Designspecifikation

Remotely Operated Underwater Vehicle

Version 1.0

Martin Lindfors
2013-10-09

Status

Granskad	Alla	2013-10-07
Godkänd	Isak Nielsen	2013-10-09

Projektidentitet

E-postlista till gruppen: tsrt10rov@googlegroups.com

Hemsida: http://www.isy.liu.se/edu/projekt/reglerteknik/2013/rov/

Beställare: Isak Nielsen, ISY, Linköpings Universitet
Telefon: +46(0) 13 282804
E-post: isak.nielsen@liu.se

Kund: Micael Derelöv, Saab Dynamics, Underwater Systems
Telefon: +46(0) 13 281165
E-post: micael.derelov@saabgroup.com

Kursansvarig: Daniel Axehill, ISY, Linköpings Universitet
Telefon: +46(0) 13 284042
E-post: daniel@isy.liu.se

Projektledare: Malte Moritz
Telefon: 070 - 515 07 66
E-post: malmo541@student.liu.se

Handledare: Jonas Linder, ISY, Linköpings Universitet
Telefon: +46(0) 13 282804
E-post: jonas.linder@liu.se

Gruppmedlemmar

Namn		**Ansvar**	**Telefon**	**E-post**
Malte Moritz	(MM)	Projektledare	070 - 515 07 66	malmo541
Kristoffer Bergman	(KB)	Testansvarig	073 - 847 31 51	kribe606
Johan Karlén	(JK)	Modellansvarig	070 - 992 32 35	johka641
Per-Erik Karlsson	(PK)	Mjukvaruansvarig	073 - 899 49 85	perka625
Martin Lindfors	(ML)	Designansvarig	070 - 264 08 01	marli984
Tobias Magnusson	(TM)	Dokumentansvarig	073 - 443 81 72	tobma696
Katarina Mollén	(KM)	Informationsansvarig	070 - 926 18 91	katmo425
Jacob Svensson	(JS)	Hårdvaruansvarig	073 - 613 58 36	jacsv832
Fredrik Söderstedt	(FS)	Kommunikationsansvarig	072 - 727 70 01	freso273

Dokumenthistorik

Version	Datum	Ändringar	Utförda av	Granskad
1.0	2013-10-09	Första versionen	Alla	Alla
0.4	2013-10-07	Fjärde utkastet	Alla	FS,ML,MM
0.3	2013-10-04	Tredje utkastet	Alla	JS,TM
0.2	2013-10-02	Andra utkastet	Alla	MM
0.1	2013-09-26	Första utkastet	Alla	MM, ML

Innehåll

1 Inledning

Detta dokument innehåller en översiktlig beskrivning av hela ROV-systemet samt dess delsystem. Plattformen som vidareutvecklas är designad och konstruerad på universitetet och består av en torpedliknande ubåt som är utrustad med styrsystem, motorer och ett antal olika sensorer. Inom såväl civila som militära tillämpningar ökar intresset och behovet av autonoma farkoster som kan utföra uppdrag till sjöss, i luften och på land utan kontakt med en operatör. För en undervattensfarkost kan ett sådant uppdrag vara att detektera och desarmera minor.

1.1 Syfte och mål

Det övergripande syftet med detta projekt är att få kunskap om och erfarenhet av utveckling av dessa undervattensfarkoster.

Projektet är en del av ett större projekt vars långsiktiga mål är att utveckla en helt autonom farkost som kan vara med i den europeiska tävlingen för autonoma undervattensfarkoster SAUC-E. Tävlingen går ut på att farkosten ska kunna utföra vissa givna uppdrag på så kort tid som möjligt. För att lyckas med detta så behöver den befintliga ROV:n utvecklas vidare till en helt autonom farkost som klarar av att orientera sig i sin omgivning. Dessutom måste den vara utrustad med hård- och mjukvara som klarar av de uppgifter som tilldelas.

Det kortsiktiga målet med projektet är att utveckla ett robust styrsystem för en väl fungerande reglering och navigering för ROV:n. Regulatorprestandan ska utvärderas efter implementation av både decentraliserad reglulator för referensföljning av vinkelhastigheter och en regulator av mer avancerad karaktär för att reglera djup och orientering. Denna utökade regulator har inom gruppen valts till en LQ-baserad regulator, och kommer hädanefter att benämnas LQ-regulatorn. Observatören av vinklar och djup ska utvärderas, och funktionaliteten och precisionen ska uppfylla kraven i givna i kravspecifikationen, Malte Moritz m.fl. [6]. Den korrekt framtagna modellen av ROV:n, som togs fram i slutet av förra projektet, ska utvärderas och parametriseras. SONAR-sensorer ska integreras och med hjälp av dessa ska det implementeras positionering för navigering. Annan hårdvara som ska integreras är nytt styr- och mätkort samt trycksensorer för djupmätning.

En detaljerad genomgång av kraven återfinns i kravspecifikationen för projektet, Malte Moritz m.fl. [6].

1.2 Användning

Systemet är tänkt att användas i undervattensmiljöer med hårda krav på precis manöverförmåga, exempelvis i tävlingen SAUC-E nämnd ovan. Systemet ska utvärderas i en bassäng som tillhandahålls av Saab Dynamics, Underwater Systems.

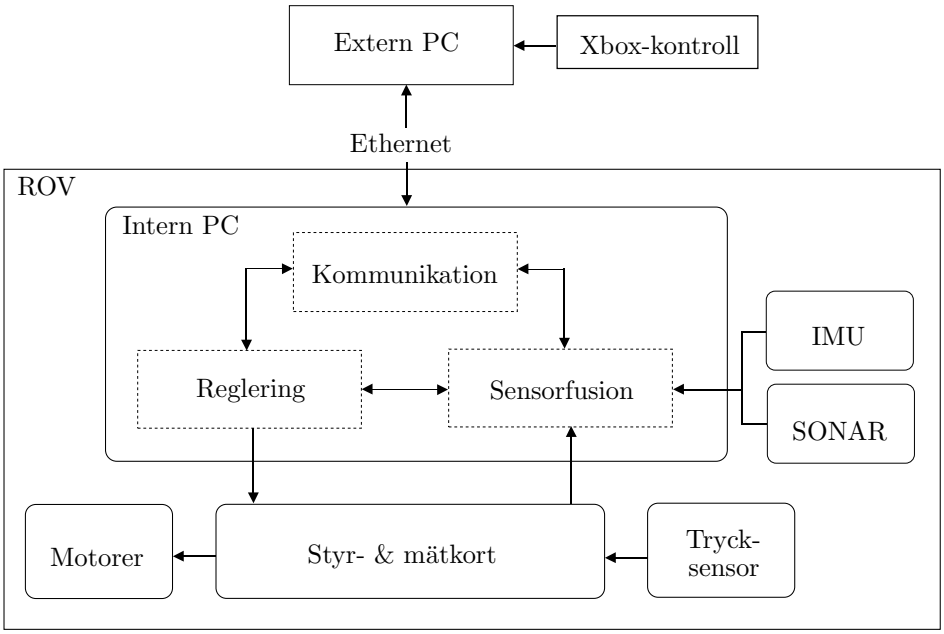
1.3 Notation

ROV	Remotely Operated Vehicle
AUV	Autonomous Underwater Vehicle
SAUC-E	Student AUV Challenge Europe
LQ	Linear Quadratic
ROS	Robot Operating System
SONAR	SOund Navigation And Ranging
IMU	Inertial Measurement Unit
EKF	Extended Kalman Filter
GUI	Graphical User Interface

2 Översikt av systemet

Detta kapitel ger en överskådlig bild av det nuvarande systemet samt över vilka förändringar som kommer att genomföras under projektets gång.

I figur 1 nedan åskådliggörs systemet i form av ett blockdiagram. En Xbox-kontroll kommunicerar med den externa PC:n, som i sin tur är kopplad till ROV:n via en Ethernetkabel. Den interna PC:n är uppdelad i tre delsystem, Kommunikation, Reglering och Sensorfusion, som alla kan kommunicera internt med varandra. Dessutom finns det ett antal olika sensorer, ett styr- och mätkort samt motorer.



Figur 1: En översikt av systemet. Streckade block motsvarar avgränsningar i mjukvara, heldragna rundade block motsvarar avgränsningar i ROV:ns inbyggda hårdvara medan heldragna kantiga block motsvarar avgränsningen mellan ROV och extern PC.

2.1 Modell

Den modell som kommer att användas för att modellera systemet är framtagen med hjälp av Fossen [3] och sedan vidareutvecklad av projektgruppen. Den modell som användes i föregående projekt kommer att frångås, då den visade sig vara felaktig. Parametrar kommer att skattas under projektets gång. Mer information om modellen finns under kapitel 3.

2.2 Hårdvara

Hårdvaran på ROV:n utgörs av en intern PC, en mikrokontroller med tillhörande Arduinokort, fem styrkort, en trycksensor, en IMU samt en läckagesensor. ROV:n är också utrustad med fem thruster-propellrar. En thruster är riktad bakåt, två nedåt och två åt sidan. Under projektets gång kommer en framåtriktad SONAR att integreras. Utöver det finns ett antal batterier, två lampor samt en kamera. Mer information om hårdvaran på ROV:n finns under kapitel 4.

2.3 Intern PC

På den interna PC:n finns mjukvara som hanterar reglering, navigering och kommunikation. Den interna PC:n på ROV:n körs i grund och botten med ett Linuxbaserat operativsystem. Den interna PC:n är uppdelad i tre delsystem:

- Reglering
- Sensorfusion
- Kommunikation

Kommunikationen mellan delsystemen går via operativsystemet ROS (Robot Operating System) som ligger ovanpå det Linuxbaserade operativsystemet. Mer information om den interna PC:ns mjukvara hittas i kapitel 5.

2.4 Extern PC

En extern PC, bestående av en laptop med operativsystemet Linux, finns ansluten till ROV:n via en Ethernetkabel. Den externa PC:n övervakar läckagesensorn och meddelar ROV:n om läckage indikeras. I PC:ns GUI kan användaren avläsa ROV:ns position, batterinivåer. Dessutom kan användaren också välja driftläge och även stoppa och föra upp ROV:n till ytan i en nödsituation. Mer om GUI:t och den externa PC:n kan läsas i kapitel 9.

Enligt krav från Malte Moritz m.fl. [6], skall systemet kunna köras i tre driftlägen.

- Manuellt läge
- Decentraliserad regulator
- LQ-regulator

Användaren av systemet kan styra ROV:n genom antingen den externa PC:ns GUI, eller med hjälp av en Xbox-kontroll som går att ansluta till den externa PC:n.

När det manuella läget är igång ska varje motor styras av varsin styrsignal med hjälp av antingen Xbox-kontrollen eller den externa PC:ns GUI, utan att någon regulator är inkopplad.

När ROV:n är i det läge då den decentraliserade regulatorn används, ska vinkelhastigheterna (yaw och pitch) och linjära hastigheter (upp/ned, höger/vänster, bak/fram) kunna styras med hjälp av Xbox-kontrollen.

När LQ-regulatorn används så ställd en önskad orientering in med hjälp av antingen GUI:t på den externa PC:n eller med hjälp av Xbox-kontrollen.

3 Modell

Den modell som kommer att användas i projektet är framtagen från grunden enligt Fossen [3]. Detta innebär att den "nya" modellen framtagen i Patricia Sundin m.fl. [5] till viss del frångås. Modellen kommer innehålla en i sammanhanget vedertagen notation för modellparametrarna för att förenkla såväl förståelsen för modellen som möjligheten att sätta sig in i modellen för framtida projekt.

Rörelseekvationerna härleds från Coriolis kraftekvation i det ROV-fasta koordinatsystemet, sedan läggs de bidragande krafterna till, se Fossen [3]. Dessa ekvationer beskriver hastigheterna i det kroppsfasta koordinatsystemet för ROV:n. Vinkelaccelerationerna härleds från Coriolis momentekvation tecknad i det ROV-fasta koordinatsystemet. Dessa ekvationer beskriver då vinkelaccelerationerna för ROV:en i det kroppsfasta koordinatsystemet.

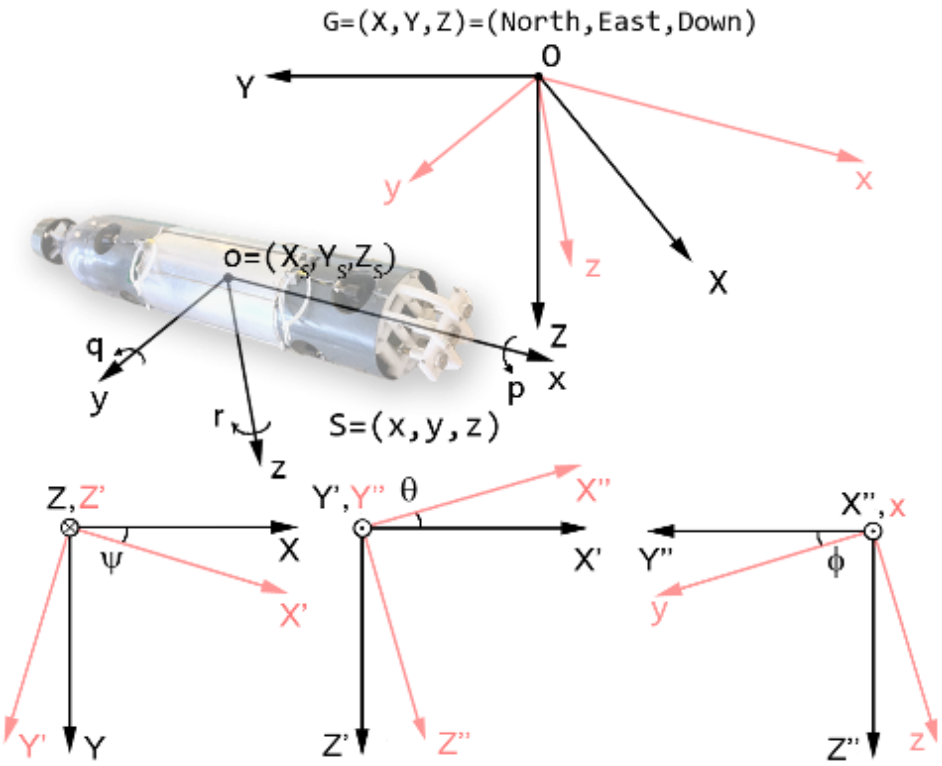
För att beskriva ROV:ns hastigheter i det jordfasta koordinatsystemet ges en ny orientering genom tre rotationer. Eftersom Eulervinklar används kommer singularitet kunna inträffa då pitch-vinkeln $\theta = \pm 90°$. Därför kommer man i sensorfusions-delsystemet att använda en alternativ modell med kvaternioner $\boldsymbol{q} = (q_0, q_1, q_2, q_3)$. I delsystem Reglering kommer dock Eulervinklar att användas, eftersom en av vinklarna inte ska regleras.

3.1 Koordinatsystem

För modellering av ROV:n införs två koordinatsystem, ett ROV-fixt $(x, y, z)^T$ samt ett jord-fixt $(X, Y, Z)^T$. Det sistnämda betraktas som ett inertialsystem där Newtons rörelseekvationer gäller. Koordinatsystemet har sitt origo vid vattenytan och är ett så kallat North- East- Downsystem med X-axeln pekandes norr, Y-axeln pekandes österut och Z-axeln pekandes nedåt. Det ROV-fixa koordinatsystemet har x-axeln pekandes rakt fram längs ROV:ns kropp, y-axeln pekandes åt styrbord och z-axeln nedåt. Origo placeras i ROV:ns masscentrum, se figur 2.

Transformationen från jord-fixa accelerationer till ROV-fixa sker med rotationsmatrisen $\boldsymbol{R}$ som definieras av Eulervinklarna, se ekvation (3.14). Matrisen härleds från tre rotationer. Först tecknas ett koordinatsystem $(X', Y', Z')^T$ genom en rotation ψ kring den jord-fixa Z-axeln. Därefter införs ännu ett nytt koordinatsystem $(X'', Y'', Z'')^T$ genom en rotation θ kring Y'-axeln. Slutligen erhålls det ROV-fixa koordinatsystemet utifrån en rotation ϕ runt X''-axeln.

För att uttrycka de kroppsfixa vinkelhastigheterna i Eulervinklarnas derivator används matrisen $\boldsymbol{T}$ som definieras enligt Eulervinklarna, se ekvation (3.16).



Figur 2: Illustrering av det jord-fixa samt ROV-fixa koordinatsystemet samt rotationen mellan dessa.

De variabler som används för att beskriva ROV:ns position, orientering samt motorspänning beskrivs i tabell 1.

Tabell 1: Beskrivning av variabler

n, e, d	ROV:ns globala koordinater i X-, Y- och Z-led.
ϕ, θ, ψ	Eulervinklar för roll, pitch och yaw.
q_0, q_1, q_2, q_3	Kvaternioner som beskriver transformationen mellan det lokala och det globala koordinatsystemet.
u, v, w	Hastigheter i det lokala koordinatsystemet längs x-, y-, och z-axlarna.
p, q, r	Rotationshastigheter i det lokala koordinatsystemet runt x-, y- och z-axlarna.
$u_T, u_{yf}, u_{yr}, u_{zf}, u_{zr}$	Den pålagda spänningen på respektive motor.

3.2 Förenklingar

För att få en hanterbar modell förenklas den fysikaliska modellen. Symmetri antas för xy-, xz- och yz-planet. Självklart är inte ROV:n helt symmetrisk kring dessa plan men med detta antagande kommer tröghetsmatrisen att bli diagonal. Eftersom tröghetsmomenten påverkar ekvationerna för vinkelaccelerationerna blir inte beräkningarna helt korrekta med detta antagande, men då vinkelhastigheterna mäts erhålls en korrigering av felet.

För att förenkla beräkningen av dämpningen från de hydrodynamiska effekterna antas rörelserna för ROV:n vara frikopplade på grund av ROV:ns relativt låga hastigheter. Detta antas också att dämpningen högst beror kvadratiskt på hastigheter.

Om förenklingarna i ett senare skede visar sig vara för approximativa kommer en mer komplex modell tas fram, t.ex. kan antagandet om symmetri visa sig behöva ändras.

3.3 Parameterskattning

För att beskriva systemmodellen behövs ett antal parametrar. De definieras i tabell 2.

Tabell 2: Beskrivning av parametrar

K_p, M_q, N_r	Linjära motståndskoefficienter för rotation i vatten.
$K_{p\|p\|}, M_{q\|q\|}, N_{r\|r\|}$	Kvadratiska motståndskoefficienter för rotation i vatten.
m	ROV:ns massa.
$X_{\dot{u}}, Y_{\dot{v}}, Z_{\dot{w}}$	Tillagd massa i x-, y- och z-led på grund av translation i vatten.
I_x, I_y, I_z	Tröghetsmoment runt x-, y- och z-axlarna.
$K_{\dot{p}}, M_{\dot{q}}, N_{\dot{r}}$	Ökat tröghetsmoment runt x-, y- och z-axlarna på grund av rotation i vatten.
z_B	Avstånd längs z-axeln från masscentrum till flytkraftens angreppspunkt.
$x_{yf}, x_{yr}, x_{zf}, x_{zr}$	Avstånd längs x-axeln från masscentrum till respektive motor.
X_u, Y_v, Z_w	Linjära motståndskoefficienter för translation i vatten.
$X_{u\|u\|}, Y_{v\|v\|}, Z_{w\|w\|}$	Kvadratiska motståndskoefficienter för translation i vatten.
$C_T, C_{M_T}, C_{yr}, C_{yf}, C_{zr}, C_{zf}$	Motorkoefficienter.
V	ROV:ns undanträngningsvolym.
ρ	Vattnets densitet.
g	Tyngdacceleration.

I tabellen nedan ges en första idé till framtagningen av parametrarna. Parametrarna kommer huvudsakligen bestämmas med hjälp av CAD-program, System Identification ToolBox i Matlab samt fysikaliska tester.

Motorkoefficienterna skattas med hjälp av ett fysikaliskt test där thrustrarnas krafter mäts var och en för sig med dynamometer. Insignalen varieras sedan för att få fram koefficienterna.

Den tillagda massan i $\boldsymbol{M}_A$-matrisen, se ekvation (3.2) kan beräknas genom att approximera ROV:n som en ellipsoid, se Fossen [2, sid 41].

Då det finns en viktriktig CAD-modell av ROV:n kan ett antal parametrar tas fram i CATIA, se tabell 3. Resterande parametrar tas fram genom datainsamling för ROV:n då den utför olika rörelser i vattnet. För att bestämma motorparametrar på ett alternativt sätt accelereras ROV:n. För skattning av dämpningskoefficienter hålls istället en viss hastighet eller vinkelhastighet för ROV:n, sedan stängs styrsignalerna till motorerna av för att låta ROV:n stanna. Den insamlade datan behandlas sedan i Matlab. En `idnlgrey`-modell används för att köra `pem`-funktionen på datasekvensen, på så sätt skattas parametrarna genom prediktionsfelsminimeringsmetoden. Se tabell 3.

Tabell 3: Metoder för parameterskattning

K_p, M_q, N_r	Grey-box.
$K_{p\|p\|}, M_{q\|q\|}, N_{r\|r\|}$	Grey-box.
m	Våg.
$X_{\dot{u}}, Y_{\dot{v}}, Z_{\dot{w}}$	Ellipsoid-approximation.
I_x, I_y, I_z	CAD.
$K_{\dot{p}}, M_{\dot{q}}, N_{\dot{r}}$	Ellipsoid-approximation.
z_B	CAD.
$x_{yf}, x_{yr}, x_{zf}, x_{zr}$	CAD.
X_u, Y_v, Z_w	Grey-box.
$X_{u\|u\|}, Y_{v\|v\|}, Z_{w\|w\|}$	Grey-box.
$C_T, C_{M_T}, C_{yr}, C_{yf}, C_{zr}, C_{zf}$	Dynamometer eller grey-box.
V	CAD.
ρ	Antas vara konstant för ett nominellt värde på vattentemperatur.
g	Känd.

3.4 Framtagning av modell

Med notationen $\boldsymbol{\eta} = (n, e, d, \phi, \theta, \psi)^T$ alternativt $\boldsymbol{\eta} = (n, e, d, q_0, q_1, q_2, q_3)^T$ för position och orientering samt $\boldsymbol{\nu} = (u, v, w, p, q, r)^T$ för de kroppsfixa hastigheterna kan ROV:ns dynamik beskrivas i det kroppsfixa koordinatsystemet enligt

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} \tag{3.1}$$

där

$\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A$ = tröghetsmatris för en stel kropp samt tillagd massa
$\boldsymbol{C}(\boldsymbol{\nu}) = \boldsymbol{C}_{RB}(\boldsymbol{\nu}) + \boldsymbol{C}_A(\boldsymbol{\nu})$ = matris beskrivande Corioliseffekt för en stelkropp samt tillagd massa
$\boldsymbol{D}(\boldsymbol{\nu})$ = hydrodynamisk dämpningsmatris
$\boldsymbol{g}(\boldsymbol{\eta})$ = vektor med tyngdkraftens inverkan på ROV:en
$\boldsymbol{\tau}$ = kraft- och momentvektor som beskriver motorernas inverkan på ROV:n

Om symmetri antas kring de tre planen som de kroppsfixa axlarna spänner upp fås enligt Fossen [3, sid 122,173]

$$\begin{aligned}\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A &= \text{diag}[m, m, m, I_x, I_y, I_z] + \text{diag}[X_{\dot{u}}, Y_{\dot{v}}, Z_{\dot{w}}, K_{\dot{p}}, M_{\dot{q}}, N_{\dot{r}}] \\ &= \text{diag}[m + X_{\dot{u}}, m + Y_{\dot{v}}, m + Z_{\dot{w}}, I_x + K_{\dot{p}}, I_y + M_{\dot{q}}, I_z + N_{\dot{r}}]\end{aligned} \tag{3.2}$$

Med omskrivningarna $\boldsymbol{v} = (u, v, w)$ och $\boldsymbol{\omega} = (p, q, r)$ samt med definitionen $\boldsymbol{S}(a\boldsymbol{A})\boldsymbol{B} = a\boldsymbol{A} \times \boldsymbol{B}$ kan Corioliseffekten $\boldsymbol{C}_{RB}(\boldsymbol{\nu})\boldsymbol{\nu}$ skrivas (enligt Fossen [3, sid 49])

$$\boldsymbol{C}_{RB}(\boldsymbol{\nu})\boldsymbol{\nu} = \begin{bmatrix} m\boldsymbol{S}(\boldsymbol{\omega}) & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & -\boldsymbol{S}(\boldsymbol{I}\boldsymbol{\omega}) \end{bmatrix} \begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} m\boldsymbol{\omega} \times \boldsymbol{v} \\ -(\boldsymbol{I}\boldsymbol{\omega}) \times \boldsymbol{\omega} \end{bmatrix} \tag{3.3}$$

Den tillagda massans inverkan på Corioliseffekten kan förenklas enligt Fossen [3, sid 122]

$$C_A = \begin{bmatrix} 0 & 0 & 0 & 0 & -Z_{\dot{w}}w & Y_{\dot{v}}v \\ 0 & 0 & 0 & Z_{\dot{w}}w & 0 & -X_{\dot{u}}u \\ 0 & 0 & 0 & -Y_{\dot{v}}v & X_{\dot{u}}u & 0 \\ 0 & -Z_{\dot{w}}w & Y_{\dot{v}}v & 0 & -N_{\dot{r}}r & M_{\dot{q}}q \\ Z_{\dot{w}}w & 0 & -X_{\dot{u}}u & N_{\dot{r}}r & 0 & -K_{\dot{p}}p \\ -Y_{\dot{v}}v & X_{\dot{u}}u & 0 & -M_{\dot{q}}q & K_{\dot{p}}p & 0 \end{bmatrix} \tag{3.4}$$

Med antagandet att ROV:n har tre symmetriplan, har frikopplad rörelse samt att termer av högre ordning än två kan försummas kan den hydrodynamiska dämpningsmatrisen enligt Fossen [2, sid 43] skrivas

$$\begin{aligned}\boldsymbol{D}(\boldsymbol{\nu}) = &-diag[X_u, Y_v, Z_w, K_p, M_q, N_r]- \\ &-diag[X_{u|u|}|u|, Y_{v|v|}|v|, Z_{w|w|}|w|, K_{p|p|}|p|, M_{q|q|}|q|, N_{r|r|}|r|]\end{aligned} \tag{3.5}$$

ROV:n är byggd för att vara stabil, och ska återgå till jämviktsläge. Flytpunkten är alltså placerad rakt ovanför tyngdpunkten, i det kroppsfixa koordinatsystemet befinner sig flytpunkten ovanför tyngdpunkten längs z-axeln enligt Bernhard & Johansson [1]. Därmed kan tyngdkraftens inverkan uttryckas

$$\boldsymbol{g}(\boldsymbol{\eta}) = \begin{bmatrix} (W-B)s\theta \\ -(W-B)c\theta s\phi \\ -(W-B)c\theta c\phi \\ -z_b B c\theta s\phi \\ -z_b B s\theta \\ 0 \end{bmatrix} = \begin{bmatrix} g(m-\rho V)s\theta \\ -g(m-\rho V)c\theta s\phi \\ -g(m-\rho V)c\theta c\phi \\ -z_b \rho g c\theta s\phi \\ -z_b \rho g s\theta \\ 0 \end{bmatrix} \tag{3.6}$$

Krafterna och momenten från motorerna modelleras som fluidkrafter där kraften är proportionell mot kvadraten av thrustrarnas rotationshastighet. Rotationshastigheten antas vara proportionell mot motorns styrsignal.

Thrustrarna är placerade så att deras kraftbidrag verkar ortogonalt mot x-axeln vilket resulterar i moment kring y- och z-axeln. Sidothrustrarna är placerade på en axel som går genom ROV:ns geometriska centrum och är parallell med x-axeln. Eftersom masscentrumet befinner sig under ROV:ens geometriska centrum bidrar sidothrustrarna även med ett vridande moment i roll-led. Detta moment antas vara försummbart.

$$\boldsymbol{\tau} = \begin{bmatrix} C_T u_T |u_T| \\ C_{yr} u_{yr} |u_{yr}| + C_{yf} u_{yf} |u_{yf}| \\ C_{zr} u_{zr} |u_{zr}| + C_{zf} u_{zf} |u_{zf}| \\ C_{M_T} u_T |u_T| \\ C_{zr} x_{zr} u_{zr} |u_{zr}| - C_{zf} x_{zf} u_{zf} |u_{zf}| \\ C_{yr} x_{yr} u_{yr} |u_{yr}| - C_{yf} x_{yf} u_{yf} |u_{yf}| \end{bmatrix} \tag{3.7}$$

Ekvation (3.1) kan nu skrivas på formen

$$\dot{u} = \frac{C_T}{m+X_{\dot{u}}}u_T|u_T| + \frac{X_u + X_{u|u|}|u|}{m+X_{\dot{u}}}u - \frac{g(m-\rho V)}{m+X_{\dot{u}}}\sin\theta - \frac{m-Z_{\dot{w}}}{m+X_{\dot{u}}}wq + \frac{m-Y_{\dot{v}}}{m+X_{\dot{u}}}vr \tag{3.8}$$

$$\dot{v} = \frac{C_{yr}}{m+Y_{\dot{v}}}u_{yr}|u_{yr}| + \frac{C_{yf}}{m+Y_{\dot{v}}}u_{yf}|u_{yf}| + \frac{Y_v + Y_{v|v|}|v|}{m+Y_{\dot{v}}}v + \frac{g(m-\rho V)}{m+Y_{\dot{v}}}\cos\theta\sin\phi + \\ + \frac{m-Z_{\dot{w}}}{m+Y_{\dot{v}}}wp - \frac{m-X_{\dot{u}}}{m+Y_{\dot{v}}}ur \tag{3.9}$$

$$\dot{w} = \frac{C_{zr}}{m+Z_{\dot{w}}}u_{zr}|u_{zr}| + \frac{C_{zf}}{m+Z_{\dot{w}}}u_{zf}|u_{zf}| + \frac{Z_w + Z_{w|w|}|w|}{m+Z_{\dot{w}}}w + \frac{g(m-\rho V)}{m+Z_{\dot{w}}}\cos\theta\cos\phi + \\ + \frac{m-X_{\dot{u}}}{m+Z_{\dot{w}}}uq - \frac{m-Y_{\dot{v}}}{m+Z_{\dot{w}}}vp \tag{3.10}$$

$$\dot{p} = \frac{C_{M_T}}{I_x+K_{\dot{p}}}u_T|u_T| + \frac{K_p + K_{p|p|}|p|}{I_x+K_{\dot{p}}}p + \frac{z_B\rho V g}{I_x+K_{\dot{p}}}\cos\theta\sin\phi - \\ - \frac{(I_z-N_{\dot{r}})-(I_y-M_{\dot{q}})}{I_x+K_{\dot{p}}}qr - \frac{Y_{\dot{v}}-Z_{\dot{w}}}{I_x+K_{\dot{p}}}vw \tag{3.11}$$

$$\dot{q} = \frac{C_{zr}x_{zr}}{I_y+M_{\dot{q}}}u_{zr}|u_{zr}| - \frac{C_{zf}x_{zf}}{I_y+M_{\dot{q}}}u_{zf}|u_{zf}| + \frac{M_q + M_{q|q|}|q|}{I_y+M_{\dot{q}}}q + \frac{z_B\rho V g}{I_y+M_{\dot{q}}}\sin\theta - \\ - \frac{(I_x-K_{\dot{p}})-(I_z-N_{\dot{r}})}{I_y+M_{\dot{q}}}pr - \frac{Z_{\dot{w}}-X_{\dot{u}}}{I_y+M_{\dot{q}}}uw \tag{3.12}$$

$$\dot{r} = \frac{C_{yr}x_{yr}}{I_z+N_{\dot{r}}}u_{yr}|u_{yr}| - \frac{C_{yf}x_{yf}}{I_z+N_{\dot{r}}}u_{yf}|u_{yf}| + \frac{N_r + N_{r|r|}|r|}{I_z+N_{\dot{r}}}r - \\ - \frac{(I_y-M_{\dot{q}})-(I_x-K_{\dot{p}})}{I_z+N_{\dot{r}}}pq - \frac{X_{\dot{u}}-Y_{\dot{v}}}{I_z+N_{\dot{r}}}uv \tag{3.13}$$

Högerledet i ekvationerna betecknas f_b så att $\dot{\boldsymbol{\nu}} = f_b(\boldsymbol{\nu}, \boldsymbol{\eta})$. För att uttrycka de kroppsfixa accelerationerna i globala koordinater tecknas rotationsmatrisen $\boldsymbol{R}$ som funktion av antingen Eulervinklar eller kvaternioner. De kroppsfixa vinkelhastigheterna transformeras till derivator av Eulervinklar alternativt kvaternioner genom matrisen $\boldsymbol{T}$.

$$\boldsymbol{R} = \boldsymbol{R}(\phi,\theta,\psi) = \begin{bmatrix} c\theta c\psi & c\theta s\psi & -s\phi \\ s\phi s\theta c\psi - c\phi s\psi & s\phi s\theta s\psi + c\phi c\psi & s\phi c\theta \\ c\phi s\theta c\psi + s\phi s\psi & c\phi s\theta s\psi - s\phi c\psi & c\phi c\theta \end{bmatrix} \tag{3.14}$$

$$\boldsymbol{R} = \boldsymbol{R}(q_0,q_1,q_2,q_3) = \begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2q_1q_2-2q_0q_3 & 2q_0q_2+2q_1q_3 \\ 2q_0q_3+2q_1q_2 & q_0^2-q_1^2+q_2^2-q_3^2 & -2q_0q_1+2q_2q_3 \\ -2q_0q_2+2q_1q_3 & 2q_2q_3+2q_0q_1 & q_0^2-q_1^2-q_2^2+q_3^2 \end{bmatrix} \tag{3.15}$$

$$\boldsymbol{T} = \boldsymbol{T}(\phi,\theta,\psi) = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & \frac{s\phi}{c\theta} & \frac{c\phi}{c\theta} \end{bmatrix} \tag{3.16}$$

$$\boldsymbol{T} = \boldsymbol{T}(q_0,q_1,q_2,q_3) = -\frac{1}{2}\begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ -q_3 & q_0 & q_1 \\ q_2 & -q_1 & q_0 \end{bmatrix} \tag{3.17}$$

Sambandet mellan lokala och globala hastigheter blir således $\dot{\boldsymbol{\eta}} = \boldsymbol{J}\boldsymbol{\nu}$ där $\boldsymbol{J}$ är transformationsmatrisen

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{T} \end{bmatrix} \tag{3.18}$$

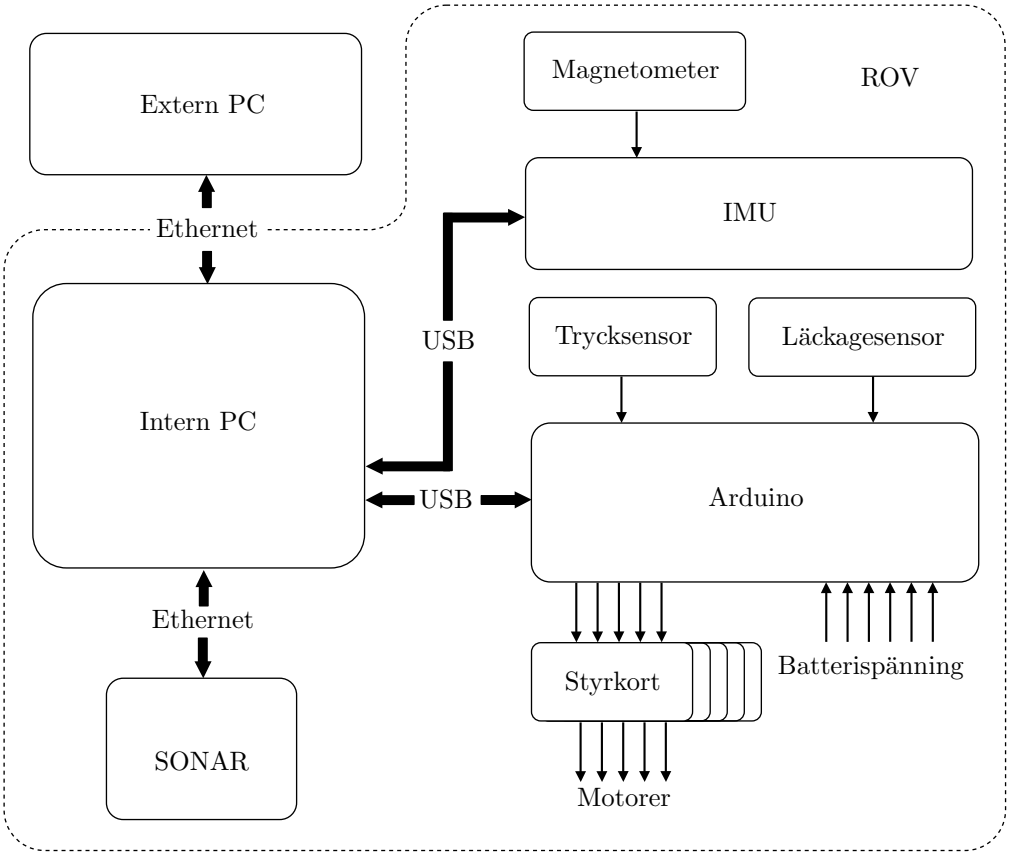
Där $\boldsymbol{0}$ är nollmatriser av lämplig dimension.

Med $\boldsymbol{x} = (\boldsymbol{\eta}^T, \boldsymbol{\nu}^T)^T$ samt $\boldsymbol{u} = (u_T, u_{yf}, u_{yr}, u_{zf}, u_{zr})$ skrivs den fullständiga modellen

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \boldsymbol{J}\boldsymbol{\nu} \\ f_b \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}\boldsymbol{v} \\ \boldsymbol{T}\boldsymbol{\omega} \\ f_b(\boldsymbol{\nu}, \boldsymbol{\eta}) \end{bmatrix} \tag{3.19}$$

4 Hårdvara

Hårdvaran på ROV:n utgörs av en PC (benämnd den interna PC:n), en mikrokontroller med tillhörande Arduinokort, fem styrkort, en trycksensor, en IMU, en SONAR samt en läckagesensor. Utöver det finns en antal batterier, thrusterpropellrar, lampor samt en kamera. I detta kapitel kommer Arduinokortet, sensorerna, propellrarna och PC:n att beskrivas.



Figur 3: En översikt av hur de olika hårdvarukomponenterna är sammankopplade

4.1 Arduinokortet

Styr- och mätkortet som sitter i ROV:en är av typen Arduino Mega 2560. Kortet bygger på mikrokontrollern ATmega2560 som har utrustats med bland annat USB-anslutning, strömanslutning, 16MHz

oscillator och en återställningsknapp. Arduino-kortet har analoga ingångar och utgångar som används för att mäta sensor-data och för att skicka styrsignaler till motorernas drivkort. Arduinon kommunicerar seriellt med den interna PC:n med hjälp av ROS. Arduinon programmeras med en modifierad variant av C/C++ som bland annat har tillagda funktionsprototyper. Arduinon är sedan tidigare projekt programmerad för att styra motorer och läsa sensorvärden, dock kommer dess kod behöva förbättras eftersom tidigare projekt upplevt problem med en överbelastad databuss.

4.2 Trycksensor

Trycksensorn som sitter på ROV:en är från serien PX2 tillverkad av Honeywell. Den är placerad på den delen av ROV:ns tryckkärl (höljet som innesluter elektroniken) som är riktad mot aktern. Utsignalen från trycksensorn är en spänning som är proportionell mot trycket och kan anta värden mellan 10 % och 80 % av matningsspänningen. Lägsta utsignalen svarar mot ett tryck på 0 Pa medan maximala utsignalen svarar mot 689 kPa (100 psi). Med en matningsspänning på 5 V och antagandet att sambandet är linjärt erhålls

$$U = k \cdot p + c \tag{4.1}$$

där U är uppmätt spänning och p är aktuellt tryck. Trycket är summan av lufttrycket och vattentrycket. k och c är konstanter som enkelt kan beräknas med vetskapen om att 0 Pa svarar mot 0,5 V och 689 kPa mot 4 V. Det slutgiltiga sambandet mellan utspänning och tryck ges av

$$U = 5.1 \cdot 10^{-6} \cdot p + 0.5, \quad 0 \leq p \leq 689 \cdot 10^3 \tag{4.2}$$

4.2.1 Upplösning i Arduinon

Trycksensorn är kopplad till en A/D-omvandlare på Arduinokortet. A/D-omvandlaren har en upplösning på 10 bitar, det vill säga 1024 steg, vilket svarar mot en spänning mellan 0 V och en valbar referensspänning V_{ref}. Med detta givet blir upplösningen i A/D-omvandlingen

$$\frac{V_{ref}}{1024} \tag{4.3}$$

Med referensspänningen satt till 2,56 V (vilket Arduinon har inbyggt stöd för) innebär det att upplösningen blir 2,50 mV. Instoppat i ekvation (4.2) motsvarar det 492 Pa. För att få en uppfattning om det är tillräckligt bra kan motsvarande djupupplösning beräknas via sambandet

$$p = \rho h g \tag{4.4}$$

där ρ är vattnets densitet (998 kg/m^3), h är vattendjupet och g är tyngdaccelerationen (9,82 m/s^2). Med siffror instoppade erhålls en upplösning på 5,0 cm som kommer att användas i delsystem Sensorfusion för att skatta ett djup för farkosten. Enligt kravspecifikationen (Malte Moritz m.fl. [6, krav 52]) ska farkosten kunna skatta sitt djup med en noggrannhet på ±5 cm.

Det maximala trycket som kan mätas med den inställningen blir med ekvation (4.2) 404 kPa. Med antagandet att lufttrycket är 101 kPa (SMHI [7]) ges det maximala djupet som kan mätas av ekvation (4.4), det vill säga 41 m.

4.2.2 Omvandling till tryck

Med vetskapen om vilken upplösning A/D-omvandlaren har, dess största och minsta möjliga spänningar samt ekvation (4.2), kan vi mappa resultatet från A/D-omvandlaren till tryck. I tabellen nedan visas

ytterligheterna i mappningen. Den övre gränsen på 2,56 V sätts av referensspänningen medan den undre gränsen på 0,5 V är den minimala spänningen trycksensorn kan ge ifrån sig.

Binärt tal	Omvandlad spänning	Uppmätt tryck	Motsvarande vattendjup
200	0,50 V	0 kPa	- (vakum)
406	1,02 V	101kPa	0 m
1023	2,56 V	404 kPa	41 m

Det linjära, kvantiserade sambandet mellan binärt tal och tryck blir således

$$p = \begin{cases} 0 & b_2 < 200 \\ 491 \cdot b_2 - 98.2 \cdot 10^3 & 200 \leq b_2 \leq 1023 \\ 404 \cdot 10^3 & b_2 > 1023 \end{cases} \tag{4.5}$$

där b_2 är det binära talet.

4.3 IMU och magnetometer

IMU:n (Internal Measurement Unit) med tillhörande magnetometer används för att mäta vinkelhastigheter, vinkelacceleration och magnetfält. Under projektets gång kommer två olika IMU:s användas och integreras. IMU:n som använts i tidigare projekt (Xsens MTi) har blivit störd av magnetiska fält som uppstått i ROV:n. Den största störningskällan har enligt Bernhard & Johansson [1] varit ROV:ns motorer. Dock har ingen utredning gjort om även spolar, hårddiskar, kablar och reläer bidragit till störningar. Därför kommer mätningar göras för att hitta en lämplig placering som både tar hänsyn till att minimera störningar samtidigt som det är fördelaktigt ur mätsynvinkel att ha en placering nära ROV:ns masscentrum. De två lösningar som kommer att användas beskrivs nedan.

4.3.1 Xsens MTi

Projektet har till en början tillgång till en Xsens MTi. Fördelen med denna IMU är att den är beprövad och gränssnittet mot den interna PC:n är USB vilket gör den relativt lätt att komma igång med. Nackdelen är att IMU:n är en begränsad resurs och inte kommer finnas tillgänglig för framtida projekt eftersom den ska användas i andra applikationer. Dock kommer denna att använda i inledningen av projektet så att skattningen av tillstånd kan göras för regleringen.

IMU:n har egna algoritmer för att med hjälp av sensorfusion beräkna tillstånden, men i detta projekt används enbart rådatan. Den kalibrerade rådatan skickas till interna PC:n enligt följande:

accX	accY	accZ	gyrX	gyrY	gyrZ	magX	magY	magZ	TS

där varje värde är ett 32-bitars flyttal. Accelerationen mäts i m/s^2, vinkelhastigheten mäts i rad/s och magnetfältet är en arbiträr enhet normalizerad mot jordens magnetfält. TS är ett värde för tidpunkten.

4.3.2 APM 2.6 med extern kompass

Den andra lösningen bygger på ett separat kort ArduPilot APM 2.6 som har en inbyggd krets med både accelerometer och gyrometer (MPU-6000). Till kortet kopplas en extern magnetometer (HMC5883L) som kommunicerar med sladd via I2C och som därför kan placeras i valfritt ställe i ROV:en. Placeringen kan därför väljas till en plats där de magnetiska störningarna är tillräckligt små.

Varje sensor mäter värden i x-, y- och z-led vilket gör att kortet totalt har nio frihetsgrader. APM:en kommunicerar med den interna PC:n via USB och kommer likt Xsens MTi strömma rå sensordata till PC:n. Kortet har även 8 st ingångar och 8 st utgångar som går att använda på liknande sätt som Arduinons. Detta projekt kommer dock begränsas till att använda APM:ens interna sensorer samt magnetometer för att mäta sensorvärden.

4.4 SONAR Delta-T 837

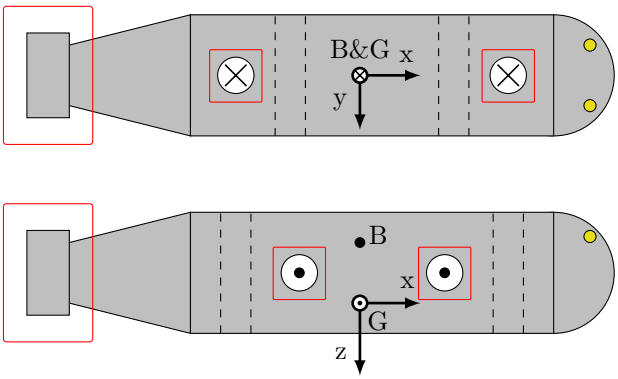
SONAR:en som ska användas i detta projekt och tillhandahålls av Saab Dynamics, Underwater Systems, är Delta-T modell 837 av tillverkare Imagenex. SONAR:en ska sitta längst fram i ROV:en och läsa av omgivningen i ROV:ens primära rörelseriktning.För varje ping som SONAR:en skickar ut mäter den hur lång tid det tar för att få svar för varje av de 120 diskreta vinklarna. Det avsökta området täcker 120° spridning i yawled och 0,75° i pitchled och skickas 20 gånger per sekund. Av mätdatan skapas därefter ett 83P UDP datagram som skickas vidare till den interna PC:n. För mer information om datagrammet och mjukvaran som behövs för att tolka data, se 5.2. SONAR:en ansluts med en specialgjord kabel som har åtta ledare och en vattentät kontakt. På andra sidan ansluts fyra av ledarna till en Ethernetkabel och två av ledarna till en spänningskälla. SONAR:en drivs med 22-50 VDC som bör tas från en rippelfri spänningskälla. Därför kan det vara aktuellt att antingen ha ett separat batteripaket ämnat för SONAR:en eller att seriekoppla ett filter.

4.5 Läckagesensor

Läckagesensorn är utvecklad i tidigare projekt kring ROV:n. Den är placerad under batterierna och ska detektera om vatten har trängt in i farkosten. Sensorn består av ett motstånd vilket spänningen mäts över. Då inget läckage förekommer är denna spänning 4.8 V. När vatten kommer i kontakt med sensorn jordas motståndet via vattnet och spänningen över motståndet sjunker. Spänningen över motståndet är utsignalen från sensorn och ett spänningsfall kan enkelt upptäckas av Arduinon. En mer detaljerad beskrivning av läckagesensorn finns att tillgå i Bernhard & Johansson [1, sid 18].

4.6 Motorer

På ROV:n finns fem thrusterpropellrar. En schematisk skiss över dess placering kan ses i figur 4. Propellern längst bak styr surge-rörelsen (u) medan två är placerade i vertikalled och styr sway-rörelsen (v) tillika yaw-rotationen (r) och två är placerade i horisontalled och styr heave-rörelsen (w) tillika pitch-rotationen (q). Detta är möjligt tack vare att propellrarna genererar samma kraft oavsett vilket håll de körs åt enligt Bernhard & Johansson [1, sid 15]. ROV:n har alltså fem styrbara frihetsgrader.



Figur 4: Motorernas placering på ROV:n, markerade med en röd rektangel. B anger var flytkraften påverkar ROV:n och G visar var ROV:ns tyngdpunkt är. Observera att figuren inte är skalenlig.

Varje motor är kopplat till ett styrkort. Styrkortets uppgift är att översätta styrsignalen från Arduinokortet till en pulsbreddsmodulerad signal. De olika motorerna kräver olika mycket ström, varför styrkorten anpassas individuellt till varje motor.

4.7 Intern PC

Den interna PC:n innehåller

- Processor Intel D525MW
- Minne Kingston 2x2 GB DDR3 PC3-8500 1066 MHz
- Hårddisk OCZ 60 GB Vertex II E-series

Mjukvaran på interna PC:n beskrivs närmare i kapitel 5.

4.8 Övrig hårdvara

På ROV:n finns även följande hårdvara:

- Webbkamera längst fram på ROV:n som ansluts via USB till den interna PC:n.
- Lampor längst fram som styrs via Arduinon och lyser upp för kameran.
- Totalt 6 st batteripaket som ger ca 24 volt och spänningssätter hela ROV:n.

5 Mjukvara

På den interna PC:n finns den mjukvara som hanterar reglering, navigering och kommunikation. I grund och botten körs ett Linuxsystem på PC:n. Ovanpå det körs delsystemen i operativsystemet ROS (Robot Operating System). ROS och gränssnitten mellan de olika delsystemen beskrivs närmare i kapitel 5.1 medan de olika delsystemen beskrivs i kapitel 6, 7 och 8. Mjukvaran för SONAR:en körs separat från ROS genom gränssnittet Wine vilket beskrivs mer i kapitel 5.2.

5.1 ROS

ROS är inte ett operativsystem i vardaglig mening eftersom det kräver en underliggande Linuxdistribution för att köras. ROS tillhandahåller dock många av de funktioner som kan förväntas av ett operativsystem, såsom

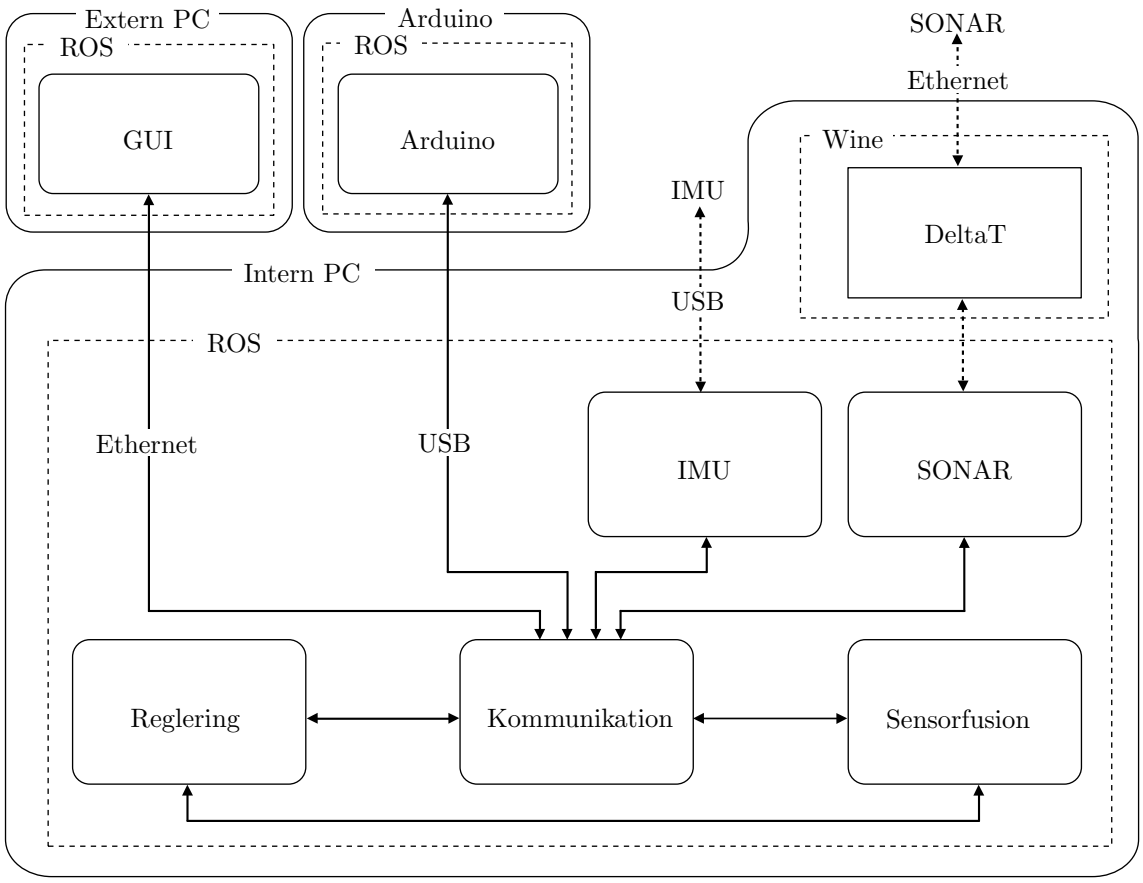
- Abstraktion av hårdvara
- Möjligheten att köra flera processer samtidigt
- Kommunikation mellan processer i form av meddelandepassning

ROS gör det alltså möjligt att bygga upp ett modulärt system samt att kunna abstrahera bort skillnaden mellan en ren mjukvarukomponent och extern hårdvara. Processerna, som kallas för noder, kommunicerar sinsemellan med meddelanden, så kallade topics. En mer utförlig beskrivning av ROS finns att tillgå på ROS.org [8].

I figur 5 visas de olika noderna och hur de kommunicerar. Delsystem Reglering, Sensorfusion och Kommunikation motsvaras av varsin nod och är ren mjukvara.

I ROS kommer en nod, se 5.1.1, vid namn GUI att implementeras. Denna nod kommer att finnas i den externa PC:n och abstraherar bort kommunikationen över Ethernet-länken. När en nod publicerar något som GUI prenumererar på, kommer ROS att se till att informationen når externa PC:ns GUI. Arduinon fungerar på liknande sätt.

Noden IMU fungerar idag på ett annat sätt än de andra noderna. Den läser seriell data på USB-porten som IMU-hårdvaran är ansluten till och konverterar detta till ett format som kan publiceras. Detta är relativt enkelt eftersom det finns stöd för den specifika IMU:n i ROS. När IMU:n sedan byts ut kommer den troligtvis att implementeras på samma sätt som GUI-noden och Arduino-noden.



Figur 5: En schematisk bild av mjukvaran. Heldragna, rundade boxar representerar olika hårdvarukomponenter. Streckade, kantiga boxar representerar de olika miljöerna som mjukvarorna körs i. Heldragna, rundade boxar motsvarar olika noder i ROS. Streckade, kantiga boxar visar övriga processer. Heldragna pilar motsvarar utbyte av topics mellan noderna. Streckade pilar motsvarar övrig kommunikation.

5.1.1 Noder

Med ROS:s terminologi kallas en process för en nod. De tre delsystemen, Reglering, Sensorfusion och Kommunikation, är implementerade som varsin nod. Hårdvarukomponenterna Arduino, IMU, SONAR samt extern PC ska även de vara noder. Delsystem Kommunikation kommer att fungera som en brygga för kommunikationen mellan den interna och den externa PC:n via TCP/IP samt den interna PC:n och Arduinon via seriell överföring. Genom denna brygga kan Arduinon och externa PC:n kommunicera med varandra. Alla noder är listade i tabell 4.

Tabell 4: Alla noder

Namn	**Motsvarande del i systemet**	**Kapitel**
control	Reglersystem	6
sensor	Sensorfusion	7
com	Kommunikation	8
GUI	Grafiskt användargränssnitt	9
arduino	Arduino	4.2.1
IMU	IMU	4.3
SONAR	SONAR	5.2

5.1.2 Topics

I ROS kommunicerar noder med hjälp av så kallade topics. Varje topic är kopplat till ett meddelande, så kallade messages. Ett meddelande består av en datastruktur som innehåller olika typade (heltal, flyttal) fält. När en nod vill delge information skickar den ut ett meddelanden till en topic. En topic fungerar som en identifierare för det specifika meddelandet. De noder som är intresserade av att få den informationen prenumererar på den topicen.

En topic fungerar alltså som en kommunikationsbuss. Alla noder kan vara publicerare (publishers) och alla kan vara prenumeranter (subscribers) för den aktuella noden. De behöver heller inte känna till varandras existens. Nuvarande topics kan avläsas i Appendix A. Topics för sensordata kommer att läggas till under projektets gång.

5.2 DeltaT

DeltaT är ett program som körs utanför ROS på den interna PC:n. Programmet ansluter till SONAR:en via TCP och signalbehandlar data från SONAR:en till formatet ".83P", som består av en avståndsmätning för varje lob. Den behandlade SONAR-datan skickas sedan vidare till ROS via UDP. Mjukvaran är utvecklad för Windows men den interna PC:n använder Ubuntu som operativsystem så programmet körs genom Wine som är en miljö som "emulerar" Windows.

5.2.1 Externa kontrollkommandon

För att kunna styra SONAR:en och ställa in inställningar från ROV:n (utan GUI:t i DeltaT.exe) krävs att *ExternalControlEnableTCP* sätts till 1 i konfigurationsfilen DeltaT.ini. DeltaT.exe kommer då agera server och kunna kommunicera via TCP. Det går då bra att ansluta med en extern kontrollenhet såsom ROS i den interna PC:n för att få ut data. För att ansluta skickar man ett externt kontrollkommando på 255 byte enligt en förbestämd mall som innehåller information om antalet strålar som ska användas, öppningsvinklar etc. DeltaT.exe svarar då med ett första 83P datagram.

5.2.2 83P Datagram

Med SONAR:en finns möjlighet att få ut råa mätdata (83B) som är vinklar med intensitetsarrayer eller en skattad avstånd av föremål för varje lob (83P). Då ROV:en ska navigera i en bassäng med tydliga kanter används den senare varianten. Datan man får ut är en vektor som beskriver avståndet för varje stråle till närmsta objekt. Ett objekt bakom ett annat objekt hamnar i skuggan och kan därför inte upptäckas. Efter varje ping så kommer DeltaT.exe returnera ett paket på totalt 495 bytes enligt följande:

Byte #	Beskrivning
0 - 255	**Rubrik** som innehåller information från SONAR:ens interna tillstånd vid mättillfället. Några exempel är information om överföringsversion, tid, datum, antal strålar, startvinkel, mätavstånd, frekvens, specificerad ljudhastighet i vatten, pingnummer, fördröjning etc.
256 - 495	**Avståndsprofil** för aktuellt ping. Vardera av de 120 strålarnas avstånd beskrivs med två bytes. Exempel för det returnerade avståndet för den första strålen (byte 256-257): Avstånd = (Byte 256 << 8 | Byte 257) * Avståndsupplösning / 1000 meter

Avståndsprofilen kan även kompletteras med ytterligare två bytes per ping som beskriver varje pings intensitet. Den totala paketstorleken blir då 735 bytes.

6 Delsystem Reglering

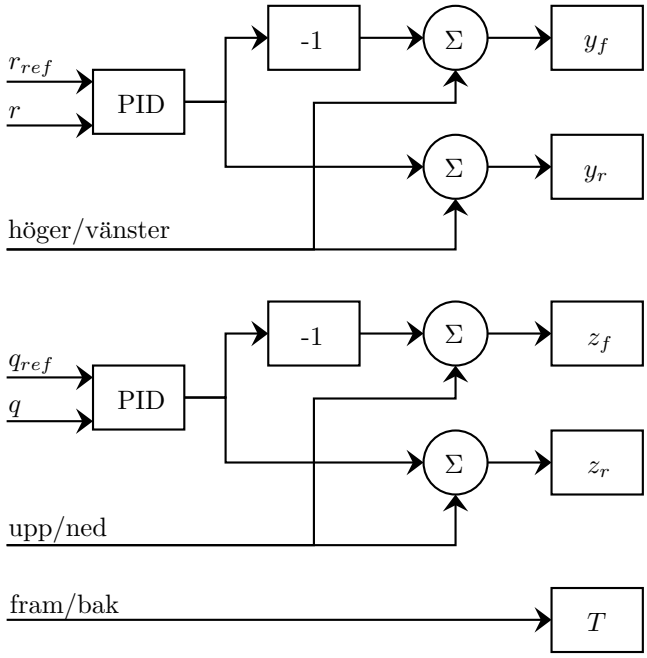
Två olika typer av regulatorer ska användas för att styra ROV:n, en decentraliserad regulator och en LQ-regulator. Den decentraliserade regulatorn ska utnyttjas för att styra ROV:ns vinkelhastigheter i yaw- och pitchled oberoende av varandra. Eftersom den nuvarande motoruppsättningen inte möjliggör direkt kontroll av ROV:n i roll-led så kommer det inte att vara möjligt att reglera i den ledden. LQ-regulatorn ska användas då ROV:n ska inställa sig på ett önskat djup med en önskad orientering (yaw och pitch), och sedan bibehålla dess position. Vilket reglersystem som används väljs med den externa PC:n.

6.1 Decentraliserad regulator

Decentraliserad reglering innebär att man reglerar ett system med flera in- och utsignaler genom att låta en insignal styra en utsignal. Resultatet blir flera oberoende loopar med varsin tillhörande regulator.

I ROV:ns system ska den decentraliserade regulatorn frikoppla vinkelhastigheterna i yaw- och pitchled i det kroppsfixa koordinatsystemet från varandra. Det ska också vara möjligt att styra ROV:n framåt/båkåt, upp/ned och höger/vänster i det kroppsfixa koordinatsystemet. Detta innebär att en referenssignal endast ska påverkar en utsignal. På så sätt ska vinkelhastigheterna och de linjära hastigheterna kunna styras oberoende av varandra.

I figur 6 beskrivs hur regulatorn kommer att fungera. Regulatorns insignaler är referenssignaler från externa PC:n samt skattade tillstånd från delsystem Sensorfusion. Dessa kommer sedan att skickas till varsin PID-regulator, vilka bestämmer styrsignaler för de fem thrustrarna. För att styra vinkelhastigheterna kommer de lodrätt riktade motorerna respektive de vågrätt riktade motorerna fram och bak köras i motsatt riktning för att motverka drift i positionen. För att köra bakåt/framåt kommer den bakre motorn att användas. För att köra upp/ned respektive höger/vänster kommer de lodrätt och vågrätt riktade motorerna att användas. En skalfaktor för styrsignalerna kommer också att behövas, eftersom motorerna har olika avstånd till rotationscentrum. Därmed kommer ingen modell av systemet behöva användas vid framtagningen av den decentraliserade regulatorn.



Figur 6: En översikt av den decentraliserade regulatorn, med insignaler och utsignaler.

6.2 LQ-regulator

Linjärkvadratisk reglering (LQ-reglering) innebär att systemet regleras med en avvägning mellan snabbhet och små styrsignaler. LQ-regulatorn innehåller en tillståndsåterkoppling från den tillståndsskattning som beskrivs under delsystem Sensorfusion. Referenssignalerna till regulatorn kommer vara orientering (yaw och pitch) samt djup (Z). Regulatorn kommer också ha referenssignaler för horisontell position (X, Y) för att förhindra drift i position hos ROV:n. Utsignalerna från regulatorn kommer vara styrsignaler till de fem thrustrarna.

6.2.1 Linjärisering av ROV:n

För att använda en LQ-regulator måste en linjär tillståndsbeskrivning av systemet tas fram. I detta projekt kommer exakt linjärisering att användas. Exakt linjärisering fungerar på så sätt att en styrsignal till systemet väljs så att de olinjäriteter som finns kompenseras, vilket resulterar i ett linjärt system.

Betrakta ekvationerna som beskriver modellen, hämtade från avsnitt 3.

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}\boldsymbol{\nu} \tag{6.1}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} \tag{6.2}$$

Där både $\boldsymbol{\nu}$ och $\boldsymbol{\eta}$ kommer att estimeras i delsystem Sensorfusion, och därmed finnas tillgängliga för delsystem Reglering. Genom att derivera den kinematiska ekvationen av systemet (6.1) med avseende på tiden fås

$$\dot{\boldsymbol{\nu}} = \boldsymbol{J}^{-1}[\ddot{\boldsymbol{\eta}} - \dot{\boldsymbol{J}}\boldsymbol{\nu}] \tag{6.3}$$

Eftersom $\boldsymbol{J}^{-1}$ inte säkert är inverterbar för alla värden på $\boldsymbol{\eta}$ så kommer pseudoinversen att användas. Om man väljer följande olinjära styrlag (där $\boldsymbol{a}^b$ beskriver accelerationen i det kroppsfixa koordinatsystemet)

$$\boldsymbol{\tau} = \boldsymbol{M}\boldsymbol{a}^b + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + g(\boldsymbol{\eta}) \tag{6.4}$$

och sätter in i (6.2) samtidigt som resultatet i (6.3) används, fås följande ekvation

$$\boldsymbol{M}(\dot{\boldsymbol{\nu}} - \boldsymbol{a}^b) = \boldsymbol{M}\boldsymbol{J}^{-1}[\ddot{\boldsymbol{\eta}} - \dot{\boldsymbol{J}}\boldsymbol{\nu} - \boldsymbol{J}\boldsymbol{a}^b] = 0 \tag{6.5}$$

Om man betraktar

$$\ddot{\boldsymbol{\eta}} = \boldsymbol{a}^n \tag{6.6}$$

där $\boldsymbol{a}^n$ ses som den önskade accelerationen i det jordfixa NED-koordinatsystemet. Genom att välja

$$\boldsymbol{a}^n = \dot{\boldsymbol{J}}\boldsymbol{\nu} + \boldsymbol{J}\boldsymbol{a}^b \tag{6.7}$$

och sätta in (6.7) i ekvation (6.5) fås följande linjära system

$$\boldsymbol{M}^*(\ddot{\boldsymbol{\eta}} - \boldsymbol{a}^n) = 0 \tag{6.8}$$

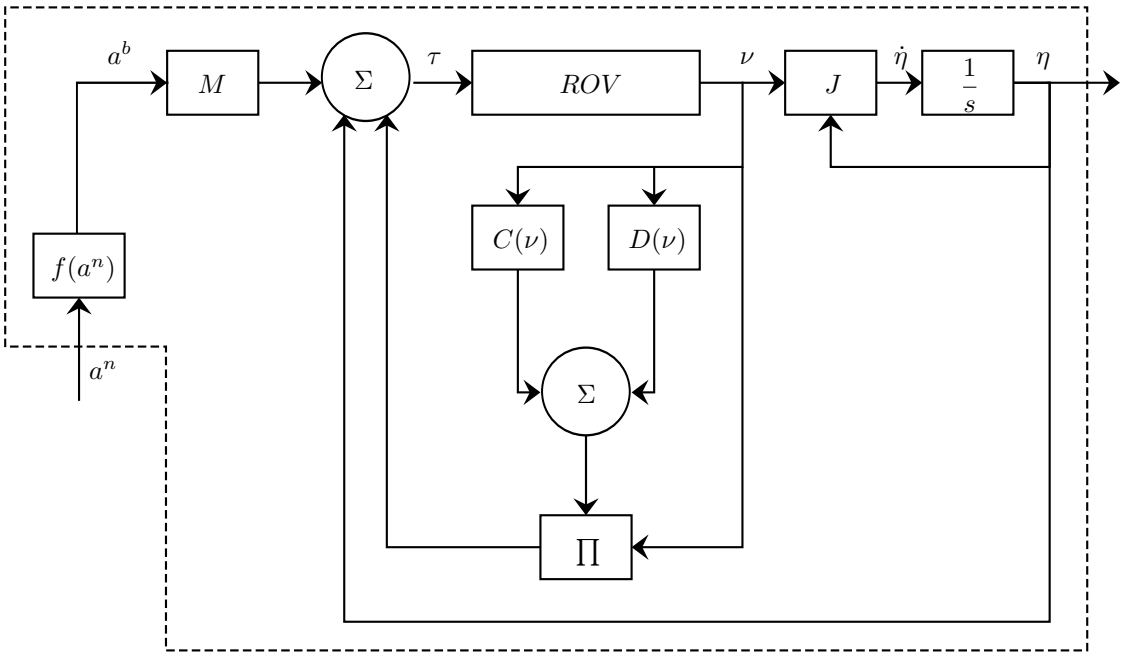
Där $\boldsymbol{M}^* = \boldsymbol{J}^{-T}\boldsymbol{M}\boldsymbol{J} > 0$. Ekvation (6.8) är därför ekvivalent med att

$$\ddot{\boldsymbol{\eta}} - \boldsymbol{a}^n = 0 \tag{6.9}$$

Från (6.7) kan man få ut att

$$\boldsymbol{a}^b = \boldsymbol{J}^{-1}[\boldsymbol{a}^n - \dot{\boldsymbol{J}}\boldsymbol{\nu}] = \boldsymbol{f}(\boldsymbol{a}^n) \tag{6.10}$$

$\boldsymbol{a}^n$ ses som styrsignal till det linjära systemet (6.9), och kommer att väljas av LQ-regulatorn. I figur 7 återfinns en schematisk bild på hur den exakta linjäriseringen av ROV:n kommer att gå tillväga.



Figur 7: En översikt av den exakta linjäriseringen av ROV:n, med a^n som insignal.

6.2.2 Diskretisering och utökning av modell

För att implementera en LQ-regulator behövs en tidsdiskretisering av den exakt linjäriserade modellen, detta görs enligt:

$$x_{k+1} = Fx_k + Gu_k \tag{6.11}$$

$$z_k = Cx_k \tag{6.12}$$

där x_k är tillståndsvektorn, u_k är styrsignalsvektorn och z_k är reglerstorhetsvektorn.

För att motverka stationära fel ska LQ-regulatorn ha integralverkan på pitch- och yawvinklar samt position. Detta kommer att implementeras genom att föra in ett antal nya tillstånd enligt:

$$x_{I,k+1} = x_{I,k} + T_s \cdot (z_k - r_k) \tag{6.13}$$

där z_k är vektorn med de tillstånd som integreras och r_k är vektorn med referensvärden. Ekvation (6.11) tillsammans med (6.13) ger det utökade systemet:

$$\begin{bmatrix} x_{k+1} \\ x_{I,k+1} \end{bmatrix} = F_u \begin{bmatrix} x_k \\ x_{I,k} \end{bmatrix} + G_u u_k - \begin{bmatrix} 0 \\ T_s I_5 \end{bmatrix} r_k \tag{6.14}$$

$$z_k = \begin{bmatrix} C & 0_{5\times 5} \end{bmatrix} \begin{bmatrix} x_k \\ x_{I,k} \end{bmatrix} \tag{6.15}$$

där

$$F_u = \begin{bmatrix} F & 0_{12x5} \\ CT_s & I_5 \end{bmatrix} \tag{6.16}$$

$$G_u = \begin{bmatrix} G \\ 0_{5x5} \end{bmatrix} \tag{6.17}$$

I_n är enhetsmatrisen med storlek $n \times n$ och $0_{n \times m}$ är nollmatrisen med storlek $n \times m$

6.2.3 Val av regulatorparametrar

Valet av tillståndsåterkoppling väljs genom att minimera kostnadsfunktionen nedan:

$$J_\infty(x_0) = \min_{x,u} \sum_{k=0}^{\infty} (\|x(k)\|_Q^2 + \|u(k)\|_R^2) \tag{6.18}$$

och sedan sätta styrsignalen enligt tillståndsåterkopplingen:

$$u_k = -Lx_k \tag{6.19}$$

L bestäms genom att lösa den algebraiska Riccatiekvationen för det linjära systemet (6.11).

Q och R i ekvation (6.18) är viktmatriser, $x(k)$ är tillståndsvektorn och $u(k)$ är styrsignalsvektorn. Viktmatrisernas förhållande till varandra avgör vilken vikt man ska lägga på snabbhet respektive storlek på styrsignaler. Är till exempel Q stor jämfört med R så ökar vikten av att systemet ska vara snabbt, på bekostnad av att styrsignalerna blir större. Straffmatriserna Q och R kommer att tas fram genom empiriska tester och kommer att utformas på så sätt att systemet uppfyller prestationskraven specifierade i kravspecifikationen [6].

6.3 Kommunikation med andra delsystem

Delsystem Reglering, som kommer vara implementerad som node *control* i ROS, ska ta emot referenssignaler från den externa PC:n samt information om vilket reglersystem som skall användas. Från delsystem Sensorfusion kommer skattade tillstånd att inhämtas och användas som mätsignaler. Utsignalerna från delsystemet, som är styrsignaler till motorerna, kommer att skickas till styr- och mätkortet, externa PC:n samt delsystem Sensorfusion, se figur 1.

7 Delsystem Sensorfusion

Delsystemet Sensorfusion består av ett filter som utgående från ett antal olika sensorer skattar ROV:ns tillstånd i form av en tredimensionell orientering (yaw, pitch och rollvinkel), tredimensionell position (n, e, d) relativt initialpositionen, samt motsvarande vinkelhastigheter och hastigheter. Dessa skattningar kommer sedan skickas vidare till berörda delsystem.

7.1 Kommunikation med andra delsystem

Oavsett vilket driftläge ROV:n befinner sig i kommer delsystemet att genomföra en skattning på systemets tillstånd. För att kunna göra dessa skattningar kommer sensordata hämtas från IMU:n, SONAR:erna och trycksensorn, samt att reglersystemet kommer skicka motorernas styrsignaler. De färdiga skattningarna kommer sedan att skickas vidare till de båda delsystemen Reglering och Kommunikation.

7.2 Observatör

Det finns två vanliga sätt att implementera EKF; EKF1 och EKF2. Man gör antingen en första eller en andra ordningens taylorutveckling av den olinjära tillståndsekvationen. Då EKF2 kräver extra beräkningar brukar man vanligtvis använda sig av EKF1 om det fungerar tillräckligt bra.

Med en olinjär modell från Gustafsson [4] enligt:

$$x_{k+1} = f(x_k, u_k) + v_k, \quad \text{där } v_k \sim N(0, Q_k) \tag{7.1a}$$
$$y_k = h(x_k, u_k) + e_k, \quad \text{där } e_k \sim N(0, R_k) \tag{7.1b}$$

erhålls algoritmen för EKF1 enligt följande.

$$S_k = R_k + h'(\hat{x}_{k|k-1})P_{k|k-1}(h'(\hat{x}_{k|k-1}))^T \tag{7.2a}$$
$$K_k = P_{k|k-1}(h'(\hat{x}_{k|k-1}))^T S_k^{-1} \tag{7.2b}$$
$$\epsilon_k = y_k - h'(\hat{x}_{k|k-1}) \tag{7.2c}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \epsilon 1_k \tag{7.2d}$$
$$P_{k|k} = P_{k|k-1} - P_{k|k-1}(h'(\hat{x}_{k|k-1}))^T S_k^{-1} h'(\hat{x}_{k|k-1})P_{k|k-1} \tag{7.2e}$$
$$\hat{x}_{k+1|k} = f(\hat{x}_{k|k}) \tag{7.2f}$$
$$P_{k+1|k} = Q_k + f'(\hat{x}_{k|k})P_{k|k}(f'(\hat{x}_{k|k}))^T \tag{7.2g}$$

Tillståndsmodellen $f(x_k, u_k)$ i ekvation (7.1a) finns beskriven under kapitel 3 och kommer representeras av ekvation (3.19).

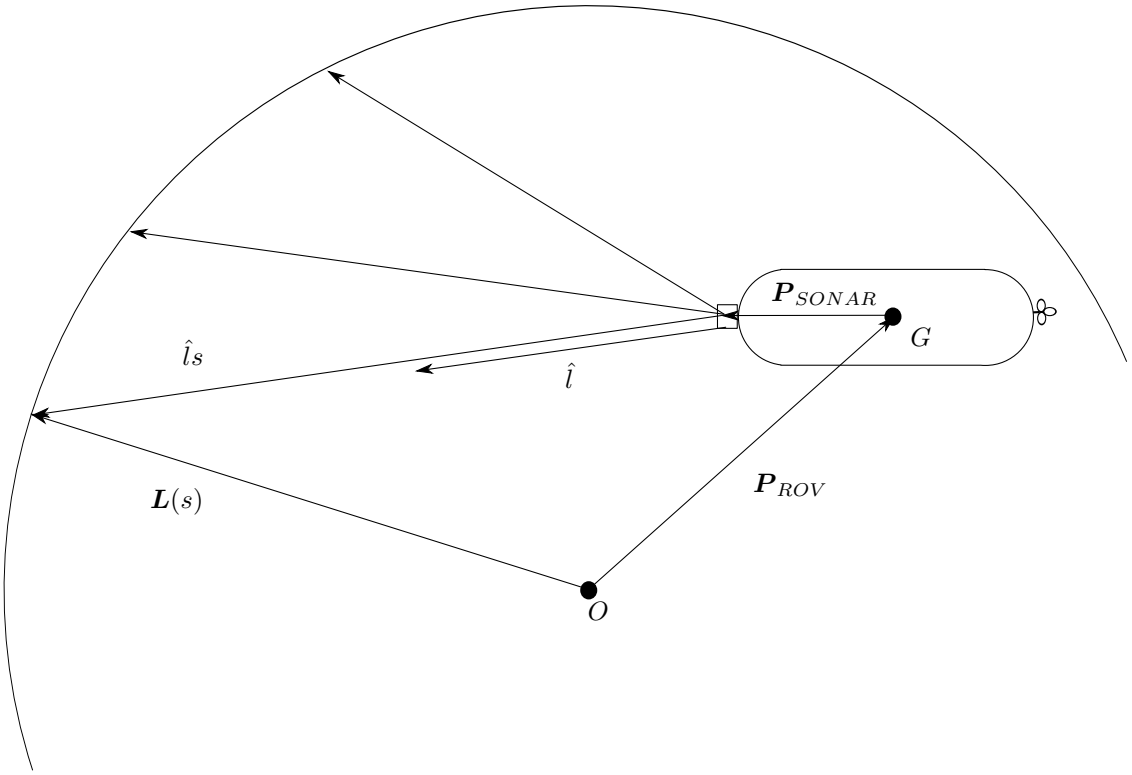
Mätmodellen $h(x_k, u_k)$ i ekvation (7.1b) fås från sensormodellerna som för SONAR:en ges av ekvation (7.7), trycksensorn ges av ekvation (7.9), accelerometern ges av ekvation (7.10), magnetometern ges av ekvation (7.12) och för gyroskopet ges av ekvation (7.15).

7.3 Mätuppdatering

Under denna delen kommer alla sensorers mätuppdateringar att beskrivas. Dessa mätuppdateringar kommer sedan användas i EKF:n för att skatta positioneringen och orienteringen.

7.3.1 Mätuppdatering SONAR

Ett förslag på algoritm för att positionera sig i testbassängen är att utnyttja att den är månghörning. Då kan bassängen approximeras med en cylinder och få fram en analytisk algoritm som är konvex om orienteringen antas vara given.



Figur 8: Vektorbeskrivning av SONAR:ens lober.

$$\boldsymbol{L}(s) = \boldsymbol{p}_{ROV} + R(q)\boldsymbol{p}_{SONAR,l} + R(q)\widehat{l}_{lob,l}s \tag{7.3}$$

I ekvation (7.3) är SONAR:en och loben angivna i rovens koordinatsystem och matrisen R roterar en vektor till ROV:ens globala orientering. Skalären s blir då det avstånd som beräknas från varje lob.

Vid stor pitch kommer botten eller ytan att börja påverka SONAR:en så en rimlig approximation är att pitchen antas vara noll och vid stora vinklar kan inte positioneringen användas. Den linjära ekvationen förenklas då till en ekvation i horisontalplanet där bassängen beskrivs med en cirkel.

$$\boldsymbol{L}(s)^T\boldsymbol{L}(s) = R_{radie}^2 \tag{7.4}$$

En avståndsmätning s representerar en position på cirkeln. Detta är en andragradsekvation som man lösas analytisk för s.

$$\boldsymbol{p}_{SONAR,g} = \boldsymbol{p}_{ROV} + R_{yaw}(q)\boldsymbol{p}_{SONAR,l} \tag{7.5}$$

$$\widehat{l}_{lob,g} = R_{yaw}(q)\widehat{l}_{lob,l} \tag{7.6}$$

$$s = \boldsymbol{p}_{SONAR,g} \cdot \widehat{l}_{lob,g} + \sqrt{R^2 - \boldsymbol{p}_{SONAR,g} \cdot \boldsymbol{p}_{SONAR,g} + \boldsymbol{p}_{rov} \cdot \widehat{l}_{lob,g}} \tag{7.7}$$

Det är bara skärningen mellan linjen och cirkeln framför ROV:en som är intressant så lösningen för negativa s används inte.

7.3.2 Mätuppdatering trycksensor

De mätvärden som kommer till sensorfusion från trycksensorn är absolut tryck.

$$P_{tot} = P_{luft} + P_{vatten} = P_{luft} + \rho g d \tag{7.8}$$

Mätuppdatering för trycksensorn blir.

$$d = \frac{P_{tot} - P_{luft}}{\rho g} \tag{7.9}$$

Eftersom lufttrycket kan variera 50 hPa från dag till dag enligt SMHI [7] måste lufttrycket kalibreras vid körning. Det görs genom att sätta P_{luft} till trycket från trycksensorn vid vattenytan. Densiteten för vatten ändras även med temperaturen, men i testbassängen kommer den att vara relativt konstant.

7.3.3 Mätuppdatering accelerometer

För att räkna ut pitch används accelerometern.

$$\boldsymbol{a} = R^T(\boldsymbol{q})(\boldsymbol{g} + \boldsymbol{a}_{\text{kropp}}), \quad \boldsymbol{g} = g\hat{\boldsymbol{d}} \tag{7.10}$$

Man vill använda denna mätuppdatering då $\boldsymbol{a}_{\text{kropp}}$ är liten, eftersom man då mäter tyngdkraften. Därför kastar man bort mätvärden då

$$||\boldsymbol{a}| - g| > \epsilon_a \tag{7.11}$$

där ϵ_a är en designparameter.

7.3.4 Mätuppdatering magnetometer

Magnetometern används för att bestämma yaw.

$$\boldsymbol{m} = R^T(\boldsymbol{q})(\boldsymbol{b}_{\text{jord}} + \boldsymbol{b}_{\text{el}}), \quad \boldsymbol{b}_{\text{jord}} = b_{\text{jord}} \sin(\theta_{\text{dip}})\hat{\boldsymbol{d}} + b_{\text{jord}} \cos(\theta_{\text{dip}})\hat{\boldsymbol{n}} \tag{7.12}$$

Där θ_{dip} är inklinationen för magnetfältet. Den är typiskt runt 71° i Sverige.

Elektronik i ROV:n kan skapa störningar b_{el} i magnetfältet. Dessa är oönskade, varför man kastar bort mätvärden då

$$||\boldsymbol{m}| - b_{\text{jord}}| > \epsilon_m, \quad \text{eller} \tag{7.13}$$

$$|\hat{\theta}_{\text{dip}} - \theta_{\text{dip}}| > \epsilon_\theta, \quad \hat{\theta}_{\text{dip}} = \arctan \frac{\hat{\boldsymbol{n}}^T R(\boldsymbol{q})\boldsymbol{m}}{\hat{\boldsymbol{d}}^T R(\boldsymbol{q})\boldsymbol{m}} \tag{7.14}$$

där ϵ_m och ϵ_θ är designparametrar.

7.3.5 Mätuppdatering gyro

Gyroskopet används för att mäta de tre vinkelhastigheterna på ROV:n. Vinkelhastigheterna p, q och r är definerade i kapitel 3 som roll, pitch och yaw.

$$\boldsymbol{\omega} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{7.15}$$

7.4 Signalbehandling SONAR-data

Större delen av signalbehandlingen sköts av sluten programvara från Imagenex, den kommer idealt köras på interna PC:en alternativt på en extra Windows-PC. Programvaran tar information från SONAR:en och skattar avståndsmätningar till objekt i varje lob.

8 Delsystem Kommunikation

Delsystem Kommunikation har till uppgift att sköta kommunikationen mellan den externa PC:n och övriga delar av ROV:n. Den har också till uppgift att kontrollera om kommunikationen mellan ROV:n och den externa PC:n bryts. I ROS är delsystemet implementerat som en node *com*.

8.1 Kommunikation med externa enheter

Delsystemet kommer att skicka data med sensorvärden, tillståndsskattningar och styrsignaler till den externa PC:n.

Den externa PC:n kommer att förse delsystemet med referenssignaler, val av manuellt eller stabiliserat läge, vilken regulator som ska användas, om ROV:n skall vara på eller avslagen samt om ROV:n ska sättas i nödläge. Denna data kommer sedan att delas upp och skickas vidare till de delar som är berörda.

Utav Arduinon blir delsystemet försett med data från läckagesensorn, batteriernas laddningsnivå samt data från trycksensorn.

8.2 Kommunikation med interna delsystem

Delsystemet skickar referenssignaler och information om manuellt eller stabiliserat läge samt val av regulator till delystem Reglering.

Delsystmet erhåller från delsystem Reglering motorernas utställda styrsignaler. Från delsystem Sensorfusion erhålls skattade tillstånd. Sensorvärden erhålls från IMU:n samt från Arduinokortet.

Om kontakten med den externa PC:n bryts, eller om den externa PC:n meddelar att ROV:n ska sättas i nödläge (vid läckage eller vid nedtryckning av emergency-knappen i GUI:t) kommer delsystem Kommunikation se till att ROV:n förs upp till vattenytan horisontellt. Detta genom att meddela delsystem Reglering att försätta ROV:n i nödläge. Nödläge innebär att ROV:n avbryter sin drift och förflyttar sig horisontellt till ytan.

9 Extern PC

Den externa PC:n är en laptop med operativsystemet Linux. Den har ett program med ett GUI, med vilket man kan styra ROV:n och välja vilket av de olika driftlägena som ska vara aktivt, se kapitel nedan.

Här kan man också hämta information om ROV:ns orientering och position. Den externa PC:n håller också koll på läckagesensorn och meddelar kommunikation om ROV:n ska sättas i nödläge. PC:n ansluts till ROV:n via en Ethernet-kabel.

9.1 GUI

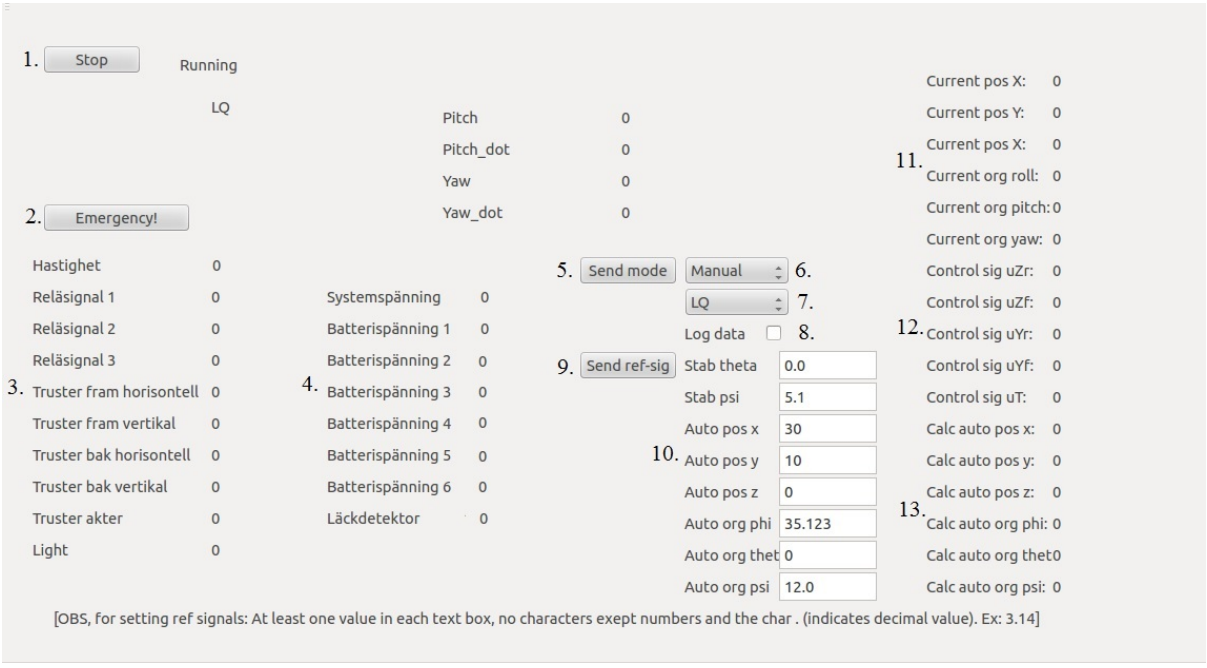
GUI:t är utvecklat av tidigare projekt och examensarbeten rörande ROV:n. Genom GUI:t kan man avläsa batterispänning, position och orientering. Vidare kan användaren utföra följande:

- Växla mellan nödläge, manuell och stabiliserad drift
- Växla mellan decentraliserad regulator och LQ-regulator
- Nödstopp

Larm visas i GUI:t i form av en popup-ruta. Larm inkluderar läckage, låg batterinivå och bruten kontakt mellan PC:n och ROV:n.

9.1.1 Utseende

I detta kapitel kommer GUI:ts utseende att förklaras.



Figur 9: En skärmdump av det nuvarande grafiska användargränssnittet. Vad varje siffra representerar står förklarat nedan.

Siffrorna i figur 9 är förklarade i följande lista:

1. *Start/Stopp*-knapp. Startar och stoppar systemet.

2. *Emergency*-knapp. Aktiverar nödläge. I nödläge så avbryter ROV:n sin drift och förflyttar sig horisontellt till ytan.
3. ROV:ns hastighet, relävärden, motorernas värden samt indikation om lampan är tänd eller ej.
4. ROV:ns systemspänning och batterispänning, samt indikation om läckage.
5. *Send mode*-knapp. Skickar vidare den information som valts i punkt 6-8 nedan.
6. Flervalsalternativ där användaren kan välja mellan nödläge, manuell eller stabiliserad drift.
7. Flervalsalternativ där användaren väljer mellan LQ-regulator eller decentraliserad reglulator.
8. Checkbox där användaren kan välja att spara data till loggfiler.
9. *Send ref-sig*-knapp. Skickar vidare de i punkt 10 valda referensvärderna, i stabiliserad drift.
10. Textfält där användaren kan skriva in värden för stabiliserad drift.
11. ROV:ns position och orientering.
12. ROV:ns styrsignaler till motorerna.
13. ROV:ns beräknade referenssignaler i autonom drift. Denna del kommer dock inte användas i detta projekt.

9.2 Xbox-kontroll

En Xbox-kontroll, se bild 10, kan anslutas till den externa PC:n för att styra ROV:n i de tre olika driftlägen som specificerats tidigare i dokumentet. När den decentraliserade regulatorn används ska vinkelhastigheterna kunna styras med hjälp av den högra joysticken på kontrollen. När LQ-regulatorn används så ska yaw- och pitchvinklarna styras med den högra joysticken och djup med den vänstra. Gemensamt för alla driftlägen är att det kommer finnas en knapp för nödstopp som försätter ROV:n i nödläge, vilket kommer föra den upp till ytan.

Figur 10: Xbox-kontrollen

Referenser

[1] Bernhard, J., Johansson, P., *Remote control of a remotely operated underwater vehicle*. Institutionen för systemteknik (ISY), Examensarbete, 2012.

[2] Fossen, T., *Guidance and Control of Ocean Vehicles*. Wiley, 1994.

[3] Fossen, T., *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley, 2011.

[4] Gustafsson, F., *Statistical Sensor Fusion*. Studentlitteratur, 2012.

[5] TSRT10 Projektgrupp ROV, Patricia Sundin m.fl. (2012),
Tillgänglig: <http://www.isy.liu.se/edu/projekt/tsrt10/2012/rov/> (2013-09-10)

[6] TSRT10 Projektgrupp ROV, Malte Moritz m.fl. (2013), *Kravspecifikation Remotely Operated Underwater Vehicle*.

[7] SMHI Lufttryck (2013),
Tillgänglig: <http://www.smhi.se/kunskapsbanken/meteorologi/lufttryck-1.657> (2013-09-10)

[8] ROS.org *ROS Concepts* Tillgänglig: <http://wiki.ros.org/ROS/Concepts> (2013-10-01)

A Topics och messages

I tabellen nedan så listas alla topics och deras innehåll.

Tabell 5: Topics och ders innehåll, samt publishers och subscribers

Topic	**Innehåll**	**Publisher**	**Subscribers**
top_Mode	Systemets status. Nödläge, manuell-, stabiliserad- eller autonom drift. LQ eller MPC.	com	control, sensor
top_States	Systemets skattade tillstånd.	sensor	control, sensor
top_ControlSig	Styrsignaler till motorer	control	com, sensor
top_SensorData	Data från sensorer	com	sensor
top_SetParam	Parametrar till LQ- och MPCregulatorer samt parametrar för filtrering.	com	sensor, control
arduino	Batterinivåer, läckage- och tryckinformation från Arduino.	arduino	com
control	Motor och relädata till Arduino.	arduino	com
GUI/update	Motordata, batterinivåer och läckagesensor-info.	com	GUI
GUI/send	Intern PC redo att ta emot Xbox-signaler	com	GUI
GUI/state	Stoppa/starta mainloopen.	GUI	com
imu/data	Data från IMU:n.	IMU	com
xbox	Data från Xbox handkontroll.	GUI	com
top_GuiMode	Systemets status. Nödläge, manuell-, stabiliserad- eller autonom drift. LQ eller MPC. Logga data till fil.	GUI	com
top_GuiRef	Referenssignaler till regulatorerna, i stabiliserad- och autonom drift.	GUI	com
top_GuiPing	Skickas till interna PC:n så att bruten kontakt med GUI:t ska kunna detekteras.	GUI	com
top_GuiControlSig	Styrsignaler till motorerna.	com	GUI
top_GuiStates	Systemets skattade tillstånd	com	GUI