.25% av en periode med frekvens på 20 Hz. Derfor er det heller ikke et problem med tidsbruk selv om reguleringen er aktiv.

B.2.6 Konklusjon

Testen viser at tranceiverkretsen på kretskortet virker slik at mikrokontrolleren er i stand til å både motta og sende pakker over CAN-bussen. Testen viser også at mikrokontrolleren håndterer avbruddet riktig, og sender ut riktig pakke etter at thrusterpådragene er regnet ut. Tiden denne utregninen tar er neglisjerbar i forhold til frekvensen dataen sendes med. Tester på selve utregningen av thrusterpådragene er dokumentert i testkapittel B.3.

B.3 Kretskort: Styring

I dette testkapittelet presenteres tester på behandlingen av styredata fra operatørgrensesnittet.

B.3.1 Hensikt

Formålet med testen er å vise at det blir sendt ønskede PWM-signaler til motorkontrollerene ved bestemte styresignal fra Xbox-kontrolleren. Testen reflekterer ikke ROV-ens bevegelse i vann ved gitte pådrag, men skal verifisere at utregningen av pådragssignaler i mikrokontrolleren er riktig.

B.3.2 Metode

Ved å måle PWM-signalene som sendes fra kretskortet til motorkontrolleren, kan det verifiseres at utregningene presentert i kapittel 6 blir utført riktig av mikrokontrolleren. For å kontrollere pådragsutregningene blir det brukt MATLAB-kode som utfører matematikken i de aktuelle ligningene. Her legges styredata inn manuelt, og koden regner ut pådrag i prosent, samt pulsbredden PWM-signalet bør ha.

Under vises koden for utregning av utgangssignal ved kjøring i translasjonsretning. Denne koden tilsvarer utregningene vist i kapittel 6. Utregningene tar også hensyn til skalering i forhold til motorkarakteristikk og maksimalt pådrag. For de horisontale thrusterne tilsvarer negativt pådrag den sterkeste retningen, og derfor er det disse som nedskaleres.

Kode B.1: Kode for utregning av bevegelse i translasjonsretning.

```
%% Horisontal bevegelse
clc; clear all
x=100;
y=0;
P_k = 100;
theta_k = atan2(y,x);

% Motorpaadrag
HHF = P_k*cos(-pi/4 - theta_k);
HHB = P_k*cos(-3*pi/4-theta_k);
HVB = -HHF;
HVF = -HHB;

%Skaler med hensyn paa motorkarakteristikk
if HHF < -10
    HHF = HHF/(1+abs(HHF)*0.0024);
end
if HHB < -10
    HHB = HHB/(1+abs(HHB)*0.0024);
end
if HVB < -10
    HVB = HVB/(1+abs(HVB)*0.0024);
end
if HVF < -10
    HVF = HVF/(1+abs(HVF)*0.0024);
end
```

```
27
28 %Skaler ned paadragene hvis noen av verdiene er over 100
29 if max([abs(HHF) abs(HHB) abs(HVB) abs(HVF)])>100
30     skaler = max([abs(HHF) abs(HHB) abs(HVB) abs(HVF)]);
31     HHF = 100*HHF/skaler;
32     HHB = 100*HHB/skaler;
33     HVB = 100*HVB/skaler;
34     HVF = 100*HVF/skaler;
35 end
36
37 HHF_Pulsbredde = 1500+4*HHF
38 HHB_Pulsbredde = 1500+4*HHB
39 HVB_Pulsbredde = 1500+4*HVB
40 HVF_Pulsbredde = 1500+4*HVF
```

Videre brukes egen kode for å regne utgangssignal ved giring, vist i utsnittet under. Denne koden er enklere, da alle thrustere skal ha samme pådrag. Det er lagt inn en skalering som demper pådraget med 50 % ettersom dette også er gjort på mikrokontrolleren.

Kode B.2: Kode for utregning av gir-rotasjon.

```
1  %% Gir hoyre
2  y=100;
3  demping=0.5;
4  HHF=-y*demping;
5  HHB=y*demping;
6  HVB=-y*demping;
7  HVF=y*demping;
8
9  %Skaler med hensyn paa motorkarakteristikk
10 if HHF < -10
11     HHF = HHF/(1+abs(HHF)*0.0024);
12 end
13 if HHB < -10
14     HHB = HHB/(1+abs(HHB)*0.0024);
15 end
16 if HVB < -10
17     HVB = HVB/(1+abs(HVB)*0.0024);
18 end
19 if HVF < -10
20     HVF = HVF/(1+abs(HVF)*0.0024);
21 end
22
23 HHF_Pulsbredde = 1500+4*HHF
24 HHB_Pulsbredde = 1500+4*HHB
25 HVB_Pulsbredde = 1500+4*HVB
26 HVF_Pulsbredde = 1500+4*HVF
```

Til slutt vises koden for utregning av utgangssignal ved translasjon i hiv-retning. Her er det lagt inn en demping på 60%. For de vertikale thrusterne tilsvarer positivt pådrag den sterkeste retningen, og derfor er det disse som nedskaleres.

Kode B.3: Kode for utregning av translasjon i hiv-retning.

```
1 %% Hiv ned
2 lt = 100;
3 demping=0.6;
4 VHF = lt*demping;
5 VHB=VHF;
```

```
6  VVB=VHF;
7  VVF=VHF;
8
9  if VHF > 10
10     VHF = VHF/(1+abs(VHF)*0.0024);
11 end
12 if VHB > 10
13     VHB = VHB/(1+abs(VHB)*0.0024);
14 end
15 if VVB > 10
16     VVB = VVB/(1+abs(VVB)*0.0024);
17 end
18 if VVF > 10
19     VVF = VVF/(1+abs(VVF)*0.0024);
20 end
21
22 HHF_Pulsbredde = 1500+4*HHF
23 HHB_Pulsbredde = 1500+4*HHB
24 HVB_Pulsbredde = 1500+4*HVB
25 HVF_Pulsbredde = 1500+4*HVF
```

Verdiene generert av matlab-koden er vist i delkapittel B.3.5.

B.3.3 Utstyr

- Komplett elektronikkhus
- Oscilloskop av typen *Keysight InfiniVision DSO-X 3054T*
- Måleprober

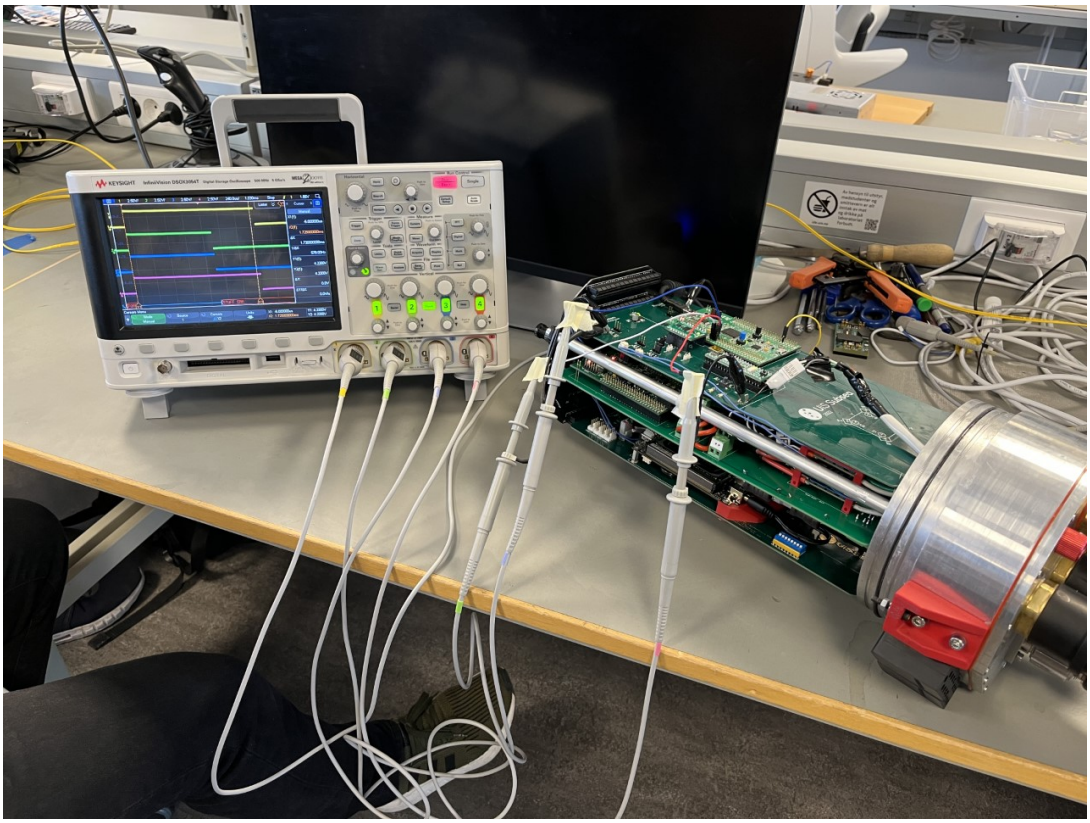
B.3.4 Fremgangsmåte

For å teste pådragene til hver mikrokontroller, kobles oscilloskopet til utgangspinnene på mikrokontrolleren som sender ut hvert sitt PWM-signal. I tabellen under vises hvilken pinne som gir PWM-signal til hvilken thruster.

Tabell B.1: Liste over PWM signal til thrustere og tilhørende pinne på mikrokontrolleren.

Thruster	Pinne på mikrokontroller
HHF	PA15
HHB	PA1
HVB	PA2
HVF	PA3
VHF	PC6
VHB	PB5
VVB	PB0
VVF	PB1
Klype	PB8

Figuren under viser hvordan måleoppsettet ble koblet opp på elektrolaben.

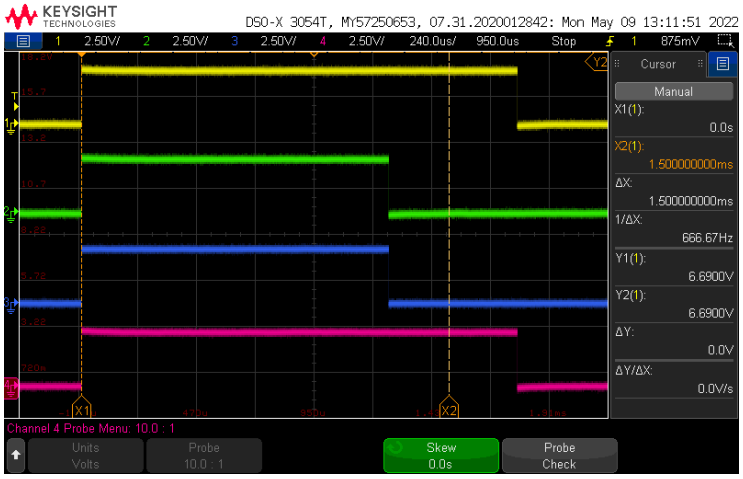


Figur B.9: Testoppsett for måling av PWM-signalene.

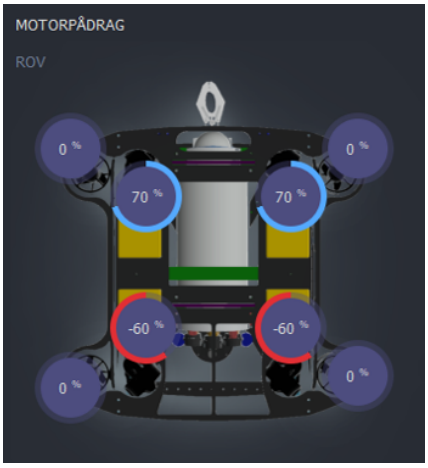
Testen gjennomføres ved å sende styredata fra operatørgrensesnittet ved bruk av Xbox-kontroller. Styredataene mottas og behandles i mikrokontrolleren på kretskortet vårt. Oscilloskopet kobles via måleprober til de aktuelle utgangspinnene på plattformkortet for å måle pulsbredden på PWM-signalet. Det prosentvise pådraget blir sendt tilbake til operatørgrensesnittet, og kan verifiseres på monitoren. Verdiene på styredataene blir notert ned, og lagt inn i MATLAB-koden for teoretisk utregning.

B.3.5 Resultat

I figur B.10 vises de målte PWM-signalene på oscilloskopet, og et utsnitt fra operatørgrensesnittet som viser prosentvis pådrag. I dette tilfellet har venstre styrestikke maksimalt utslag rett frem, som tilsvarer styredata x = 100, og y = 0. Ved kjøring rett frem, skal kraftbidraget fra alle thrustere være like stort. Ettersom de horisontale thrusterne er sterkest i negativ retning, bør de negative pådragene være litt mindre i absoluttverdi enn de positive.



(a) PWM-signal ved kjøring rett frem. HHF(Gul), HHB(Grønn), HVB(Blå), HVF(Rosa).



(b) Utsnitt fra operatørgrensesnittet ved kjøring rett frem.

Figur B.10

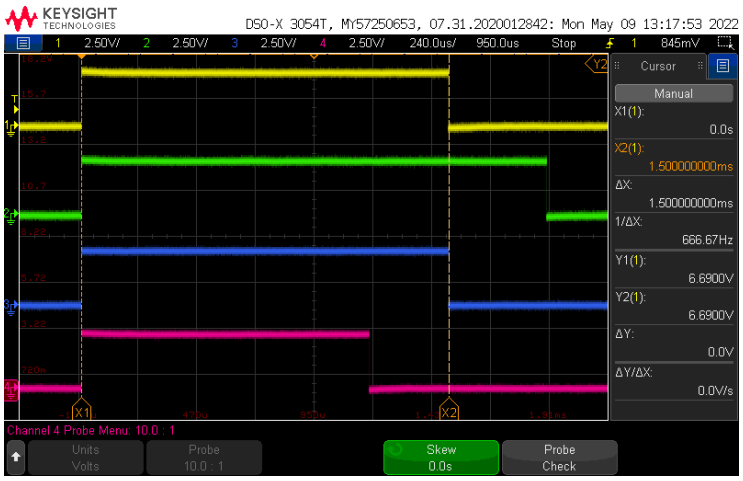
De målte pulsbreddene og prosentvise pådragene er vist sammen med de teoretiske verdiene i tabellen under.

Tabell B.2: Målte pulsbredder og prosentvise pådrag sammenlignet med teoretiske verdier.

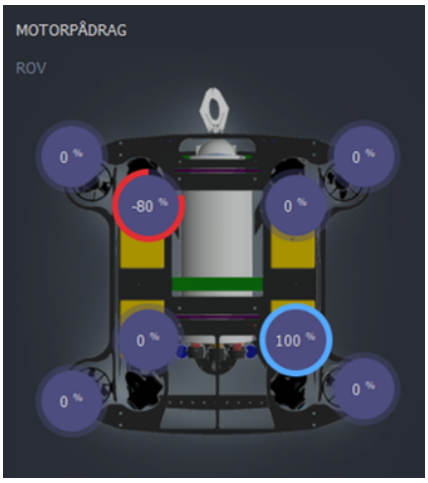
Thruster	Teoretisk pådrag [%]	Faktisk pådrag [%]	Teoretisk pulsbredde [μs]	Målt pulsbredde [μs]
HHF	70	70	1783	1780
HHB	-60	-60	1258	1256
HVB	-60	-60	1258	1256
HVF	70	70	1783	1780

Som forventet er de negative pådragene litt mindre i absoluttverdi enn de positive. Det prosentvise pådraget vist i operatørgrensesnittet tilsvarer den teoretisk utregnede verdien, men pulsbreddene avviker med noen få mikrosekunder. Dette kan komme fra unøyaktigheter i målingen på oscilloskopet, og avrundinger i utregningene på mikrokontrolleren.

I figur B.11 vises resultatene når venstre styrestikke har maksimalt utslag 45° bakover til venstre, som tilsvarer styredata x = -71, og y = -71. Her foreventes det at thruster HHF og HVB gir null bidrag ettersom disse står vinkelrett på ønsket kjøreretning.



(a) PWM-signal ved kjøring 45 grader bakover til venstre. HHF(Gul), HHB(Grønn), HVB(Blå), HVF(Rosa).



(b) Utsnitt fra operatørgrensesnittet ved kjøring 45 grader bakover til venstre.

Figur B.11

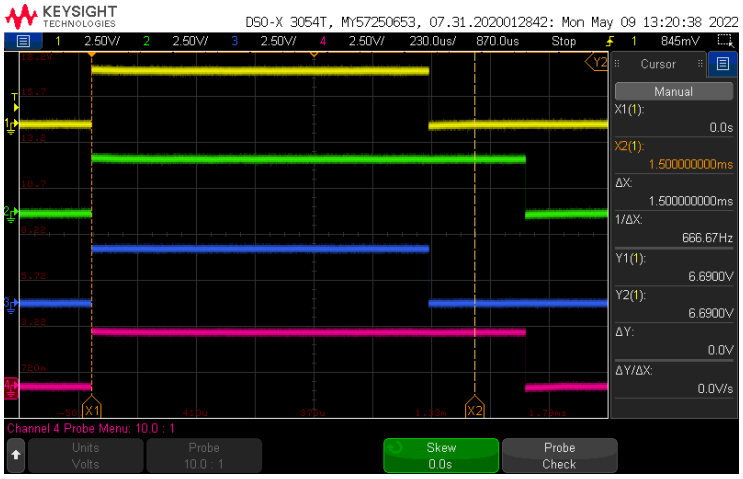
De målte pulsbreddene og prosentvise pådragene er vist sammen med de teoretiske verdiene i tabellen under.

Tabell B.3: Målte pulsbredder og prosentvise pådrag sammenlignet med teoretiske verdier.

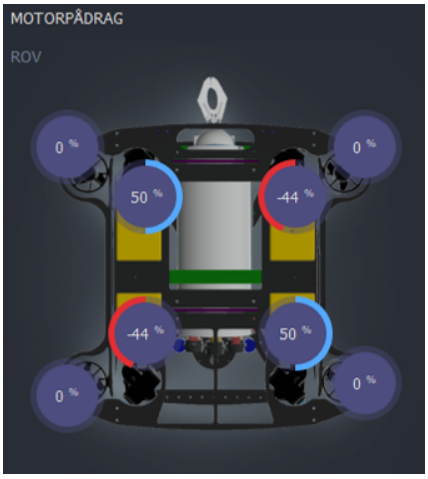
Thruster	Teoretisk pådrag [%]	Faktisk pådrag [%]	Teoretisk pulsbredde [μs]	Målt pulsbredde [μs]
HHF	0	0	1500	1500
HHB	100	100	1900	1900
HVB	0	0	1500	1500
HVF	-80	-80	1177	1175

Som forventet gir thruster HHF og HVB null bidrag. Det prosentvise pådraget vist i operatørgrensesnittet tilsvarer den teoretisk utregnede verdien. Igjen ser man små avvik i målt pulsbredde, men kun for thruster HVF.

I figur B.12 vises resultatene når høyre styrestikke har maksimalt utslag rett til høyre, som tilsvarer styredata y = 100. Her foreventes det at alle thrustere gir samme kraftbidrag i rotasjonsretning. Dette betyr at HHF og HVB skal ha negativt pådrag, og dermed litt mindre pådrag i absoluttverdi.



(a) PWM-signal ved max gir til høyre. HHF(Gul), HHB(Grønn), HVB(Blå), HVF(Rosa).



(b) Utsnitt fra operatørgrensesnittet ved max gir til høyre.

Figur B.12

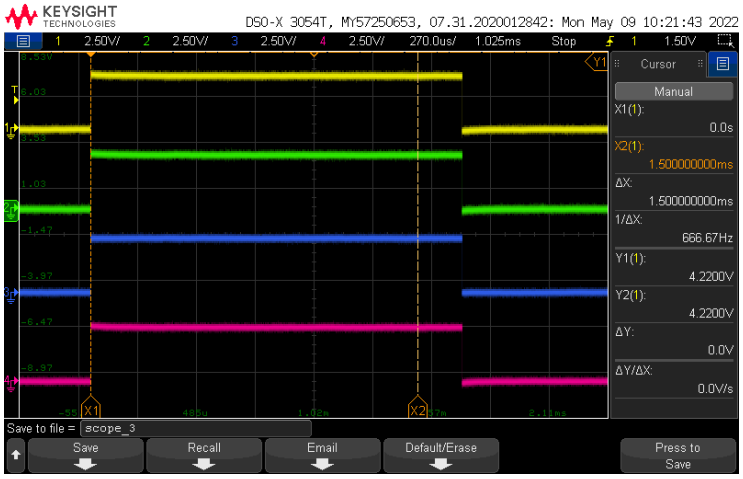
De målte pulsbreddene og prosentvise pådragene er vist sammen med de teoretiske verdiene i tabellen under.

Tabell B.4: Målte pulsbredder og prosentvise pådrag sammenlignet med teoretiske verdier.

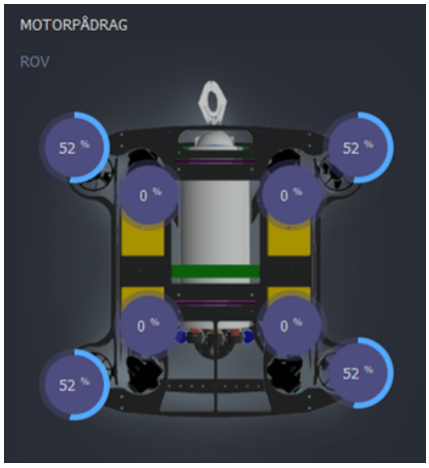
Thruster	Teoretisk pådrag [%]	Faktisk pådrag [%]	Teoretisk pulsbredde [μs]	Målt pulsbredde [μs]
HHF	-44	-44	1321	1320
HHB	50	50	1700	1700
HVB	-44	-44	1321	1320
HVF	50	50	1700	1700

Igjen er prosentvise pådrag riktige, mens pulsbreddene har små avvik. Som forventet har HHF og HVB litt mindre pådrag i absoluttverdi.

I figur B.13 vises resultatene når knappen *LT* på kontrolleren er trykket inn maksimalt, som tilsvarer styredata LT = 100. Her forventes det at alle thrusterene gir samme kraftbidrag. På grunn av dempingen skal dette pådraget være 60 %. Ettersom den sterkeste retningen for de vertikale thrusterene er ved positivt pådrag, forventes det at dette pådraget vil nedskaleres noe.



(a) PWM-signal ved kjøring nedover i hivretning. VHF(Gul), VHB(Grønn), VVB(Blå), VVF(Rosa).



(b) Utsnitt fra operatørgrensesnittet ved max hiv nedover.

Figur B.13

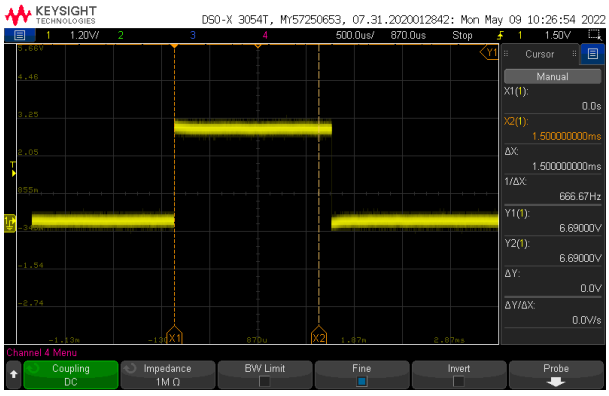
De målte pulsbreddene og prosentvise pådragene er vist sammen med de teoretiske verdiene i tabellen under.

Tabell B.5: Målte pulsbredder og prosentvise pådrag sammenlignet med teoretiske verdier.

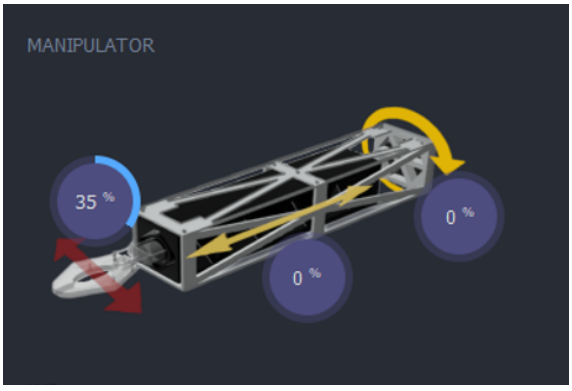
Thruster	Teoretisk pådrag [%]	Faktisk pådrag [%]	Teoretisk pulsbredde [μs]	Målt pulsbredde [μs]
VHF	52	52	1710	1708
VHB	52	52	1710	1708
VVB	52	52	1710	1708
VVF	52	52	1710	1708

Igjen er prosentvise pådrag riktige, mens pulsbreddene har små avvik. Som forventet er det prosentvise pådraget noe under 60%.

I figur B.14 vises resultatene når knappen *RB* på kontrolleren er trykket inn, som tilsvarer styredata RB = 1. Her er pådraget direkte bestemt til å være 35% når knappen er trykket inn.



(a) PWM-signal ved lukking av klype. Klype(Gul)



(b) Utsnitt fra operatørgrensesnittet ved lukking av klype.

Figur B.14

Det prosentvise pådraget er 35%. Dette er likt den teoretiske verdien. Dette skal gi en pulsbredde på 1640 μs. Den målte pulsbredden er på 1635 μs.

B.3.6 Konklusjon

Testen viser at behandlingen av styredata foregår riktig på mikrokontrolleren, og vi kan konkludere med at motorkontrollerene vil få riktige PWM-signaler. Det at PWM-signalene har riktig pulsbredde er også en verifikasjon på at konfigurasjonen av mikrokontrolleren er riktig. I tillegg bekrefter testen at kommunikasjon over CAN-bussen går som planlagt, og at mikrokontrolleren henter de ønskede verdiene fra CAN-meldingen.