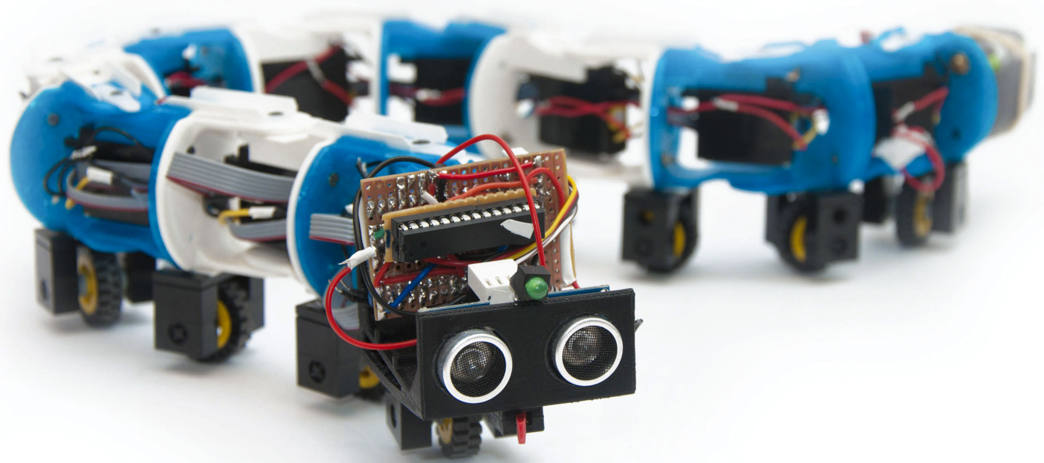
SnakeBot

En slingrande robot



Markus Hammarsten
Marcus Karlsson
Henrik Olsson
Erik Karlsson
Anneli Kalander
Patric Strandberg

Chalmers tekniska högskola
Institutionen för Signaler och System
Göteborg, Sverige 2014

Sammandrag

Denna rapport behandlar utvecklingen av en “Snake-like” robot. Målet var att utveckla en fysisk robot som kan röra sig framåt som en orm. Detta mål uppnåddes genom att en matematisk modell togs fram samt att simulationer genomfördes, varefter den fysiska roboten konstruerades.

Roboten består av tio moduler med en axel i varje. De är designade och monterade på ett sätt som möjliggör en ormliknande rörelse. Varje modul innehåller en servomotor som står för vridningen av modulen. Servomotorerna får signaler från ett kretskort som skickar ut vinklar jämnt fördelade på en sinuskurva till servomotorerna. En avståndssensor sitter monterad för att roboten ska kunna både undvika samt följa objekt. Viktigt är att roboten har lägre friktion framåt än åt sidorna vilket är nyckeln till biologiska ormars framdrivning. Detta friktionsmönster återskapas med hjälp av hjul.

Abstract

This report treats the development of a snake-like robot. The goal was to develop a physical robot which could move like a snake. This goal was achieved by developing a mathematical model and creating simulations, whereafter the physical robot was constructed.

The robot consists of ten modules with an axis in each which are designed and assembled in a way that enables a snakelike motion. Each module contains an actuator which generates the turning of the module. The actuators recieve signals from a circuit board which sends angles evenly distributed over a sinus curve to the actuators. A distance sensor is mounted so that the robot is able to both avoid and follow object. It is important that the robot has lower friction forward than to the sides which is the key to the motion of biological snakes. This friction pattern is imitated by means of wheels.

Förord

Denna rapport är, tillsammans med en robotorm, resultatet av ett kandidatarbete utfört på institutionen Signaler och System på Chalmers tekniska högskola vårterminen 2014.

Vi vill passa på att tacka Jonas Fredriksson, handledaren för detta kandidatarbete, som genom god feedback med många tips och korrekturläsning av rapporten har hjälpt oss att realisera projektets mål. Tack!

Innehåll

1 Inledning

Robotar utgör idag en allt viktigare del utav vår vardag. Utvecklingen har gått och går otroligt snabbt och det är detta tillsammans med robotars näst intill obegränsade potential som gör ämnet så intressant. Applikationsområdena är i det närmaste oändliga och robotar har kommit att ta över allt mer i till exempel fabriker i så kallade automationsprocesser. Det finns också många andra typer av robotar vars användningsområden varierar kraftigt. Gemensamt för många av dessa robotar är att de behöver någon form av framdrivning.

Den framdrivning som dagens robotar främst använder sig utav på land är hjulbaserad då denna är klart snabbast och energieffektivast. Dock är alternativet hjullös framdrivning intressant, något som naturen har lyckats väl med hos ormar. Fördelen med denna framdrivning mot hjulbaserad är dess mångsidighet och förmåga att ta sig fram i svår terräng och trånga utrymmen, [1]. Därför är det idag av intresse att med robotar återskapa denna rörelse. Detta innebär att robotarna skulle kunna användas för att ta sig fram i till exempel husras och leta efter överlevande vid katastrofer, som jordbävningar och översvämningar. De kan också användas till att undersöka platser som är otillgängliga för andra typer av robotar, detta inkluderar till exempel rör, ledningar och andra allmänt svåråtkomliga platser.

1.1 Syfte och Mål

Syftet med projektet är att undersöka ormliknande framdrivning av robotar med målet att bygga en prototyp som rör sig framåt likt en orm. Ytterligare mål är också att robotormen ska vara konstruerad för att kunna vidareutvecklas i ett senare skede.

1.2 Problem/Uppgift

Det huvudsakliga problemet består av att analysera vad som i ormars rörelse genererar en framåtkraft och hur detta kan realiseras i en robot. Problemet löses genom att besvara nedan ställda frågeställningar:

- Hur skapas den slingrande rörelsen som förflyttar en orm framåt och hur kan den beskrivas?
- Hur kan en ormliknande rörelse genereras?
- Hur ska ytan mot underlaget se ut för att roboten ska kunna röra sig framåt?
- Hur ska robotens stomme designas för att kunna generera nödvändiga rörelser och samtidigt få plats med de komponenter som krävs?
- Hur starka motorer krävs för att få roboten i rörelse?
- Hur ska roboten styras och kan undvikning av hinder implementeras?
- Vilken energikälla ska användas och hur ska den monteras i roboten?

1.3 Avgränsningar

För att underlätta arbetet och försäkra att satta mål uppnås avgränsas robotens funktionalitet. Projektet är även bundet till en budget som inte får överskridas.

1.3.1 Robotens funktionalitet

För att spara tid väljs en modulär design utan att alternativa lösningar undersöks. Fokus ligger på att konstruera en robot som kan röra sig framåt på plana ytor, då ormar oftast rör sig plant mot marken. Detta kräver endast en en-axlig robot.

1.3.2 Budget

Budgeten för projektet omfattar 5000 kr att använda till komponenter som behövs för att konstruera roboten. I ett prototyplabb finns det även tillgång till material för 2000 kr.

1.3.3 Förutsättningar

Projektet hade vid start redan tillgång till viss utrustning. Denna bestod utav:

- HS-81 servomotorer
- Arduino
- 3D-skrivare

1.4 Använda metoder

Projektet drivs framåt med hjälp utav en iterativ arbetsgång. Iterativ arbetsgång innebär att utvecklingen av en produkt sker successivt i flera steg. Först utvecklas en första version av produkten varefter den utvärderas och ligger till grund för en andra version av produkten, [10]. I den andra versionen har de fel och brister som uppdagats hos den första versionen åtgärdats. Detta medför att i utvärderingen av den andra versionen kan fel och brister i den första versionen upptäckas och åtgärdas till nästa version. Detta arbete upprepas tills dess att en funktionell produkt har erhållits.

I detta projekt itereras användandet av ett flertal olika metoder som alla bidrar med nödvändig kunskap för att ta projektet framåt. Dessa metoder finns listade nedan.

1.4.1 Litteraturstudie

En litteraturstudie används oftast i början av ett projekt. Den syftar till att ge den grundläggande informationen kring ett ämne som krävs för att ett projekt ska bli genomförbart. Genom att göra en litteraturstudie är det möjligt att bygga vidare på andras arbeten istället för att börja ett projekt helt från grunden.

1.4.2 Matematisk modellering

En matematisk modell används för att beskriva ett system eller beteende, [7]. Den matematiska modellen gör det möjligt att studera och förstå vad som ligger bakom att något beter sig eller fungerar på ett visst sätt. Ofta kombineras en modell också med en simulering som då gör det möjligt att testa hur olika parametrar påverkar varandra.

1.4.3 Simulering

Att simulera något är att vilja imitera eller återskapa ett verkligt beteende eller resultat men i en kontrollerad miljö. Fördelen med en simulering är att det oftast är betydligt billigare och enklare att simulera något än att genomföra ett verkligt test. I en simulering är det även lättare att dra slutsatser och upptäcka eventuella problem. Dessa fel kan då åtgärdas innan en verklig modell skapas. En simulering ska så långt som möjligt spegla de faktiska förhållandena som råder i verkligheten, [8].

1.4.4 Prototypbygge

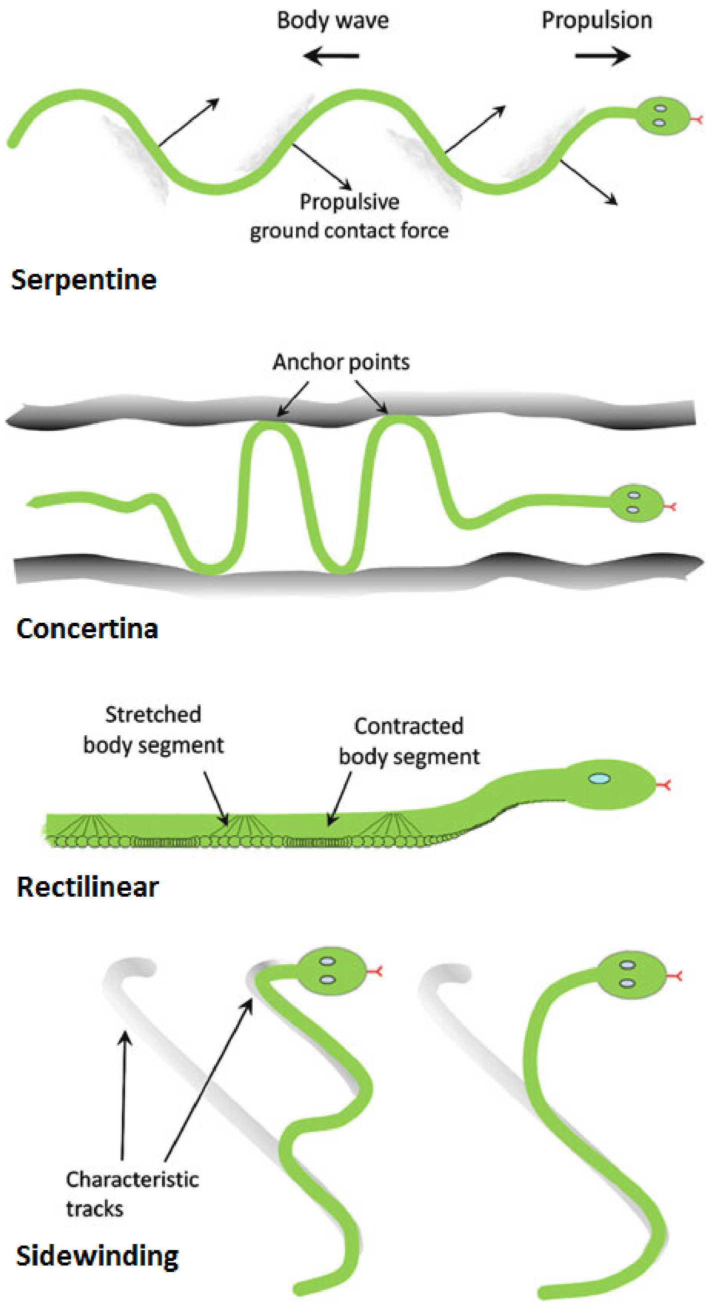
Att bygga en prototyp är att bygga en första version av slutprodukten för att se hur den teori och det förarbete som gjorts fungerar i praktiken. Tanken med att bygga en prototyp är att man på ett snabbt och enkelt sätt kan se vad som behöver förändras i utformningen för att skapa en slutprodukt med önskade egenskaper, [9].

1.5 Rapportens innehåll

Denna rapport ämnar ta upp de resultat som erhållits under projektets gång samt beskriva hur de togs fram. I kapitel 2 kommer ormars olika rörelsemönster att diskuteras vilka sedan används i kapitel 3 för att skapa modeller och simuleringar. Kapitel 4 tar upp hur roboten konstrueras med avseende på bland annat på design och styrning. De olika testerna som gjordes finns beskrivna i kapitel 5 och kapitel 6 tar upp förslag på återstående arbete som kan göras på projektet.

2 Ormars olika rörelsemönster

Det finns i dagsläget fyra kända rörelsemönster hos ormar. Vilken som används beror på hur omgivningen och terrängen ser ut, [2]. Dessa visas i figur 2.1 och beskrivs nedan.



Figur 2.1: Ormars olika rörelsemönster [1]

2.1 Serpentine

Serpentinrörelsen är det snabbaste och vanligaste sättet för ormar att ta sig fram på, [11]. Under denna rörelse utnyttjar ormen omgivningens friktion för att trycka sig framåt i en slingrande rörelse. Denna rörelse inbegriper även att ormen, vid topparna på kurvorna, lyfter kroppen lite för att få minskad friktion samt ökat tryck vid de punkter som trycker ormen framåt. Denna rörelse används även när ormar simmar och då utnyttjas vattnets friktion och tröghet för att generera den framåtriktade kraften.

2.2 Concertina

Concertina används i trånga utrymmen och innebär att kroppen samlas ihop i täta slingor för att fungera som ett ankare mot omgivningen. Huvudet förs sedan framåt så långt som möjligt och används därefter som ett nytt ankare samtidigt som resten av kroppen dras fram. Dessa två rörelser upprepas vilket förflyttar ormen framåt.

2.3 Sidewinding

Sidewinding används ofta av ormar som lever i sand eller på andra ytor med dåligt fäste. Rörelsen innebär att ormen kastar fram huvudet i en 45 graders riktning åt sidan samtidigt som resten av kroppen används som ankare. Sedan använder ormen huvudet som ankare då den flyttar fram kroppen. Detta lämnar flera parallella streck efter ormen om underlaget är mjukt.

2.4 Rectilinear

Rectilinear innebär att ormen drar ihop de bakre delarna och på så vis förkortar sig själv lite likt en larv. Därefter används fjällen som ankarpunkter samtidigt som ormen sträcker ut sig och förflyttas då framåt.

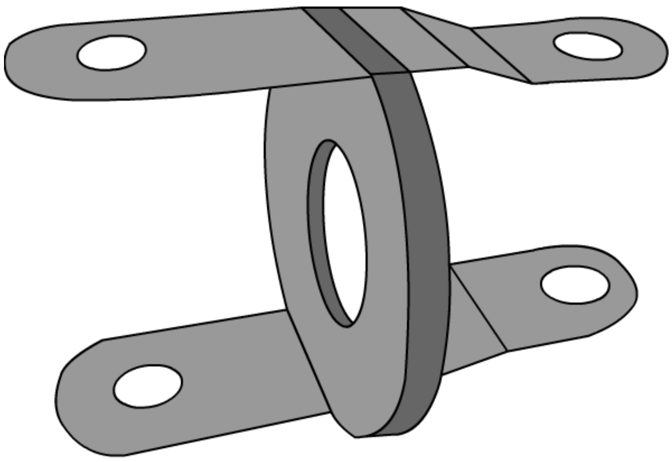
2.5 Slutsats för rörelsemönstren

Gemensamt för dessa rörelsemönster är att ormen utnyttjar att den har olika friktion i olika riktningar, så kallad *anisotrop* friktion, för att röra sig framåt. Detta åstadkoms genom att ormar utnyttjar omgivningen, lyfter olika delar av kroppen från underlaget och att skinnet består av överlappade fjäll vilka gör att friktionen i sidled och bakåt är mycket större än friktionen framåt. Om friktionen hade varit lika i alla riktningar hade ormen inte kunnat ta sig någonstans. Anisotrop friktion är alltså en av de viktigaste villkoren för att skapa ormlinknande framåtdrift.

3 Modellering och Simulering

I detta kapitel presenteras de matematiska modeller och samband som krävs för att kunna driva en robotorm. En matematisk modell för roboten tas fram för att göra det möjligt att undersöka hur en framåtdrivande rörelse kan skapas och friktionens betydelse för framdrivning tas upp. Modellen simuleras också för att göra det möjligt att testa robotormens egenskaper i datorn.

Ett begrepp som används frekvent i rapporten är länk. Med detta menas två "U-bottnar" som är monterade mot varandra. Denna enhet är stel och har ingen vridning inom sig (se figur 3.1).



Figur 3.1: Bild på en länk

3.1 Matematisk modell av serpentinrörelsen

Denna ansats bygger på en förenklad matematisk modell där punkter bundna med länkar ska följa en vanlig sinuskurva med hjälp av en optimeringsfunktion, [5]. Här antas att ormen under en serpentinrörelse rör sig med en sinusform och att detta kan generera en framåtrörelse.

3.1.1 Mål för serpentinmodellen

Målet med simuleringen av den förenklade modellen var att snabbt få fram en rörelse från vilken data kan hämtas, så som vinklar mellan varje länk och friktion mot underlaget som uppkommer vid rörelse. Detta för att snabbt och enkelt kunna testa att styra motorerna.

3.1.2 Framtagning av serpentinmodellen

Med hjälp av en optimeringsrutin i Matlab så placerades ett antal länkar ut längs en sinuskurva. Optimeringsrutinen kräver huvudsakligen två delar, dels en funktion som ska minimeras och dels en kravfunktion som beskriver de krav som ska vara uppfyllda. Referensen beskrivs med ekvation

$$x_{ref} = [0,l,2l,...,Nl], \tag{3.1}$$

tillsammans med

$$y_{ref} = \alpha \sin(x_{ref} \cdot N_{sin} \cdot \frac{2\pi}{L_x} + t\frac{2\pi}{T_{sin}}), \tag{3.2}$$

där parametrarna beskrivs i tabell 1. Denna funktion kan även relateras till 3.28 eftersom de båda beror på tiden på samma sätt. Minimeringsfunktionen beskriver summan av avstånden mellan varje länk och referensen på sinuskurvan i kvadrat, även kallat minsta kvadratmetoden vilken beskrivs med

$$minFun(\mathbf{X},\mathbf{Y}) = \sum_{i=0}^{N}(x_{ref,i} - x_i)^2 + (y_{ref,i} - y_i)^2. \tag{3.3}$$

Tabell 1: Parametrar till modellsats 1

Symbol	Enhet	Förklaring
N		Antalet länkar
l	m	Längden på varje länk
α	m	Bredden på robotens slingerrörelse
N_{sin}		Antalet sinusperioder ormen ska slingra sig
L_x	m	Robotens totala längd i x-led
t	s	Tiden
T_{sin}	s	Periodtiden för sinusvågen
$\mathbf{X},\mathbf{Y} \in \mathbb{R}^N$		Vektorer med x- och y-koordinater för den utplacerade ormen
$\mathbf{0}_{ref}$		En vektors vars element ska vara noll.
l_{ref}	m	Konstan som beskriver önskad länklängd

Punkterna ska ligga så nära referenskurvan som möjligt och kvadraten på avståndet gör att optimeringen troligare placerar alla länkar någorlunda nära kurvan. Om kvadraten utesluts kan istället optimeringen placera många länkar precis på referensen och ha en avstickande länk längre bort. Kravfunktionen beskriver att länkarna är låsta till en viss längd och inte får töjas ut (se ekvation 3.4). Vid varje tidpunkt i simuleringen förskjuts sinuskurvan och en ny optimerad position beräknas (se tidsberoendet i ekvation 3.2).

$$\mathbf{0}_{ref} = \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} - l_{ref} \tag{3.4}$$

Utifrån föregående position kan en rörelse tas fram och utifrån valda normalkrafter och friktionskoefficienter kan friktionskrafterna beräknas.

3.1.3 Resultat av serpentinmodellen

I figur 3.2 visas en stillbild från simuleringen där länkarna placerats ut längs en referenssignal formad som en sinuskurva. Simuleringen gjordes med parametrar enligt tabell 2.

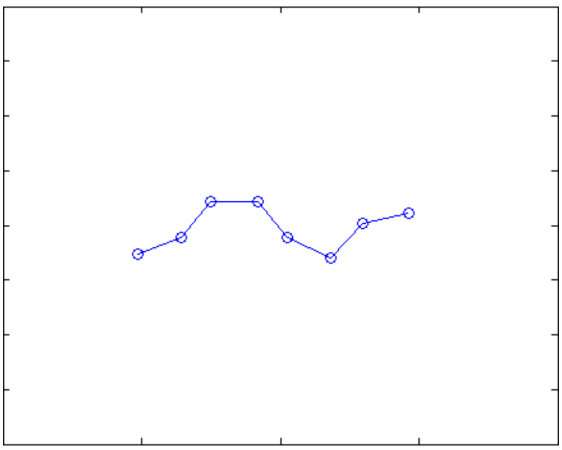
Tabell 2: Simuleringsparametrar

Amplitud rörelse	5,0 cm
Antal länkar	7
Länklängd	7,0 cm
Friktionskoefficient framåt	0.02
Friktionskoefficient sidor	0.80

Från simuleringen framgick det att en vanlig sinuskurva räcker för att få roboten att röra sig framåt.

Insignalerna till roboten i simuleringen var en referens för hur hela ormen skulle placeras och inte de enskilda vinklarna mellan varje länk vilket hade varit mer naturligt då dessa är de insignaler som behövs till den fysiska roboten. Enkelheten i denna simulering leder även till dess brister. Roboten finns inte beskriven som en matematisk modell vilket minskar modellens flexibilitet när det kommer till för att undersöka olika typer av rörelser.

I en simulering av ett rörelsemönster är det också önskvärt att beräkna den hastighet roboten kan uppnå genom sin rörelse. Då krävs korta intervaller mellan varje tidpunkt i simuleringen, något som med denna simulering innebär höga krav på datorkraft på grund av den krävande optimeringen som utförs i varje tidpunkt.



Figur 3.2: Stillbild från simulering

3.2 Matematisk modell av robotorm

För att kunna göra en grundlig analys på en robotorm behövs en matematisk modell. Förutom att modellen ger en grundläggande förståelse för robotormen så är den nödvändig för att kunna göra simuleringar och förstå hur styrningen ska fungera.

Detta kapitel är baserat på kapitel 2 i referens [1]. En grundlig genomgång av [1] finns i bilaga D. För att underlätta jämförelser med källan används samma symboler. De generella ekvationer för robotormen som används i [1] är på matrisform vilket gör det svårt att förstå ekvationernas innebörd. Detta kapitel ger därför också ett förenklat exempel i kapitel 3.2.5.

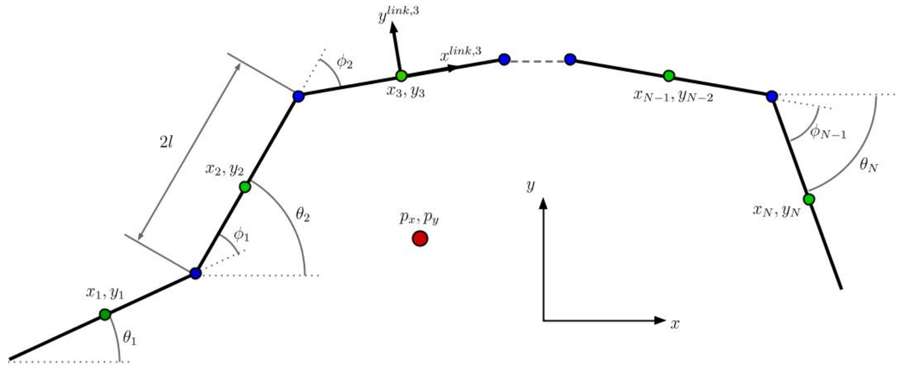
Det här kapitlet beskriver först en modell för en robotorm och dess parametrar i 3.2.2. Sedan redovisas de grundläggande förhållandena och ekvationerna för modellen, 3.2.3. En friktionsmodell tas fram i 3.2.4 följt av ett kapitel där dynamiken tas fram för en enkel robot, 3.2.5. I slutet sammanfattas den slutgiltiga modellen som tas fram i bilaga D och kapitel 2 i [1].

3.2.1 Mål för robotmodellen

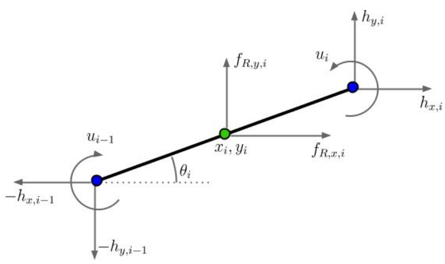
Målet med modellen är att få fram en tidsberoende tillståndsvektor för roboten där vektorns derivata är en funktion av dess nuvarande tillstånd och externa respektive pålagda krafter. De externa krafterna roboten utsätts för är friktionskrafter och de pålagda krafterna är momentet från servomotorerna i robotens leder. De slutgiltiga önskvärda tillstånden för modellen är ledernas vinklar och vinkelhastigheter samt masspunktens position och hastighet. Detta då lederna är de frihetsgrader som direkt kan påverkas av servomotererna och robotens masspunkt är det som bäst representerar robotens totala beteende.

3.2.2 Parametrar för modell av roboten

Modellen skapas i ett 2-dimensionellt horisontellt plan med homogent underlag. Robotormen är uppbyggd av N stela länkar som sitter i ihop med $N-1$ motoriserade leder. Varje länk har längden $2l$ och massan m. Bredden på varje länk tas inte hänsyn till i modellen. Då varje länk har homogen densitet hamnar varje MC (masscentrum) för varje länk vid längden l för varje länk. Tröghetsmomentet för varje länk blir då $J = \frac{1}{3}ml^2$. Den totala massa för roboten blir Nm. Länkarna påverkas dels av robotens interna krafter som är bindkrafterna $(h_{x,i},h_{y,i})$, $(-h_{x,i-1}, -h_{y,i-1})$ och ledkrafterna $(u_i,\ u_{i-1})$ och dels av de externa friktionskrafterna $(f_{R,x,i},f_{R,y,i})$ som uppstår på varje länk:s masscentrum. $(u_i,\ u_{i-1})$ är de styrbara krafterna (motorerna) i modellen. Modellens parametrar finns sammanställda i Tabell 3 och illustrerade i figurer 3.3 och 3.4.



Figur 3.3: De kinetiska parametrarna för robotormen (modifierad från [1])



Figur 3.4: Krafter och moment på varje länk i robotormen (modifierad från [1])

Tabell 3: Parametrar till modell av robotorm

Symbol	Enhet	Förklaring	Vektor
N		Antalet länkar	
m	kg	Massa per länk	
l	m	Halva längden per länk	
J	kgm^2	Tröghetsmoment på varje länk	
θ_i	rad	Vinkel mellan länk i och x-axeln	$\boldsymbol{\theta} \in \mathbb{R}^N$
ϕ	rad	Vinkel på led i	$\boldsymbol{\phi} \in \mathbb{R}^{N-1}$
x_i,y_i		Kordinater för MC på länk i	$\mathbf{X}$,$\mathbf{Y} \in \mathbb{R}^N$
p_x,p_y		Kordinater för robotens MC	$\boldsymbol{p} \in \mathbb{R}^2$
u_i	Nm	Drivande moment på länk i från länk $i+1$	$\boldsymbol{u} \in \mathbb{R}^{N-1}$
u_{i-1}	Nm	Drivande moment på länk i från länk $i-1$	$\boldsymbol{u} \in \mathbb{R}^{N-1}$
$f_{R,x,i}$,$f_{R,y,i}$	N	Friktionskrafter på länk i	$\boldsymbol{f}_{R,x}$,$\boldsymbol{f}_{R,y} \in \mathbb{R}^N$
$h_{x,i}$,$h_{y,i}$	N	Ledens tvångkraft på länk i från länk $i+1$	$\boldsymbol{h}_x$,$\boldsymbol{h}_y \in \mathbb{R}^{N-1}$
$-h_{x,i-1}, -h_{y,i-1}$	N	Ledens tvångkraft på länk i från länk $i-1$	$\boldsymbol{h}_x$,$\boldsymbol{h}_y \in \mathbb{R}^{N-1}$

Vektorerna som definieras i Tabell 3 är kolonnvektorer med dess parametrar, till exempel $\mathbf{X} = [x_1,...,x_N]^T \in \mathbf{R}^N$.

3.2.3 Ekvationer för robotmodellen

I detta kapitel undersöks modellens generella ekvationer. Genom att studera figur 3.3 kan samband mellan länkarnas position (x_i,y_i), deras vinklar θ_i, ledernas vinklar ϕ_i och robotens MC (p_x,p_y) erhållas. Ur figur 3.4 kan kraft och momentbalans för varje länk tas fram. Notera att robotormen består av $N + 2$ frihetsgrader, vilket är vinklarna på varje länk och MC:s position i planet.

Ur figur 3.3 utläses att ledernas vinklar relaterar till länkarnas vinklar enligt

$$\phi_i = \theta_i - \theta_{i+1} \tag{3.5}$$

och ormens MC $\boldsymbol{p}$ relaterar till länkarnas position enligt

$$\boldsymbol{p} = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} \frac{1}{Nm}\sum_{i=1}^{N} mx_i \\ \frac{1}{Nm}\sum_{i=1}^{N} my_i \end{bmatrix} = \frac{1}{N}\begin{bmatrix} \boldsymbol{e}^T\mathbf{X} \\ \boldsymbol{e}^T\mathbf{Y} \end{bmatrix}. \tag{3.6}$$

Då alla länkarna är homogena blir MC länkarnas medelposition.

Sambandet mellan länk i och länk $i + 1$ i led i där länkarna är bundna till varandra ger en relation mellan länkarnas position (x_i,x_{i+1}) och länkarnas vinkel (θ_i,θ_{i+1}) enligt

$$x_{i+1} - x_i = l\cos\theta_i + l\cos\theta_{i+1}, \tag{3.7a}$$

$$y_{i+1} - y_i = l\sin\theta_i + l\sin\theta_{i+1}. \tag{3.7b}$$

Kraftbalansen för en generell länk, i, erhålls ur figur 3.4 enligt

$$m\ddot{x}_i = f_{R,x,1} + h_{x,i} - h_{x,i-1}, \tag{3.8a}$$

$$m\ddot{y}_i = f_{R,y,1} + h_{y,i} - h_{y,i-1}. \tag{3.8b}$$

$$\tag{3.8c}$$

och momentbalansen för en generell länk, i, ges av

$$J\ddot{\theta}_i = u_i - u_{i-1} - l\sin(\theta_i)(h_{x,i} + h_{x,i-1}) + l\cos(\theta_i)(h_{y,i} + h_{y,i-1}). \tag{3.9}$$

3.2.4 Friktionsmodell för roboten

Som beskrivs i kapitel 2 så är friktionen en viktig faktor för hur roboten rör sig framåt. Det förklaras också att om en robotorm som med hjälp av en vågrörelse och en konstant kontakt mot underlaget ska kunna ta sig framåt krävs att länkarna har anisotrop friktion mot underlaget. Det vill säga har olika friktion i länkens riktning, respektive normalriktning.

Det är alltså viktigt att en verklighetstrogen modell för friktionen tas fram. Den mest verklighetstrogna modellen, som är önskvärd att använda ur simuleringssynpunkt, är Coulumbs modell där friktionen är proportionell mot länkarnas vikt. Friktionskraften är riktad i motsatt riktning mot länkens hastighet.

Friktionskoefficienterna betecknas μ_t och μ_n och är koefficienter i länkens riktning, $x^{link,i}$, respektive normalriktning, $y^{link,i}$ (se figur 3.3). Friktionskraften ($f_{R,x,i}$,$f_{R,y,i}$) är riktad i motsatt riktning mot länkens hastigheter ($\dot{x}_i$,$\dot{y}_i$). Detta leder till att en $Sign$ funktion (ger tecknet på dess parameter: $1{,}0{,}-1$) naturligt kan användas då tecknet på friktionskrafterna är beroende av tecknet på hastigheterna. Notera att isotrop friktion kan skapas genom att sätta μ_t och μ_n lika.

På grund av att friktionskoefficienterna följer länkarnas riktningar, medan länkarnas hastighet beskrivs med x:y-riktningar, så blir de annars enkla friktionsekvationerna relativt svåra. Nedan ges den generella ekvationen för friktionen på varje länk enligt

$$\boldsymbol{f}_R = \begin{bmatrix} \boldsymbol{f}_{R,x} \\ \boldsymbol{f}_{R,y} \end{bmatrix} = -mg \begin{bmatrix} \mu_t \mathbf{C}_\theta & -\mu_n \mathbf{S}_\theta \\ \mu_t \mathbf{S}_\theta & \mu_n \mathbf{C}_\theta \end{bmatrix} sign\left(\begin{bmatrix} \mathbf{C}_\theta & \mathbf{S}_\theta \\ -\mathbf{S}_\theta & \mathbf{C}_\theta \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\mathbf{Y}} \end{bmatrix} \right) \in \mathbb{R}^{2N}. \tag{3.10}$$

För en analys av robotens rörelse och de krafter som friktionen påverkar ormen med är Coulumbs modell och ekvation 3.10 komplicerade. Därför introduceras nedan även en viskös friktionsmodell från kapitel 2.5 i [1]. Denna modell antar att friktionen är linjär mot hastigheten och tar därför bort behovet av $Sign$ funktionen. Den färdiga modellen visas nu i sin helhet och används vid analys av drivande krafter på robotormen i kapitel 3.3.1.

$$\boldsymbol{f}_R = \begin{bmatrix} \boldsymbol{f}_{R,x} \\ \boldsymbol{f}_{R,y} \end{bmatrix} = - \begin{bmatrix} c_t(\mathbf{C}_\theta)^2 + c_n(\mathbf{S}_\theta)^2 & (c_t - c_n)\mathbf{S}_\theta \mathbf{C}_\theta \\ (c_t - c_n)\mathbf{S}_\theta \mathbf{C}_\theta & c_t(\mathbf{S}_\theta)^2 + c_n(\mathbf{C}_\theta)^2 \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\mathbf{Y}} \end{bmatrix} \in \mathbb{R}^{2N}, \tag{3.11}$$

där c_t och c_n är nya friktionskoeficienter.

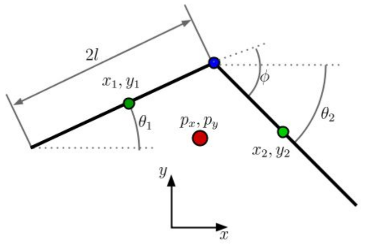
3.2.5 Dynamik för enkel modell med två länkar

I detta kapitel tas dynamiken för en enkel robot med bara två länkar fram. Detta görs för att få en förståelse för hur dynamiken för en generell robotorm med N kan beräknas. Dynamiken för den enkla roboten beskriver länkarnas och MC:s acceleration ($\ddot{\theta}_1$,$\ddot{\theta}_2$,$\ddot{p}_x$,$\ddot{p}_y$,) som en funktion av deras tillstånd ($\dot{\theta}_1$,θ_1,$\dot{\theta}_2$,θ_2,$\dot{p}_x$,p_x,$\dot{p}_y$,p_y) och krafter på ormen, vilket är friktionskrafterna ($f_{R,x,1}$,$f_{R,y,1}$,$f_{R,x,2}$,$f_{R,y,2}$) och motorns kraft i leden (u).

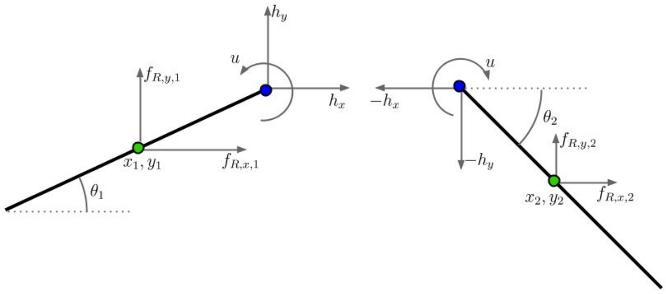
Den enkla roboten har parametrar enligt figur 3.5 som är en förenkling av figur 3.3. Kraftmodellen från figur 3.4 och friläggning av de två länkarna i den enkla roboten ger kraftförhållandena som visas i figur 3.6.

Ur figur 3.6 erhålls kraftbalansen för de två länkarna enligt

$$m\ddot{x}_1 = f_{R,x,1} + h_x, \tag{3.12a}$$
$$m\ddot{y}_1 = f_{R,y,1} + h_y, \tag{3.12b}$$
$$m\ddot{x}_2 = f_{R,x,2} - h_x, \tag{3.12c}$$
$$m\ddot{y}_2 = f_{R,y,2} - h_y. \tag{3.12d}$$



Figur 3.5: Kinetiska parametrarna för enkel modell



Figur 3.6: Friläggning av enkel modell

De två länkarnas acceleration kan även erhållas genom att först skriva om ekvation 3.7 till den enkla modellen och derivera två gånger med avseende på tiden vilket ger

$$x_2 - x_1 = l(\cos(\theta_1) + \cos(\theta_2)) \Rightarrow \left\{\frac{d^2}{dt^2}\right\} \Rightarrow \tag{3.13a}$$

$$\ddot{x}_1 - \ddot{x}_2 = l(\ddot{\theta}_2 \sin(\theta_1) + \dot{\theta}_2^2 \cos(\theta_1) + \ddot{\theta}_2 \sin(\theta_2) + \dot{\theta}_2^2 \cos(\theta_2)), \tag{3.13b}$$

$$y_2 - y_1 = l(\sin(\theta_1) + \sin(\theta_2)) \Rightarrow \left\{\frac{d^2}{dt^2}\right\} \Rightarrow \tag{3.13c}$$

$$\ddot{y}_1 - \ddot{y}_2 = l(-\ddot{\theta}_1 \cos(\theta_1) + \dot{\theta}_1^2 \sin(\theta_1) - \ddot{\theta}_2 \cos(\theta_2) + \dot{\theta}_2^2 \sin(\theta_2)). \tag{3.13d}$$

MC:s acceleration erhålls genom att skriva om ekvation 3.6 till den enkla modellen och derivera två gånger med avseende på tiden vilket ger

$$p_x = \frac{x_1 + x_2}{2} \Rightarrow \left\{\frac{d^2}{dt^2}\right\} \Rightarrow \ddot{p}_x = \frac{\ddot{x}_1 + \ddot{x}_2}{2}, \tag{3.14a}$$

$$p_y = \frac{y_1 + y_2}{2} \Rightarrow \left\{\frac{d^2}{dt^2}\right\} \Rightarrow \ddot{p}_y = \frac{\ddot{y}_1 + \ddot{y}_2}{2}. \tag{3.14b}$$

Kontaktkrafterna (h_x,h_y) elimineras genom att sätta in ekvation 3.12 i 3.14 vilket resulterar i

$$\ddot{p}_x = \frac{1}{2m}(f_{R,x,1} + f_{R,x,2}), \tag{3.15a}$$

$$\ddot{p}_y = \frac{1}{2m}(f_{R,y,1} + f_{R,y,2}). \tag{3.15b}$$

Som väntat ger detta att accelerationen på MC är lika med de externa krafterna på roboten delat på dess massa.
Nedan tas momentbalansen fram för de två länkarna utifrån figur 3.6

$$J\ddot{\theta}_1 = u - l\sin(\theta_1)h_x + l\cos(\theta_1)h_y, \tag{3.16a}$$

$$J\ddot{\theta}_2 = -u - l\sin(\theta_2)h_x + l\cos(\theta_2)h_y. \tag{3.16b}$$

Kontaktkrafterna tas bort ur 3.16 genom att lösa ut (x_1,x_2,y_1,y_2) ur ekvation 3.12 och stoppa in i 3.13 enligt

$$\frac{f_{R,x,1} - f_{R,x,2} + 2h_x}{m} = l(\ddot{\theta}_2 \sin(\theta_1) + \dot{\theta}_2^2 \cos(\theta_1) + \ddot{\theta}_2 \sin(\theta_2) + \dot{\theta}_2^2 \cos(\theta_2)), \tag{3.17a}$$

$$\frac{f_{R,y,1} - f_{R,y,2} + 2h_y}{m} = l(-\ddot{\theta}_1 \cos(\theta_1) + \dot{\theta}_1^2 \sin(\theta_1) - \ddot{\theta}_2 \cos(\theta_2) + \dot{\theta}_2^2 \sin(\theta_2)). \tag{3.17b}$$

h_x och h_y löses ut ur 3.17

$$h_x = \frac{f_{R,x,2} - f_{R,x,1}}{2} + \frac{ml}{2}(\ddot{\theta}_2 \sin(\theta_1) + \dot{\theta}_2^2 \cos(\theta_1) + \ddot{\theta}_2 \sin(\theta_2) + \dot{\theta}_2^2 \cos(\theta_2)), \tag{3.18a}$$

$$h_y = \frac{f_{R,y,2} - f_{R,y,1}}{2} + \frac{ml}{2} l(-\ddot{\theta}_1 \cos(\theta_1) + \dot{\theta}_1^2 \sin(\theta_1) - \ddot{\theta}_2 \cos(\theta_2) + \dot{\theta}_2^2 \sin(\theta_2)). \tag{3.18b}$$

I sista steget sätts ekvation 3.18 in i ekvation 3.16 vilket löser ut $\ddot{\theta}_1$, $\ddot{\theta}_2$.

$$\begin{aligned} &\ddot{\theta}_1(J + \frac{ml^2}{2}(\sin^2\theta_1 + \cos^2\theta_1)) + \ddot{\theta}_2(\frac{ml^2}{2}(\sin\theta_1 \sin\theta_2 + \cos\theta_1 \cos\theta_2)) \\ &+ \dot{\theta}_2^2(\frac{ml^2}{2}(\sin\theta_1 \cos\theta_2 - \cos\theta_1 \sin\theta_2)) \\ -f_{R,x,1}(\frac{l}{2}\sin\theta_1) + f_{R,x,2}(\frac{l}{2}\sin\theta_1) &+ f_{R,y,1}(\frac{l}{2}\cos\theta_1) + f_{R,y,2}(\frac{l}{2}\cos\theta_1) = u, \end{aligned} \tag{3.19}$$

$$\begin{aligned} &\ddot{\theta}_1(\frac{ml^2}{2}(\sin\theta_1 \sin\theta_2 + \cos\theta_1 \cos\theta_2)) + \ddot{\theta}_2(J + \frac{ml^2}{2}(\sin^2\theta_2 + \cos^2\theta_2)) \\ &+ \dot{\theta}_1^2(\frac{ml^2}{2}(\cos\theta_1 \sin\theta_2 - \sin\theta_1 \cos\theta_2)) \\ -f_{R,x,1}(\frac{l}{2}\sin\theta_2) + f_{R,x,2}(\frac{l}{2}\sin\theta_2) &+ f_{R,y,1}(\frac{l}{2}\cos\theta_2) + f_{R,x,2}(\frac{l}{2}\cos\theta_2) = -u, \end{aligned} \tag{3.20}$$

som kan sammanfattas med

$$\begin{aligned} &\begin{bmatrix} J + \frac{ml^2}{2}(\sin^2\theta_1 + \cos^2\theta_1) & \frac{ml^2}{2}(\sin\theta_1 \sin\theta_2 + \cos\theta_1 \cos\theta_2) \\ \frac{ml^2}{2}(\sin\theta_1 \sin\theta_2 + \cos\theta_1 \cos\theta_2) & J + \frac{ml^2}{2}(\sin^2\theta_2 + \cos^2\theta_2) \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} \\ + &\begin{bmatrix} 0 & \frac{ml^2}{2}(\sin\theta_1 \cos\theta_2 - \cos\theta_1 \sin\theta_2) \\ (\frac{ml^2}{2}(\cos\theta_1 \sin\theta_2 - \sin\theta_1 \cos\theta_2) & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \end{bmatrix} \\ + &\begin{bmatrix} -\frac{l}{2}\sin\theta_1 & \frac{l}{2}\sin\theta_1 \\ -\frac{l}{2}\sin\theta_2 & \frac{l}{2}\sin\theta_2 \end{bmatrix} \begin{bmatrix} f_{R,x,1} \\ f_{R,x,2} \end{bmatrix} + \begin{bmatrix} \frac{l}{2}\cos\theta_1 & \frac{l}{2}\cos\theta_1 \\ \frac{l}{2}\cos\theta_2 & \frac{l}{2}\cos\theta_2 \end{bmatrix} \begin{bmatrix} f_{R,y,1} \\ f_{R,y,2} \end{bmatrix} = \begin{bmatrix} u \\ -u \end{bmatrix} \end{aligned} \tag{3.21}$$

Tillsammans med ekvation 3.15 beskriver dessa ekvationer den enkla modellens dynamik. Anmärkningsvärt är att det räcker med bara två länkar för att skapa relativt svåra ekvationer. Genom att skriva om ekvationerna för roboten i matrisform kan ekvationerna lättare implementeras och även göras generella för en robotorm med N länkar.

Genom att introducera tillståndsvektor $\mathbf{x} = [\theta_1,\theta_2,p_x,p_y,\dot{\theta}_1,\dot{\theta}_2,\dot{p}_x,\dot{p}_y]^T$ kan den enkla modellen skrivas på tillståndsformen

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{p}_x \\ \dot{p}_y \\ \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{p}_x \\ \ddot{p}_y \end{bmatrix} = \mathbf{F}(\mathbf{x},u) \tag{3.22}$$

där $\mathbf{F}(\mathbf{x},u)$ kan lösas genom att lösa ut $\ddot{\theta}_1,\ddot{\theta}_2,\ddot{p}_x$ och $\ddot{p}_y$ ur ekvation 3.21.

3.2.6 Ekvationer för att styra roboten

För att kunna styra en robotorm med N antal länkar måste ekvation 3.21 skrivas om i matrisform vilket görs i bilaga D.1.6 eller kapitel 2.6 i [1]. Ekvationen måste även skrivas om så att drivande frihetsgrader och icke drivande frihetsgrader separeras, vilket görs i bilaga D.1.7 och kapitel 2.7 i [1]. Detta betyder att dynamiken inte beror på länkvinklarna $\boldsymbol{\theta}$ utan på ledvinklarna $\boldsymbol{\phi}$ som går att styra. De icke drivande frihetsgraderna är positionen av robotens MC och sista länkens vinkel, det vill säga huvudets riktning i rummet. En förenkling kan göras där ledernas vinkelaccelerationer $\ddot{\boldsymbol{\phi}}$ direkt kan regleras med en styrsignal $\bar{\mathbf{u}}$ vilket beskrivs i bilaga D.1.8 och kapitel 2.8 i [i].

Sammanfattningvis betyder detta att robotormens leder kan regleras med en styrsignal som är anpassad efter önskat rörelsemönster enligt ekvation D.37a. Robotens riktning och position i rummet påverkas av styrsignalen och robotens nuvarande tillstånd enligt ekvation D.37b. För att få fram vilket moment som lederna utsätts för används ekvation D.36

3.3 Modell för framdrivning av robotorm

Ett av projektets huvudmål är att skapa framåtrörelse för en robotorm och i detta kapitel ska det undersökas hur framdrivningen kan åstadkommas. Modellen från kapitel 3.2 ligger som underlag för undersökningen. Kapitlet baseras på de tillvägagånssätt och resultat som erhålls i kapitel 4 i [1]. Det första delkapitlet undersöker villkoren för att skapa framåtdrivande kraft för robotormen, det andra kapitlet undersöker själva rörelsemönstret och det tredje kapitlet undersöker hur en sväng kan åstadkommas.

3.3.1 Analys av framåtdrivande kraft för robotormen

Som nämnts tidigare i kapitel 2 är anisotrop friktion ett villkor för framåtdrivning på plant underlag och i detta kapitel undersöks hur detta påverkar framåtrörelsen. Detta görs genom att undersöka rörelseekvationerna för robotens MC från kapitel 3.2 och D.1. Den friktionsmodell som används i detta kapitel är den viskösa modellen, ekvation 3.11, då den är lättare att analysera än Coulumbs modellen. De två modellerna är kvalitativt lika bra (dock inte kvantitativt).

För att undersöka modellen förutsätts att den framåtdrivande rörelsen går längs den globala x-axeln vilket betyder att den totala drivande kraften är summan av alla externa krafter på roboten i x-led. Den drivande kraften på robotormen, det vill säga på dess MC, benämns F_{driv}. Genom ekvation D.24b och 3.11 kan F_{driv} beräknas enligt

$$F_{driv} = Nm\ddot{p}_x = \mathbf{e}^T \mathbf{f}_{R,x} = -\mathbf{e}^T((c_t(\mathbf{C}_\theta)^2 + c_n(\mathbf{S}_\theta)^2)\dot{\mathbf{X}} + (c_t - c_n)\mathbf{S}_\theta \mathbf{C}_\theta \dot{\mathbf{Y}}). \tag{3.23}$$

Vektorn $\mathbf{e}^T = [1, \ldots, 1] \in \mathbb{R}^N$ multiplicerat med $\mathbf{f}_{R,x}$ betyder att man summerar alla element i $\mathbf{f}_{R,x}$. Detta betyder att ekvation 3.23 kan skrivas som

$$F_{driv} = -\sum_{i=1}^{N}((c_t \cos^2\theta_i + c_n \sin^2\theta_i)\dot{x}_i + (x_t - c_n)\sin\theta_i \cos\theta_i \dot{y}_i). \tag{3.24}$$

Den drivande kraften som varje länk bidrar med, $F_{driv,i}$ ges alltså av

$$F_{driv,i} = -F_x(\theta_i)\dot{x}_i - F_y(\theta_i)\dot{y}_i, \tag{3.25}$$

där

$$F_x(\theta_i) = c_t \cos^2\theta_i + c_n \sin^2\theta_i, \tag{3.26}$$

$$F_y(\theta_i) = (c_t - c_n)\sin\theta_i \cos\theta_i. \tag{3.27}$$

Ur ekvation 3.25 kan utläsas att den drivande kraften framåt består av två delar, en som beror på länkens rörelse i robotens rörelseriktning, $F_x(\theta_i)\dot{x}_i$, och en som beror på länkens rörelse i robotens normalriktning, $F_y(\theta_i)\dot{y}_i$. Minustecken innebär att dessa delar bidrar till den drivande kraften när de är negativa. c_t, c_n och $F_x(\theta_i)$ är alltid positiva. Här förutsätts att roboten rör med positiv riktning, $\dot{p}_x > 0$, och att ingen länk rör sig mot robotens rörelseriktning, $\dot{x}_i > 0$ vilket resulterar i att produkten $F_x(\theta_i)\dot{x}_i$ alltid kommer att vara positiv. Detta betyder att denna produkt aldrig bidrar till den drivande kraften utan fungerar som en ren friktionskraft mot rörelseriktningen, vilket är naturligt.

Så den drivande kraften måste skapas av länkarnas rörelse i sidled mot rörelseriktningen. Undersökning av ekvation 3.27 ger att om $c_n = c_t$ så blir $F_y(\theta_i) = 0$. Om $c_n > c_t$ dock så måste $F_y(\theta_i)\dot{y}_i > 0$ för att skapa en drivande kraft. Här antas att alla länkar är riktade i rörelseriktnigen, det vill säga $-90° < \theta_i < 90°$, vilket gör att $sign(F_i(\theta_i))$ beror på $sign(\sin(\theta_i)) = sign(\theta_i)$. Varje länk bidrar alltså till den framåtdrivande kraften genom förhållandet $sign(F_{driv,i}) = sign(sign(\theta_i)sign(\dot{y}_i))$.

Genom att undersöka extremvärdet på $F_y(\theta_i)$ syns att den största drivande kraften fås vid $\theta_i = \pm 45°$. Det betyder att en länk som är vriden 45° mot robotens rörelseriktnig bidrar med den största drivande kraften i rörelseriktningen. Den drivande kraften kan också ökas genom att öka skillnaden mellan c_t och c_n.

3.3.2 Framåtdrivande rörelse för robotormen

I kapitel 3.3.1 förklarades hur den drivande kraften för en robotorm skapades vilket uppstod av friktionskrafterna från länkarnas rörelse i normalrikning mot robotens rörelse. För att skapa en framåtdrivande rörelse borde därför länkarna pendla mellan att åka åt höger respektive vänster i robotens rörelseriktning.

Utifrån serpentinrörelsen som presenterades i kapitel 2, de resultat som togs fram i kapitel 3.1 och kapitel 4.6-4.7 i [1] så ges här den grundläggande ekvationen för hur robotormens leder ska förhålla sig till varandra under framåtrörelse.

$$\phi_{ref,i} = \alpha \sin(\omega t + (i-1)\delta), \tag{3.28}$$

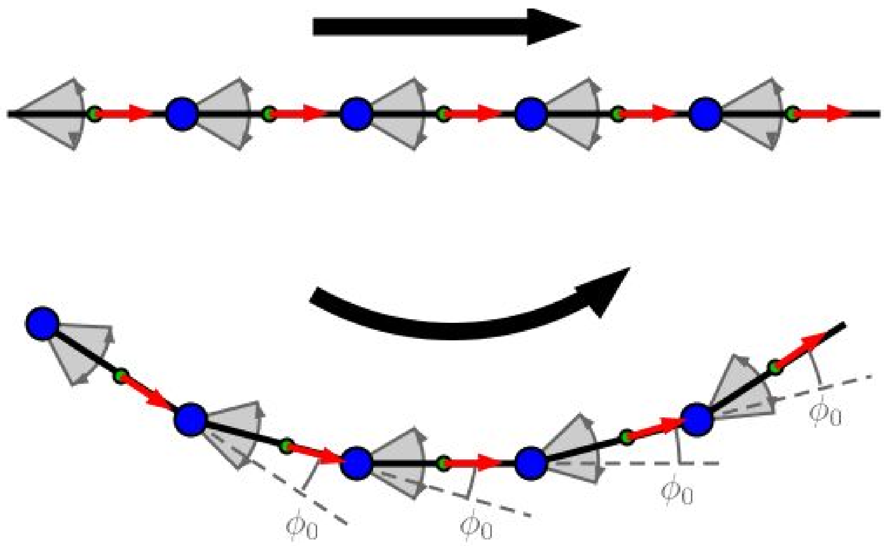
där $i \in 1, \ldots, N-1$ indexerar varje led, α amplituden på vinkeln leden ska pendla mellan, ω frekvensen på ormens rörelse och δ fasskillnaden mellan varje led.

3.3.3 Svängande rörelse för robotormen

I detta kapitel presenteras en enkel metod för att få en robotorm med serpentinrörelse att svänga. Ekvation 3.28 i föregående kapitel visade hur lederna ska röra sig för att få robotormen att röra sig framåt. Genom att modifiera ekvationen kan även en svängningsrörelse åstadkommas. Den slutgiltiga ekvationen för robotormens styrning blir

$$\phi_{ref,i} = \alpha \sin(\omega t + (i-1)\delta) + \phi_0, \tag{3.29}$$

där ϕ_0 är en vinkelförskjutning som sätts lika för alla leder. Hur detta leder till en roterande rörelse för robotormen illustreras i 3.7. Som nämnts i tidigare kapitel så ger varje länk en genomsnittlig drivande kraft längs den linje länken pendlar kring. När $\phi_0 = 0$, se övre exemplet i figur 3.7, så pendlar alla länkar kring den linje som roboten rör sig i och därmed bidrar till att röra roboten rakt fram. För $\phi_0 \neq 0$, se nedre exemplet i figur 3.7, så bildar länkarna en genomsnittlig kraft i en båge vilket får roboten att svänga. Detta betyder att ett större ϕ_0 får roboten att svänga mer.



Figur 3.7: Figur som illustrerar hur den drivande kraften uppstår och att kraften har samma riktning som den riktning varje led pendlar kring. Figuren visar även hur en svängande rörelse kan uppnås genom att ändra pendlingsriktningen, de vill säga ϕ_0, [1].

3.4 Simulering av matematisk modell

För att kunna undersöka egenskaperna hos den matematiska modell som beskrivs i kapitel 3.2 och för att analysera de rörelsemönster som tog fram i kapitel 3.3 så har modellen simulerats i en dator. Målet var att genom simuleringen av modellen kunna med tillräcklig noggrannhet undersöka beteendet hos en riktig robotorms. Simuleringen gör det möjligt att undersöka olika parametrar, tillstånd och styrsignaler med mera. Att kunna undersöka robotormen i datorn sparar dels tid men gör det också möjligt att undersöka robotormens tillstånd utan att behöva göra svåra mätningar på det riktiga systemet.

3.4.1 Implementering av matematisk modellen i MATLAB

Den matematiska modellen från kapitel 3.2 implementerades i MATLAB med hjälp av ekvationerna från D. De olika differentialekvationerna som förklaras mer i detalj i D.1.6, D.1.7 och D.1.8 lades in i MATLAB och implementerades som en funktion till MATLAB funktionen ode45. Det är framför allt modellen från D.1.8 som är relevant. Med hjälp av ode45 erhålls modellens tillstånd (hastigheter och position) som en funktion av tiden.

Det finns två huvudmodeller skapade i MATLAB. En *enkel* modell som kör de viktigaste funktionera och en *avancerad* modell som efterliknar att roboten har en avståndssensor längst fram i huvudet och på så sätt kan upptäcka objekt framför sig. Detta för att kunna simulera olika beteénden för att följa och undvika hinder.

Utöver differentialekvationsfunktionen finns fyra viktiga MATLAB skript: Ett huvudskript som sköter initiering och behandlar datan från ode45, en parameterfunktion som innehåller alla parametrar till körningen, en styrfunktion som sköter refferensvärdet till modellens leder ("motorer") och en funktionsfil som beräknar momentet på varje led under körningen. Utöver detta finns några extra skript som kan implementeras vid behov, till exempel för att beräkna friktionskrafterna på varje led eller skapa ett objekt som ormen ska följa.

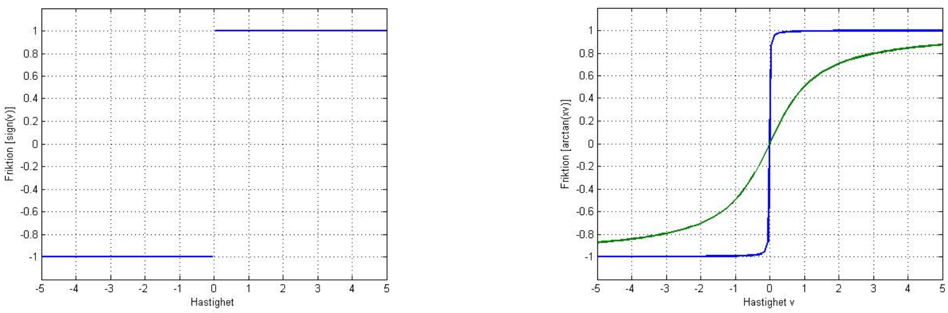
3.4.2 Modifiering av Coulumbs friktionsmodell

Friktionsmodellen som togs fram i kapitel 3.2.4 och ekvation 3.10 skapar problem vid simuleringen. *Sign* funktionen är diskontinuerlig vid hastigheten noll vilket ger problem vid implementeringen i ode45. Då det i modellen inte finns någon tröskelkraft för att få länkarna i rörelse och att de "starka" friktionskrafterna bildas redan vid mycket små hastigheter uppstår en obalans. Friktionskraften och hastigheten pendlar med motsatta tecken runt ett stillastående tillstånd.

Detta kan lösas genom att byta ut *Sign* funktionen mot en arctan funktion vilken är kontinuerlig över noll. Med en tillräckligt hög parameter i arctan funktionen erhålls en tillfredsställande och stabil modell, se figur 3.8. Den slutgiltiga friktionsmodellen för simuleringen blir

$$\boldsymbol{f}_R = \begin{bmatrix} \boldsymbol{f}_{R,x} \\ \boldsymbol{f}_{R,y} \end{bmatrix} = -\frac{2mg}{\pi} \begin{bmatrix} \mu_t \mathbf{C}_\theta & -\mu_n \mathbf{S}_\theta \\ \mu_t \mathbf{S}_\theta & \mu_n \mathbf{C}_\theta \end{bmatrix} \arctan\left(100 \begin{bmatrix} \mathbf{C}_\theta & \mathbf{S}_\theta \\ -\mathbf{S}_\theta & \mathbf{C}_\theta \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\mathbf{Y}} \end{bmatrix}\right) \in \mathbb{R}^{2N}. \quad (3.30)$$

Notera att $\frac{2}{\pi}$ läggs till för skala *arctan* funktionens asymptot till 1.



(a) $f = ksign(v)$ **(b)** $f = k\frac{2}{\pi}\arctan xv$ | blå: $x = 100$ | grön: $x = 1$

Figur 3.8: Figur som visar hur en diskontinuerlig *sign* funktion kan approximeras med en kontinuerlig arctan funktion

3.4.3 Styrsignal och styrsystem

När ode45 körs hämtar den samtidigt en en styrsignal från en extern funktion: styrsystemet. Denna funktion tar emot den nuvarande tiden och ledernas nuvarande vinkelposition, $\boldsymbol{\phi}$ och vinkelhastighet $\dot{\boldsymbol{\phi}}$. Funktionen returnerar styrsignaler för varje led, $\bar{\mathbf{u}}$.

Målet för styrsystemet är att få modellen att röra sig i en serpentinrörelse med hjälp av ekvation 3.29 från kapitel 3.3.3, alltså att modellens leder rör sig efter ϕ_{ref}. Detta kan enkelt åstadkommas genom PD-reglering vilket beskrivs i kapitel 4.8 i [1] enligt

$$\bar{\mathbf{u}} = k_p(\phi_{ref} - \phi) - k_d\dot{\phi} \tag{3.31}$$

där $k_p > 0$ och $k_d > 0$ är designparametrar. k_d parametern är i detta fallet till för att dämpa förstärkningen om vinkelhastigheten skulle bli för stark.

3.4.4 Möjliga analyser vid Simulering

Att simulera robotmodellen i MATLAB gör det möjligt att undersöka modellens egenskapar på ett snabbt och enkelt sätt. Modellen är implementerad i MATLAB så att alla parametrar enkelt kan ändras. Även antalet länkar går att ändra obegränsat. Parametrarna för referenssiganlen ϕ_{ref}, från ekvation 3.29, kan göras beroende av tiden vilket skapar möjlighet att implementera till exempel uppstartssekvenser där α och ω startar på 0 vid start eller möjlighet att skapa svängning genom att ändra på ϕ_0 vid bestämda tidpunkter.

Vid simulering plottas modellens tillstånd mot tiden vilket gör att till exempel hastigheten på robotens MC kan avläsas. Genom att använda resultatet från ode45 i momentfunktionen erhålls momentet på varje led under hela körningen vilket är viktigt för att förstå robotens begränsningar.

Genom att plotta modellens position med jämna korta tidsintervall kan en simulering skapas där robotens rörelse kan undersökas. Förutom att ge väldigt illustrativa och konkreta resultat gör simuleringen det även möjligt att undersöka hur väl modellen överensstämmer med verkligheten.

I den avancerade modellen som beskrevs i kapitel 3.4.1 visas även robotens synfält och vilka synbara objekt som finns. Ett enkelt styrsystem för att följa ett objekt har implementerats där ϕ_{ref} beror på riktningen och avstånd till objektet.

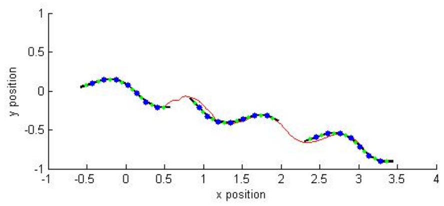
3.4.5 Slutsatser från olika testresultat

I detta kapitel redovisas resultat från simuleringarna. Dels för att testa egenskaper som friktionens betydelse för framåtrörelse och dels för att visa olika typer av rörelser. Alla tester genomfördes med nedan uppradade parametrar om inget annat anges och parametrarna har samma betydelse som i tidigare kapitel.

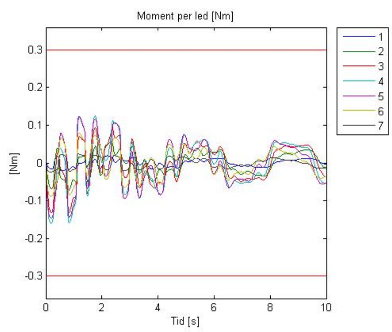
Tabell 4: Parametrar för tester av simulering

Parameter	värde	Enhet	Förklaring
$Time$	10	s	Tid på simulering
N	10		
m	0,03	kg	
l	0,08	m	
μ_t	0,1		
μ_n	0,9		
α	25°	$grader$	
ω	$0{,}3 * 360°$	$grader$	
δ	$\frac{360°}{N}$	$grader$	
ϕ_0	0	$grader$	
k_p	100		
k_d	1		

I figur 3.9a visas robotens rörelse för parametrarna ovan. Den röda linjen representerar huvudets position och hela roboten är plottad vid 0, 5 och 10 sekunder. I figur 3.9b visas momentet på varje led.



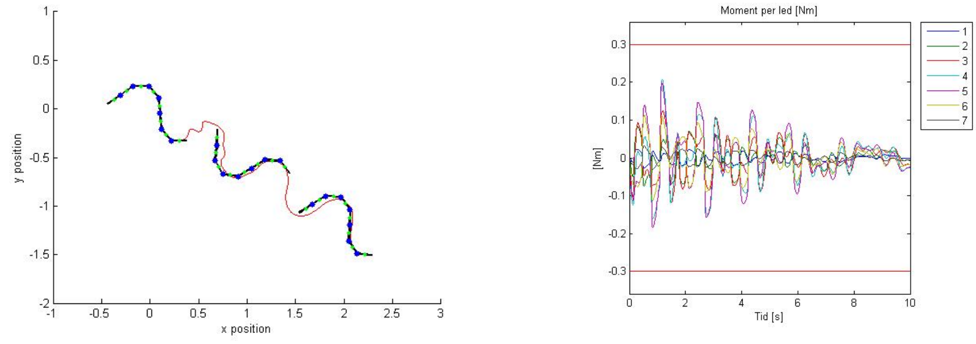
(a)



(b)

Figur 3.9: Simulering med de angivna parametrarna i table 4. Vänstra figuren visar robotens rörelse och den högra momentet på varje led

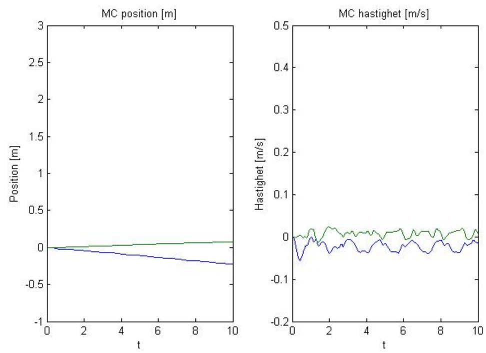
Figur 3.10 visar samma körning som i 3.9 men där α har ändrats till 50°



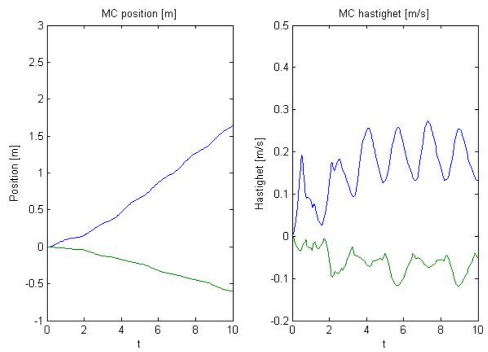
(a) (b)

Figur 3.10: Simulering som visar rörelse vid $\alpha = 50°$

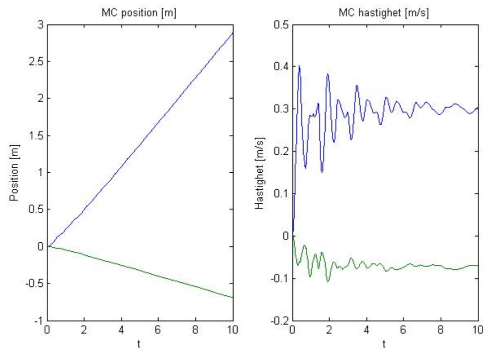
Nedan ges tre olika simuleringar där länkarnas sidofriktion, μ_n, ändras till 0,1 0,4 och 0,9. Här ses hur viktig friktionen är för att skapa en drivande kraft framåt. Med $\mu_n = 0{,}1$ är friktionen i båda leder samma och roboten står i stort sätt still. Den lilla hastighet den har får den i starten när roboten börjar slingra sig.



Figur 3.11: Robotens MC:s position och hastighet för $\mu_n = 0{,}1$. Blå linje är i x-led och grön i y-led

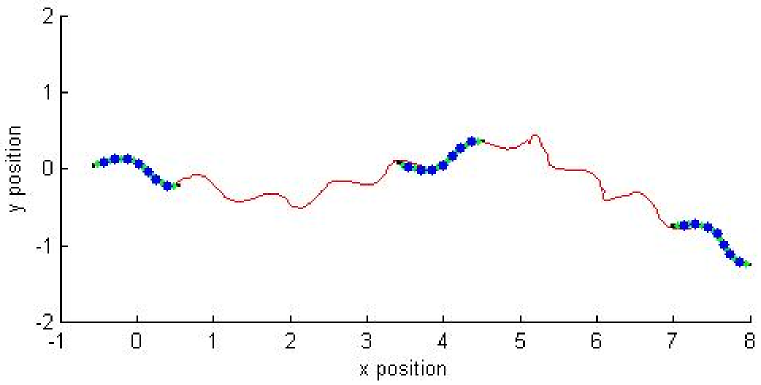


Figur 3.12: Robotens MC:s position och hastighet för $\mu_n = 0{,}4$. Blå linje är i x-led och grön i y-led



Figur 3.13: Robotens MC:s position och hastighet för $\mu_n = 0{,}9$. Blå linje är i x-led och grön i y-led

Figure 3.14 visar hur ϕ_0 kan få roboten att svänga. Tiden för simuleringen ändras till $Time = 30$ sekunder och när $5 < t < 10$ sätts $\phi_0 = -3°$ och vid $17 < t < 22$ sätts $\phi_0 = 5°$.

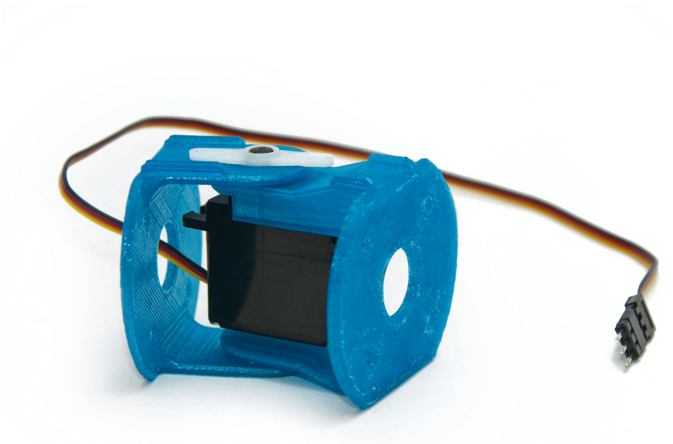


Figur 3.14: Visar hur svängande rörelse skapas med hjälp av ϕ_0

4 Konstruktion av fysisk robot

Det vanligaste sättet att med en robot efterlikna ormars rörelsemönster är att dela in roboten i flera likadana moduler, [1]. Ormliknande robotar konstruerade i moduler kan dessutom ha mycket intressanta funktioner som självmontering och självmodifikation beroende på uppgiften som den ställs inför. Det är även lätt att förlänga roboten genom att lägga till fler moduler.

Med en modul menas en enhet där den vridning som önskas sker. Beroende på komplexiteten på modulen kan den vridas i upp till tre frihetsgrader, upp/ner, höger/vänster och rotation. Hur en enaxlig modul med en frihetsgrad kan se ut kan ses i figur 4.1. Denna består av en motor inkapslad av två delar formade som “U:n”.



Figur 4.1: Bild på hur en enaxlig modul monterad kan se ut

Detta avsnitt tar upp hur arbetet med att designa och konstruera roboten gick till. Områden som behandlas är:

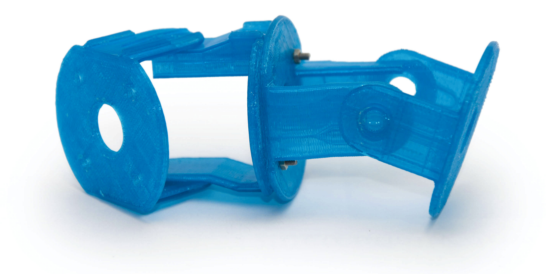
- Robotens skelett
- Friktion mellan robot och underlag
- Motorer
- Styrning
- Strömförsörjning

4.1 Robotens skelett

Robotens skelett är uppbyggt av moduler som designades med hjälp av CAD-programmet CATIA, och utskrivet med en 3D-skrivare i form av en Makerbot Replicator 2X från Makerbot. Eftersom detta var en ny produkt designad från grunden krävdes ett par iterationer av designen för att ge modulerna önskad funktionalitet (de första två moduldesignerna finns att läsa om i Appendix A). För att ge roboten en helhetsform som efterliknar en orm konstruerades två extra moduler, en svansmodul och en huvudmodul. Dessa moduler har funktionen att de kan innehålla de delar som inte får plats i de övriga modulerna.

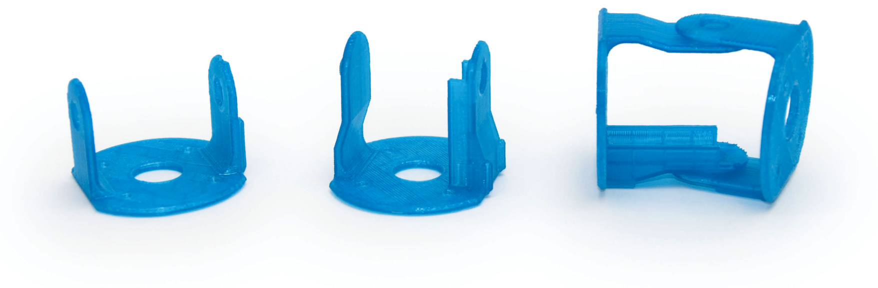
4.1.1 Robotens kropp

Då moduldesignen inte antogs påverka robotens rörelseegenskaper i någon större utsträckning valdes en en-axlig U-formsdesign som var relativt enkel att rita i CAD. Skulle en två-axlig robot dock önskas kan de en-axliga modulerna monteras med 90° vridning mellan varandra vilket skapar en två-axlig robot, se figur 4.2. Dock skulle detta innebära en lägre upplösning i robotens röresle då rotationsaxlarna i respektive dimension hamnar dubbelt så långt ifrån varandra.



Figur 4.2: Visualisering av 90 graders vridning mellan två moduler

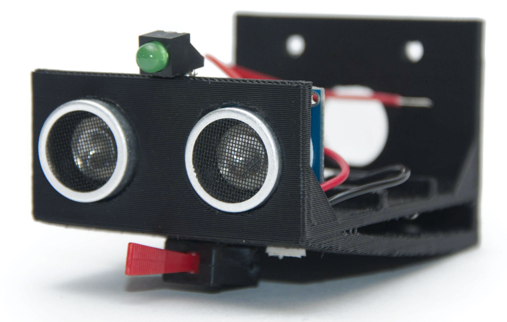
Storleken på designen baserades på storleken på servomotorerna som fanns att tillgå från projektets start eftersom dessa skulle placeras inuti modulerna. För att dra diverse kablar till servomotorerna genom modulerna konstruerades de med ett hål i mitten av modulen. De två delar som bildar modulen skiljer sig åt på några punkter. Den del motorn ska sitta fast i har stödväggar monterade vilka bildar ett spår som håller motorn på plats. På liknande sätt fick den andra halvan en nedfasning som en av de medföljande armarna till motorn kan placeras i, se figur 4.3. Prototypen har en godstjocklek på 2,5 mm och på modulens undersida sitter totalt fyra skenor som ska användas till att utvärdera hur väl skenor fungerar som friktionsunderlag (se kapitel 4.2.1).



Figur 4.3: Modul med de olika delarna isär och ihopmonterade

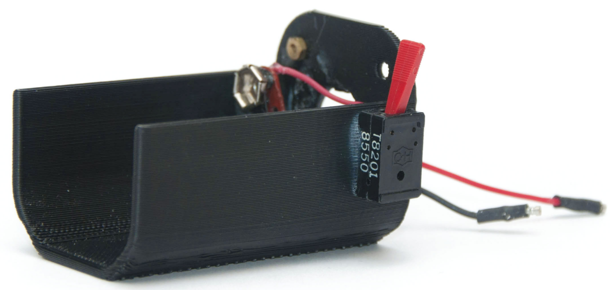
4.1.2 Robotens huvud- och svansmodul

Huvudmodulen gavs inte någon motor eller möjlighet att vinklas. Modulen är designad för att kunna innehålla en avståndssensor för syn, ett kretskort för att styra roboten samt batteri för att driva dessa. Som syns i figur 4.4 är fronten på huvudet anpassat till avståndssensorn med hål. Huvudet har också fått en strömbrytare monterad (under sensorn) för att enkelt kunna slå av och på ström till sensor och kretskort.



Figur 4.4: Bild av huvudmodulen med sensor monterad

Svansmodulen gavs precis som huvudmodulen ingen funktionalitet i avseende på slingerrörelsen utan utnyttjas istället för att rymma batterier till motorerna. Batterierna placerades i en batterihållare som ligger i modulen. En strömbrytare monterades på sidan för att enkelt kunna slå på och av motorerna.



Figur 4.5: Bild av svansmodulen

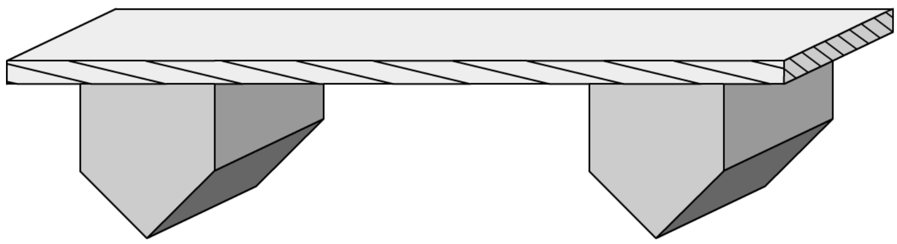
4.2 Friktion mellan robot och underlag

Som nämnts i kapitel 2 utnyttjar ormar att dess friktion framåt är lägre än den i sidled. För att en robotorm ska kunna efterlikna en riktig orms rörelsemönster måste även friktionsmönstret efterliknas. Tre olika lösningsförslag togs fram och utvärderades:

- Skenor
- Gummiplattor
- Hjul

4.2.1 Skenor

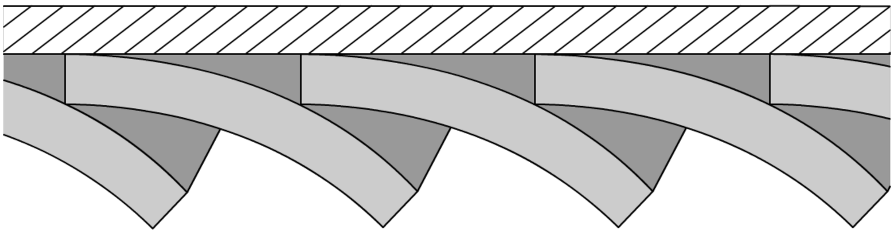
Långa skenor av ett hårt material kan placeras längs med undersidan på roboten och fungera som ett par skridskor. De skulle kunna vara av samma material som modulen i övrigt. Hårda material har dock inte lika hög friktion mot glatta underlag som till exempel gummi har. Detta kan leda till låg friktion i alla riktningar för skenorna vilket kan göra denna lösning bristfällig. Ett förslag på hur skenor kan se ut visas i figur 4.6.



Figur 4.6: Exempel på utformning av skenor

4.2.2 Gummiplattor

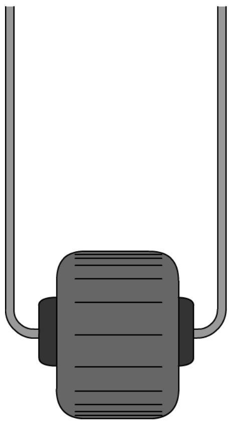
Gummi har generellt sätt hög friktion mot en mängd olika underlag. Genom att sammanfoga flera smala remsor omlott till en trappstegsform enligt figur 4.7 kan denna ge upphov till olika friktionskoefficienter i olika riktningar. I rikting längs med trappstegen är friktionen lägre och i riktning vinkelrätt mot dem är friktionen högre. Denna lösning är den som bäst efterliknar riktiga ormars skinn.



Figur 4.7: Exempel på utformning av gummiplattor

4.2.3 Hjul

I framåtriktning när hjulet rullar är friktionskrafterna mycket små medan det i sidled ger mycket hög friktion. Dock har hjul mycket liten friktion bakåt, till skillnad från ormars fjäll som har relativt stor friktion bakåt, men denna friktion är inte nödvändig för att få roboten att röra sig framåt i en serpentinrörelse om friktionen i sidled är tillräckligt stor. Att använda hjul kräver heller ingen förändring av moduldesignen. Hjulen kan klämmas eller limmas fast mellan två moduler alternativt fästas på modulernas undersida. Ett förslag på hjul visas i figur 4.8.

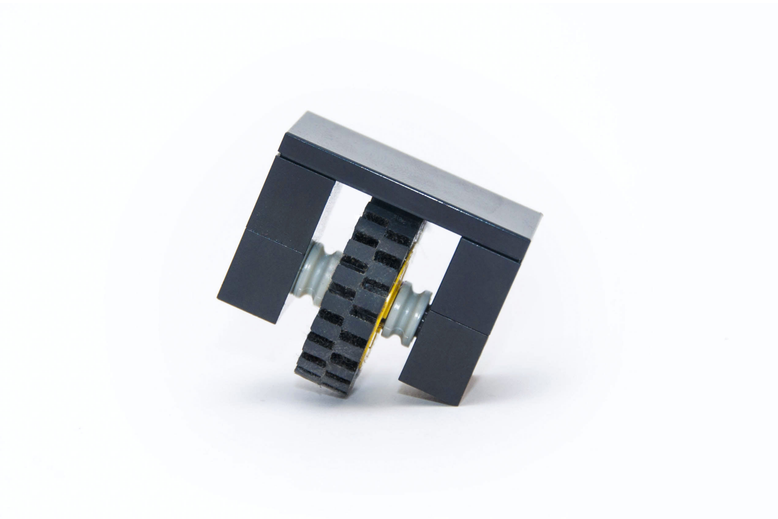


Figur 4.8: Exempel på utformning av hjul

4.2.4 Val av friktionslösning

Den enklaste och effektivaste lösningen för att få framåtdrift på plant underlag är att använda hjul. Vad gäller att utseendemässigt efterlika en verklig orm så är det dock inte önskvärt att använda hjul, men då funktionalitet prioriterades över estetik implementerades en hjullösning.

Friktion mellan roboten och underlaget skapas genom att använda LEGO-hjul, se figur 4.9. Dessa är lagom stora, enkla att montera och har däck av gummi vilket ger tillräcklig friktion mot marken. LEGO-hjulen fästes mot undersidan av kroppen.

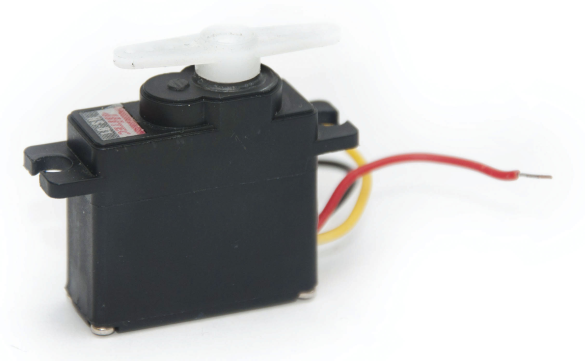


Figur 4.9: Hjullösning

4.3 Motorer

Servomotorerna, placerade i varsin modul, genererar en vinkeländring mellan modulerna, dessa visas i figur 4.10. Varje motor har tre kablar; en för ström, en för jord och en för styrning. Motorerna matas med 4,8 - 6,0 V och drar vid drift men utan last, lite över 200 mA per motor, [6].

Styrsignalerna till motorerna genereras från kretskortet i form av PWM-signaler (Pulse Width Modulation) där pulslängden styr motorns vinkel. Pulserna till motorerna ska vara mellan 600 och 2400 μs med ett intervall på 20 ms mellan varje puls.



Figur 4.10: De motorer som användes

4.4 Strömförsörjning

Då motorerna i roboten kräver 4,8 - 6,0 V drivs de av fyra seriekopplade AA-batterier på 1,2 V vardera vilket ger en spänning på totalt 4,8 V. Motorerna parallellkopplades till en huvudström- och en huvudjordkabel som löper genom hela roboten för att spara både plats och material jämfört med att ha individuella kablar för varje enskild motor. För att ha möjligheten att kunna variera antalet moduler monterades T-kontakter för att kunna koppla av och på motorerna till huvudström- och jordsladden.

Då belastningen på servomotererna varierar kommer dessa också dra varierad ström. Då strömmarna är som högst sker spänningsfall i kretsen på grund av resistansen i motorernas strömkablar och inre resistans i batterierna. Då dessa spänningsfall fick styrenheten att startas om så har kortet en separat spänningskälla. Samtliga komponenter på styrkortet kräver en spänning på 1,5 - 5,5 V, dessa matas med två seriekopplade knappcellsbatterier på 3.0 V styck vilket ger en tomgångsspänning på totalt 6.0 V. Under drift sjunker spänningen på batteriet till ca 5,5 V, en spänning vilken ligger inom alla komponenternas krav på matningsspänning.

För att enkelt kunna slå av och på roboten används två strömbrytare, ett till vardera batteripack.

4.5 Styrning

För att styra roboten designades och programmerades ett styrkort vilket genererar styrsignaler till motorerna. Med en avståndssensor monterat på huvudet får styrkortet information om det finns objekt framför roboten, samt avståndet till objektet. Detta för att kunna reglera pulserna till motorerna så att roboten kan följa efter rörliga objekt samt undvika kollision. Det finns även en lysdiod monterad för möjligheten att ge visuell feedback under drift då sensorn registerar ett objekt.

4.5.1 Styrkort

Styrkortet innehåller två identiska mikrokontroller, Atmega328p. Två dekoders, M74HC238, med åtta utgångar monterades på kortet med syftet att skapa fler utportar dit servomotorer kan anslutas. Dekodern fördelar signalen från en av mikrokontrollerns utgångar över flera av dekoderns utportar med hjälp av addressering. Tre adressbitar styr vilken av de åtta utgångar på dekodern som ska vara aktiv och motta signalen från mikrokontrollerns utgång. Detta innebär att med två dekodrar kan 16 motorer anslutas till en mikrokontroller vilket endast använder fem av mikrokontrollerns utgångar. Kopplingen med processor och två dekodrar kan dock endast skicka pulser till två motorer samtidigt. Därför krävs en seriell utmatning av pulser för att få fullständiga PWM-signaler till alla motorer.

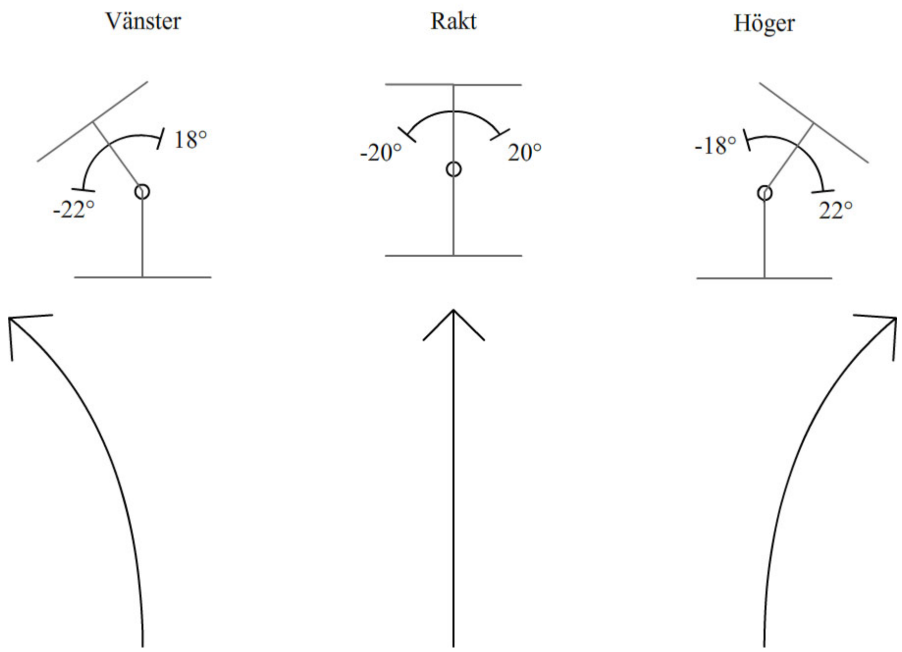
De båda dekodrarna är anslutna till den ena mikrokontrollern och den andra kontrollern är kopplad till en avståndssensor och en lysdiod. De två mikrokontrollerna är ihopkopplade med två signalkablar för seriekommunikation. Två kretsscheman, ett för varje processor finns i appendix C.

4.5.2 Avståndssensor

För att ge roboten möjlighet att upptäcka hinder i färdriktningen monterades en ultraljudssensor av typ HC-SR04, i huvudmodulen. Sensorn har inte bara möjlighet att avläsa om det finns ett föremål framför roboten utan också avståndet till detta så länge det befinner sig inom sensorns synfält, cirka 30°. Maxavståndet som sensorn kan avläsa är 4 m och det minsta avståndet är 2 cm med en mätosäkerhet på 3 mm. Sensorn har fyra anslutningar, två till strömmatning (4,5 - 5,5 V) och jord, en till att förse sensorn med en styrsignal (trig) som startar en avståndsmätning och en anslutning som returnerar längden på det uppmätta avståndet (echo) genom att skicka ut en puls vars längd är proportionell mot det uppmätta avståndet.

4.5.3 Styrteknik

För att ändra färdrikting på roboten adderas en konstant (se kap 3.3.3), en extra vinkel till alla motorer vilket resulterar i ett förskjutet intervall. Det förskjutna intervallet gör att roboten blir svängd vilket leder till en riktningsändring under framåtdrift. Figur 4.11 nedan illustreras vad som händer med vinkelintervallet när 2° dras ifrån, respektive läggs till på varje modul.



Figur 4.11: Illustration över hur roboten fås att svänga

4.6 Programmering

Mjukvaran utvecklades allteftersom roboten förändrades och programmen utvärderades kontinuerligt. Det första programmet styrde endast motorerna så att roboten rörde sig i en slingerrörelse, det andra programmet uppbyggt av timrar och avbrott medförde att sensorn kunde kontrolleras samtidigt som motorerna sändes pulser och det tredje, slutgiltiga programmet utnyttjade två processorer med seriekommunikation emellan.

Processorerna programmerades med hjälp av en Arduino. Programmet är skrivet i C och styrsignalerna baseras på de simuleringar som genomförts.

4.6.1 Kod med fördröjningar

Programmet matar servomotorerna med vinklar från jämnt fördelade punkter på en sinuskurva som förskjuts med tiden vilket får roboten att röra sig med en jämn slingerrörelse. För att skicka ut önskad längd på pulserna till motorerna används fördröjningar vilket innebär att programmet väntar en viss tid innan det fortsätter. Genom att varje dekoder skickar ut pulser till hälften av motorerna är det möjligt att använda samma adressignaler till de båda dekodrarna. Därmed skickas en puls till två motorer samtidigt, en från varje dekoder. Efter varje puls processorn skickar ut ändras adressen och de två nästkommande motorerna får signal. Då alla motorer fått en puls väntar programmet till dess att 20 ms passerat.

4.6.2 Timerkod

Genom att använda timrar behöver fördröjningar inte användas och processorn kan därmed användas på ett effektivare sätt. En timer kan generera avbrott och när ett avbrott sker pausas den vanliga exekveringen för ett anrop av en avbrottsrutin.

Varje dekoder står för utskick av pulser till hälften av motorerna vilket innebär att samma adress till båda dekodrarna kan användas. För att få olika längd på pulserna till ett motorpar användes två timrar. Timrarna startas med respektive pulslängd och avbrottsrutinen för vardera timer avbryter pulsen som börjar samtidigt som timrarna startar. Nästa steg är att skicka ut pulser till nästa motorpar, detta görs genom att öka adressen och starta pulserna samt timrarna med andra värden på nytt.

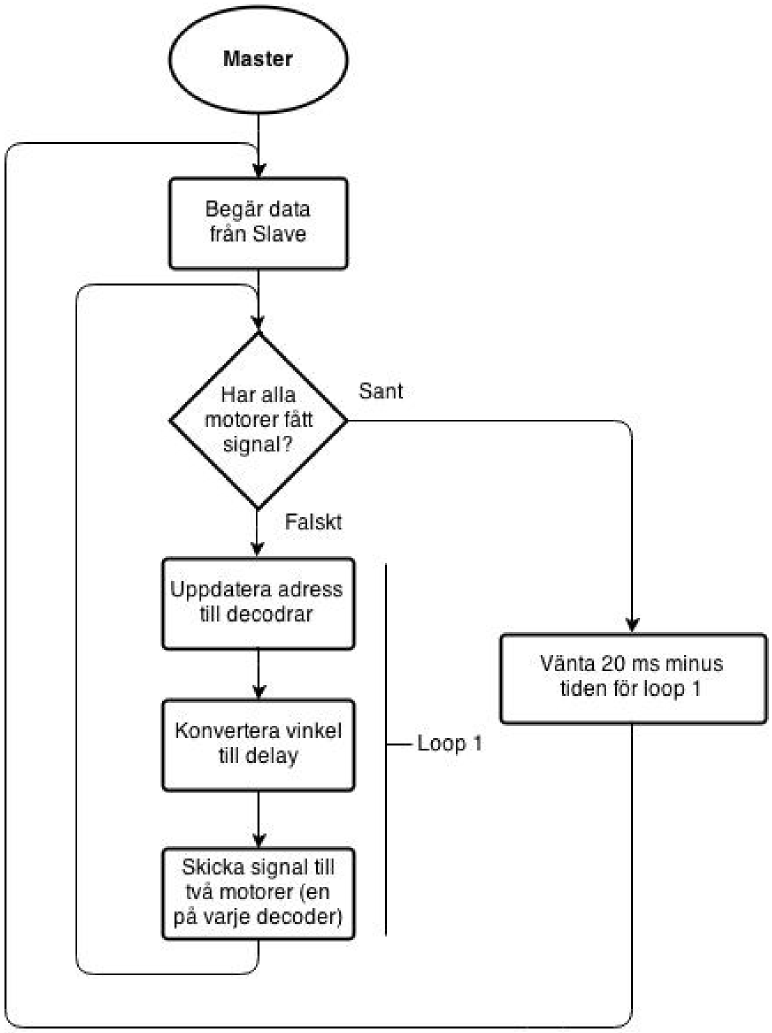
En tredje timer ser till att motorerna inte får signal för ofta. Denna timer startas och körs med en kontinuerlig frekvens. Då denna timer får avbrott startas utmatningen av pulser till motorerna på nytt.

Sensorn använder externa avbrott vilka går in i en metod om signalen på ett förutbestämt ben ändras. För att kunna avläsa hur lång pulsen som sensorn skickar ut, användes detta avbrott på echo, se 4.5.2. Programmet får ett avbrott som startar en tidräkning då en puls börjar skickas ut från sensorn och ett avbrott då pulsen avslutas. På så sätt kan längden av pulsen sensorn skickar ut mätas och avståndet till föremål kan bestämmas. En variabel som beror på avståndet bestäms och adderas till pulserna till motorerna vilket får roboten att svänga.

4.6.3 Kod för två processorer

Mjukvara för två processorer har tagits fram. De två processorerna är uppdelade i en slave som sköter sensor och styrning och en master som endast står för utskick av pulser till motorerna samt konvertering av vinklar till pulslängder.

Motorprogrammet (master): Syftet med motorprogrammet är att med hjälp av dekodrarna skicka ut individuella PWM-signaler till alla motorer. Pulsernas längd styrs av vektorn med referensvinklar som tas fram av slavenheten.

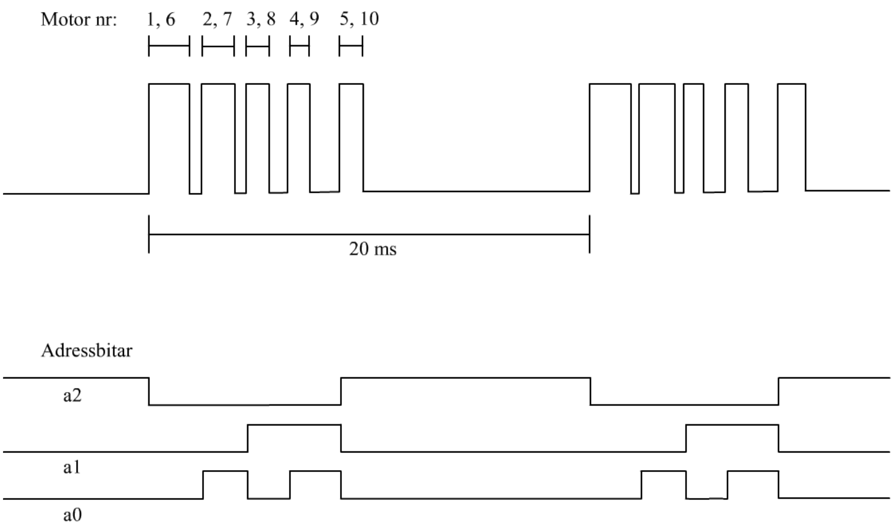


Figur 4.12: Flödesschema för Motorprocessorn

Koden byggs upp av en loop, vilken inleds med en begäran av vinkelreferenser från slavenheten och följs av utmatning av en puls till varje motor. Begäran stimulerar slavenheten till avbrott som direkt sänder över data. Då servomotorernas specifikation listar att de ska få PWM-signal med pulser var 20:e ms ska cykeltiden för denna loop alltid vara just 20 ms, flödesschema för programmet visas i figur 4.12.

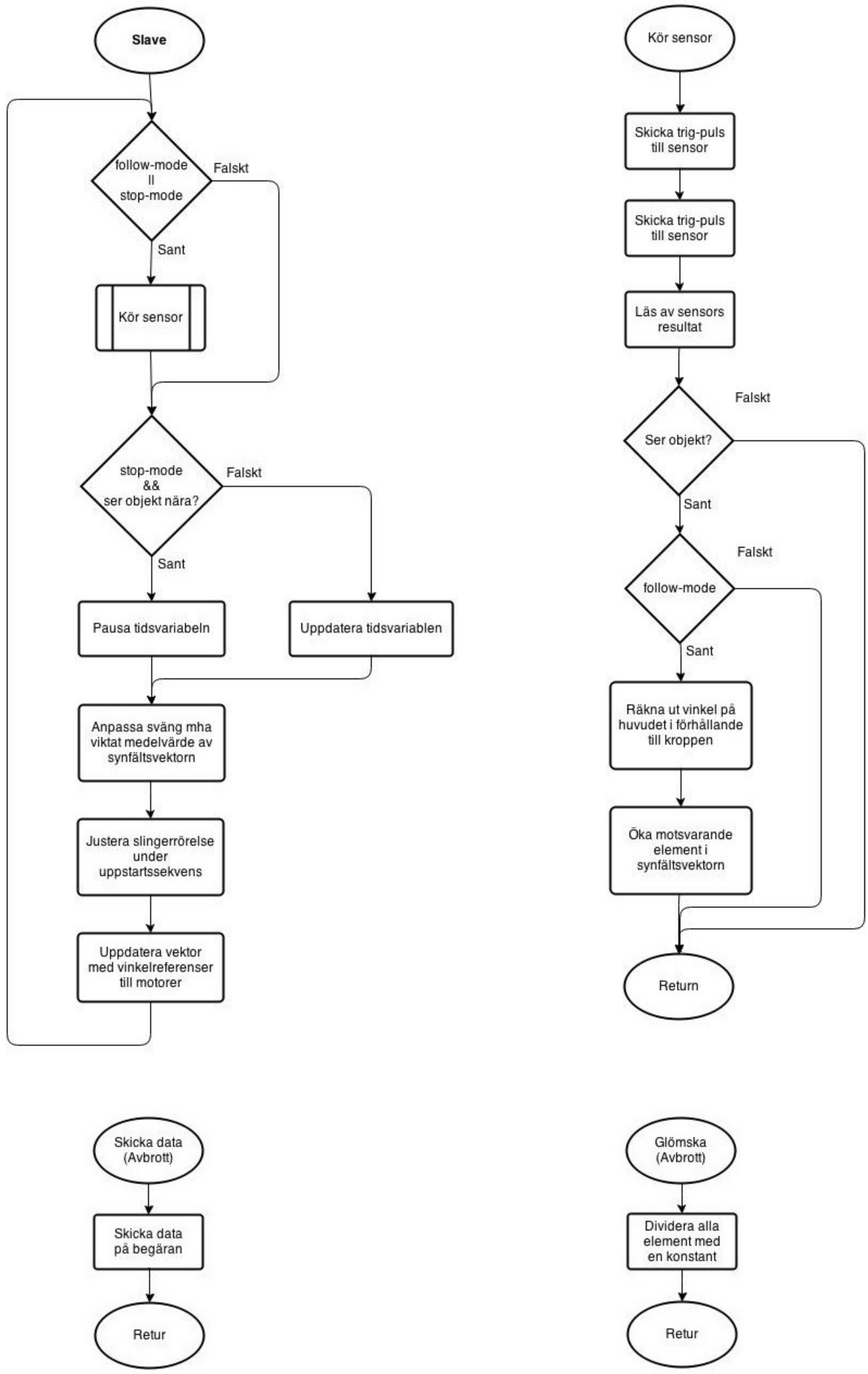
Med hjälp av en inre loop, där antalet varv styrs av antalet moduler i roboten dividerat med två, kan pulser skickas till motorerna två och två genom var sin dekoder. Adress skickas till dekodrarna för att aktivera rätt par motorer varpå pulser skickas med längder uträknade från vinkelreferenserna.

När pulser är skickade till varje motor väntar programmet för att uppnå cykeltiden på 20 ms, varpå en ny begäran skickas till slavenheten och nya pulser skickas ut till motorerna. I figur 4.13 visas signal till den ena dekodern (överst) tillsammans adressbitarna.



Figur 4.13: Stillbild över signalerna till dekodrarna

Styrprogrammet (slave): Syftet med styrprogrammet är att läsa av omgivningen med sensorn och ta fram referenser för att styra motorerna. Programmet är designat för att roboten ska kunna följa efter objekt inom synfältet, där synfältets längd specifieras av en konstant i mjukvaran och dess vidd av hur mycket huvudet vrids när roboten slingrar sig. Eftersom roboten hela tiden ändrar riktning som den tittar åt krävs ett synminne för en bra objektföljning. Programmet har därför en minnesfunktion med en glömskefaktor implementerad. Glömskefaktorn medför att nya objekt ger störst påverkan på styrningen. Ett flödesschema över hela programmet finns i figur 4.14.



Figur 4.14: Flödesschema för Styrprocessorn

När motorprogrammet begär data kommer ett avbrott att ske. Den vanliga exekveringen kommer att avbrytas för att skicka över data innan programmet återgår till där det var innan det avbröts.

Två olika styrlägen som bestämmer robotens beteende kan individuellt aktiveras under programmeringstadiet, *stop-mode* och *follow-mode*. Med stop-mode aktiverat stannar roboten för hinder som är inom ett bestämt avstånd medan follow-mode är till för att följa efter objekt som är inom ett förbestämt avstånd.

Programmet byggs upp av tre huvudvariabler som kodens funktionalitet bygger på. En tidsvariabel som kan uppdateras efter realtid eller pausas vilket ger möjligheten att stanna roboten; en vektor som symboliserar robotens synfält, där varje element beskriver vad den ser i en viss vinkel; samt en vektor med element som vart och ett beskriver en viss motors referensvinkel.

Tidsvariabeln uppdateras hela tiden med realtiden. Ifall kriteriet om att roboten ser ett objekt som är inom gränsen för stop-mode kommer tiden inte att uppdateras utan pausas tills föremålet är borta. Detta gör att vektorn med vinkelreferenser inte uppdateras och roboten står stilla.

Synfältsvektorn har egenskapen att alla dess element divideras med en konstant i periodiska avbrott, vilket medför den tidigare nämnda glömskefaktorn. Elementen kan inte överstiga ett bestämt maxvärde och inte eller minska under ett minvärde.

Om stop-mode eller follow-mode är aktiverat avläser sensorn omgivningen var 15 ms. Om follow-mode är aktiverat och sensor ser ett objekt inom gränsen för follow-mode, beräknas vinkeln på robotens huvud i förhållande till dess kropp, alltså den riktning roboten tittar åt, med hjälp av vektorn med motorernas referenser. Genom att öka motsvarande element i synfältsvektorn så sparas objektets placering i minnet. Synfältsvektorns element vägs därefter ihop för att beräkna ϕ_0 som får roboten att svänga.

Vektorn med vinkelreferenser skapas med hjälp av sinusfunktionen 3.29. Amplituden, α i sinusfunktionen är, förutom under en uppstartssekvens, konstant. Under uppstarten ökar amplituden och vinkelfrekvensen, ω långsamt för att få en mjukare uppstartsrörelse.

5 Det färdiga konceptet

Den färdiga roboten består av tio moduler samt en huvud- och en svansmodul.

Modulerna hålls ihop med hjälp av servomotorerna och varje länk är ihopsatt med skruvar. Detta samt att alla motorer är ihopkopplade med T-kontakter gör det enkelt att montera och demontera roboten.

5.1 Tester

Roboten har efter färdigställandet fått genomgå ett antal tester för att utvärdera hur bra den kan röra sig framåt. Testerna ligger också till grund för viss modifiering av framförallt koden, som nämnts ovan. Samtliga tester genomfördes på ett plant och jämnt golv i en 3 m bred korridor.

5.1.1 Kalibrering

Då motorerna inte har kunnat placeras helt perfekt i mittenläget, vilket innebär att roboten får en vridning, i detta fall åt vänster, genom kroppen när den är inställd på rak. På grund av detta har en kompensering hårdkodats in som får roboten att vrida sig mer åt höger för att få den att gå rakare.

5.1.2 Kursavvikelse

För att få en uppskattning av robotens kursavvikelse mättes avvikelsen i sidled vid körning. Roboten startades och sattes sedan i position varpå den började röra sig framåt. Hur mycket den avvek från startpositionen i sidled mättes då roboten rört sig 2 och 5 m framåt. Vid oreglerad körning, det vill säga att roboten inte gjorde någon korrektion av riktningen med hjälp av data från sensorn erhölls följande resultat.

Tabell 5: Avvikelse utan reglering

Test	Avvikelse 2 m	Avvikelse 5 m
1	0,50 m	1,50 m
2	1,50 m	-
3	0,30 m	-
4	0,15 m	-
5	1,50 m	-

Tabell 6: Medelvärde och standardavvikelse

	Standardavvikelse σ	Medelvärde
2 m	0,66	0,79
5 m	-	1,50

Slutsatserna av testet vid oreglerad körning var att det är mycket svårt/omöjligt att få roboten att köra i en korridor eftersom den avvek kraftigt från kursen och krockade med väggen innan 5 m hade nåtts.

För att testa hur väl follow-mode fungerade, d.v.s hur väl roboten kunde följa efter saker, placerades ett objekt framför roboten mitt i korridoren. Roboten startades och då den rörde sig framåt fördes samtidigt objektet framåt för att roboten inte skulle krocka med detta. Avståndet roboten var inställd att följa efter var 60 cm för att den inte skulle se och svänga mot väggarna. Vid reglering genom efterföljning av ett objekt erhölls följande värden.

Tabell 7: Avvikelse med reglering

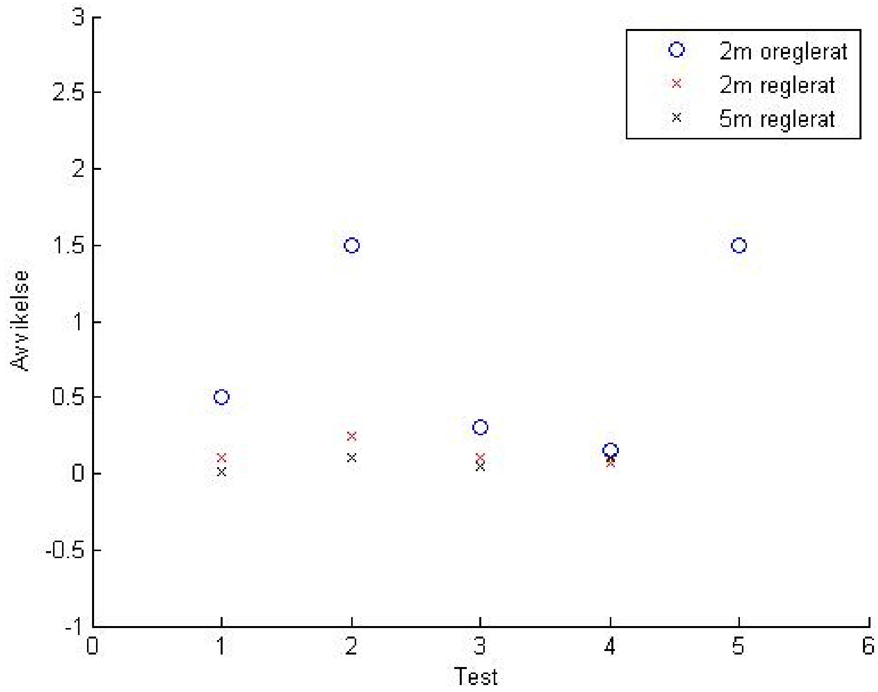
Test	Avvikelse 2 m	Avvikelse 5 m
1	0,10 m	0,01 m
2	1,50 m	-
3	0,25 m	0,10 m
4	0,10 m	0,05 m
5	0,07 m	0,10 m

Tabell 8: Medelvärde och standardavvikelse

	Standardavvikelse* σ	Medelvärde*
2 m	0,08	0,13
5 m	0,044	0,065

*Dessa värden avser alla tester utom test 2

Testet visade på stor exakthet så länge roboten inte tappade objektet den försökte följa (se test 2 i tabell 7). Reglering på detta sätt gav mycket liten avvikelse från kursen. Figur 5.1 visar hur mycket roboten avvek från kursen i samtliga tester.



Figur 5.1: Spridningen från tester

5.1.3 Hastighet

Ett mått på prestanda är ofta hastighet. För att mäta robotens hastighet mättes tiden det tog för roboten att åka 10 m i en korridor med follow-mode. α är amplituden på robotens sinusform och ω bestämmer hastigheten på rörelsen.

Tabell 9: Hastighetstest

Test	α	ω	Tid 10 m
1	25	0,5	85
2	25	0,5	95
3	25	0,7	45
4	25	0,7	50
5	25	1,0	40
6	25	1,0	42
7	25	1,0	45
8	25	1,2	62
9	25	1,2	68

Tabell 10: Medelvärde och standardavvikelse

ω	Standardavvikelse σ	Medelvärde
0,5	7,1	90,0
0,7	3,5	47,5
1,0	2,5	42,3
1,2	4,2	65,0

Från resultaten kan man utläsa att hastigheten framåt ökade med hastigheten på rörelsen fram tills ett visst värde på ω.

6 Framtida arbete

Även om robotormar idag kan vara väldigt avancerade har detta projekt varit en bra start på arbetet att efterlikna dessa. Dock finns det fortfarande mycket kvar som kan vidareutvecklas i framtida projekt vilket redogörs för nedan.

6.1 Robot utan hjul

I nuläget är roboten beroende av hjul för att ta sig framåt. Det hade varit önskvärt att ha en robotorm utan detta beroende. Både då roboten hade blivit mer estetiskt tilltalande eftersom den hade blivit mer lik en riktig orm men det hade också kunnat innebära ökad framkomlighet vid ojämna ytor. Det som kan göras är att vidare undersöka hur egenskaperna hos fjällen på ormar kan efterliknas och implementeras i en robotorm.

6.2 Rörlighet i tre dimensioner

Vad gäller körning i ojämn terräng är möjligheten att röra sig i mer än två dimensioner ett måste, då det annars är omöjligt att ta sig över hinder. Detta kan lösas på åtminstone ett par sätt. Det enklaste sättet är att växelvis vrida modulerna 90° mot varandra vilket robotens moduldesign tillåter. Detta innebär att varannan modul tillåter vridning i en led medan resterande moduler tillåter vridning 90° vinklat från denna vilket ger roboten möjlighet att röra sig i tre dimensioner. Alternativet är att designa en ny, mer komplex modul som har axlar vilka möjliggör vridning i tre dimensioner.

6.3 Fler sensorer

En av robotens nackdelar är att den inte är fullt medveten om sin omgivning utan endast i en vinkel rakt fram. Medvetenheten kan dock ökas med hjälp av fler/bättre sensorer som kan urskilja fasta objekt från rörliga. Detta skulle kunna förbättra funktionen follow-mode då roboten skulle kunna programmeras att inte påverkas av fasta objekt såsom väggar när den ska följa efter ett objekt. Fasta objekt kan i nuläget få roboten att köra mot objektet tills den kommer tillräckligt nära och sedan stanna utan möjlighet att ta sig därifrån.

6.4 Radiostyrd

I inledningen till denna rapport nämns att robotormar effektivt skulle kunna ta sig fram i bland annat husras med syfte att leta efter överlevande. För att detta skulle ske på ett säkert och effektivt sätt skulle möjligheten till någon form av operatörsbaserad styrning/kontroll vara att föredra. I nuläget fattar roboten alla beslut vad gäller färdriktning på egen hand vilket inte är optimalt i alla lägen.

6.5 Ökad funktionalitet med hjälp av mjukvara

Ett smidigt sätt att ytterligare öka robotens framkomlighet är att med mjukvara ge roboten fler funktioner såsom rörelsemönster och undvikning av hinder. Som står beskrivet i kapitel 2 är inte serpentinrörelsen alltid det mest effektiva rörelsemönstret. Möjligheten att kunna anpassa sig efter terrängen hade kunnat öka robotens framkomlighet markant. Dock kan ett annat rörelsemönster kräva att roboten behöver kunna röra sig i fler än två dimensioner vilket i dagsläget inte är möjligt. Den andra funktionen, undvikning av hinder, borde inte vara speciellt avancerad att implementera då den till hög grad bygger på den befintliga follow-mode-koden. Även förmågan att kunna backa borde gå att enkelt kunna skapa med hjälp av befintlig kod som driver roboten framåt. Roboten skulle då till exempel kunna backa och svänga undan från hinder ifall den har stått stilla för länge.

7 Diskussion

Nedan beskrivs de tankegångar som pågick då projektet stod inför beslut som eventuellt hade kunnat ge ett annorlunda slutresultat om de fattats annorlunda. Både för- och nackdelar för tagna beslut men också alternativ diskuteras. Även problem som uppstod under projektets gång redogörs.

7.1 Alternativa produktionssätt

I ett tidigt skede krävdes val av vilket material och tillverkningsmetod som skulle användas till modulerna för att komma vidare med projektet. Att det här valdes att utnyttja de 3D-skrivare som finns på Chalmers fanns det flera anledningar till. Först och främst går det snabbt och smidigt att producera många moduler på kort tid. Att använda en 3D-skrivare garanterar också att alla moduler blir likadana. Alternativet till 3D-utskrivna moduler var att bocka aluminiumplåtar till önskad form. Aluminiumplåtar är dock svåra att få identiska när de görs för hand och tar de längre tid att skapa sett till arbetstimmar då en 3D-skrivare sköter sig själv när det väl är igång. Den primära orsaken till att en alternativ lösning togs fram var för den händelse att plasten som 3D-skrivaren använder inte skulle vara tillräckligt hållbar. Därför skrevs bara en modul ut till att börja med. Efter mindre tester på denna modul ansågs det dock att plasten utan problem skulle vara tillräckligt stark för att användas i modulerna varpå en 3D-utskriven lösning valdes för det fortsatta arbetet.

Det finns två skrivare att tillgå på Chalmers, en Makerbot Replicator 2X från Makerbot och en Objet24 från Stratasys. Makerboten använder en termoplast som hettas upp och appliceras lager på lager, [3] medan Objetten använder en fotoplast som belyses med UV-ljus för att påbörja härdning, [4] med liknande lager på lager princip. Dessa båda skrivare har skillnader som måste tas i beaktning vid design av produkten. Objetten har en klar fördel när det kommer till upplösning och noggrannhet eftersom den inte behöver förlita sig på ett jämnt flöde för att plasten inte ska överhetta eller bli för sval. Objetten har dessutom ett stödmaterial som gör att den kan producera komplexa geometrier, så som överhäng och liknande, mycket enkelt. Stödmaterialet tvättas sedan bort med högtryckstvätt och produkten är därefter klar att användas. Makerboten, som inte har något stödmaterial utan använder plasten för att bygga stöd, har större svårigheter med komplexa geometrier. Efter produktion behöver också en grund som Makerboten skriver ut skäras bort tillsammans med eventuella stöd. Den största fördelen med Makerboten är att produktionskostnaden är markant lägre än för Objetten då Makerbotten använder en relativt enkel termoplast till skillnad från Objettens avancerade fotoplast. Det var denna anledning, samt att modulerna inte kräver Objettens noggrannhet som Makerboten valdes till produktionen av modulerna.

7.2 Friktion

För att få fram den nödvändiga anisotropiska friktion mellan roboten och underlaget testades ett antal idéer. Bakgrunden för dessa finns beskrivet i kapitel 4.

7.2.1 Hjul

Nackdelen med hjul är att roboten får svårt att ta sig fram på ojämna ytor då hjulen kan fastna eller lossna. Hjul ger däremot hög friktion i sidled och låg friktion i framåtriktningen vilket är det som krävs för att få roboten att röra sig framåt. Vid val av hjul var den första tanken att varje modul skulle ha två hjul för att göra roboten stabil. Problemet med detta är att de två hjulen som tillsammans bildar ett hjulpar behöver kunna snurra fristående från varandra då de snurrar olika fort när roboten utför sin slingerrörelse. Det var även svårt att hitta några modellhjul att köpa och att skapa egna hjul ansågs efter en enkel hjulprototyp vara för avancerat då hjulen måste vara lättrullade och helt parallella. Därför valdes LEGO-hjul då dessa uppfyller båda kraven. Dessa ersattes slutligen av ett hjul per modul då detta visade sig ge tillräcklig stabilitet då roboten slingrar sig.

7.2.2 Skenor

De första skenorna gjordes till en del av modulen och skrevs ut fastmonterade av 3D-skrivaren. Då dessa blev för korta och för trubbiga tillverkades istället nya skenor av tunn plast. Skenorna limmades fast på roboten med de gamla skenorna som stöd då skenorna behöver sitta ordentligt för att kunna användas. Hörnen på skenorna rundades för att roboten skulle kunna ta sig fram bättre och inte fastna i mjukare underlag. Skenorna fungerade dock dåligt vilket berodde på att de inte gav tillräcklig skillnad i friktion i längs- och sidled. Detta berodde antagligen på att roboten var för lätt och inte kunde trycka ner skenorna tillräckligt mycket i de underlag som testades, så som platsgolv, grovt tyg och skumgummi.

7.2.3 Gummiplattor

Gummiplattorna testades aldrig i monterad form. När dessa testades att dras mot olika underlag så märktes bara en liten skillnad i längs- och tvärgående riktning. Då en betydligt större skillnad märktes med skenorna och dessa inte fungerade på roboten valdes det att inte gå vidare med gummiplattor.

7.3 Strömförsörjning

Första gången roboten kördes användes en transformator för att få ström till motorerna. För att göra roboten trådlös bestämdes det att använda batterier till roboten.

I ett tidigt skede då roboten kördes användes endast de fyra AA-batterierna, vilka var placerade i svansmodulen, till både styrkort och motorer. Då startade processorn om sig med jämna mellarum vilket berodde på att processorn inte klarar av de spänningsfall som orsakas av motorernas höga strömförbrukning. Spänningsfallet påverkade dock inte motorerna i någon större utsträckning då de sker i väldigt korta impulser. Detta problem löstes enkelt genom att ge processorn en separat strömförsörjning. Dock hade även spänningsfallet över motorerna kunnat undvikas genom att att förse varje motor med en egen spänningskälla. Modulerna skulle för denna lösning behöva göras större för att få plats med ett batteri, vilket var anledningen till att detta inte genomfördes.

7.4 Styrning

För att styra motorerna designades ett eget styrkort då detta behövdes göras litet med tanke på att det skulle placeras i huvudmodulen. Alternativet var att använda den Arduino som fanns att tillgå i början av projektet, men då detta kort hade funktioner som inte behövde användas var det markant större än nödvändigt.

Valet av processor, ATmega328p, baserades på att den Arduinon som fanns att tillgå i början av projektet använde denna processor. Ifall samma processorer användes till styrkortet skulle det vara enkelt att programmera dessa med Arduinon.

7.5 Programmering

Efter att den första koden för framåtrörelse skrevs kunde det konstateras att processorn inte hade någon tid över för att styra en sensor, detta på grund av att fördröjningarna som användes för att skapa PWM-signalerna upptog större delen av cykeltiden för motorerna (20 ms). Den första tanken var att timrar skulle lösa detta. Timrar gjorde det möjligt att implementera sensorn i koden med hjälp av avbrott som kunde ske istället för fördröjningnarna som användes tidigare. Dock orsakade detta felaktiga variationer i längden på pulserna till motorerna vilket troligen orsakats då timrarna stört varandra. Försök gjordes för att lösa detta men när det bedömdes att detta skulle ta allt för lång tid valdes istället att lägga till en andra processor. Samma kod som användes i den första processorn delades då upp över de två processorerna vilket fick motorerna att röra sig felfritt.

Anledningen till att timers testades först var för att spara resurser genom att utnyttja en processor till fullo istället för att ha två processorer som går på halvfart. Det är fördelaktigt ur ett ekonomiskt perspektiv, samt sparar fysiskt utrymme, att lösa problem med mjukvara istället för hårdvara.

7.6 Framåtrörelsen

För att roboten ska kunna köra rakt måste motorerna monteras i sitt mittläge. Dock orsakar motorernas kuggar att länkarna inte hamnar helt rakt vilket kan resultera i att roboten inte blir rak varför kursen avviker något. En annan orsak kan vara att roboten inte är helt symmetrisk, vilket innebär att tyngdpunken inte hamnar i centrum för roboten. Även detta kan bidra till en avvikande kurs. Genom att ha styrning i huvudmodulen och batteri i svansmodulen har detta reducerats något men då batteri är tyngre än styrkortet blir roboten tyngre i svansen. Genom att ha flera mindre batterier utspridda över roboten hade detta undvikits. Vissa moduler hade då behövts designats om varför detta inte genomfördes.

Testerna av follow-mode (kap 5.1.2) visade att reglering på andra sätt borde fungera, exempelvis med GPS-styrning eftersom roboten kan svänga höger eller vänster med hjälp av en liten vinkeländring vilken är lätt att implementera.

7.6.1 Hur påverkar hastigheten framåtrörelsen

Från resultatet av testerna på hastighet i kap 5.1.3 sågs att hastigheten ökade med hastigheten på robotens rörelse vilket är rimligt. Då robotens rörelse blev för snabba minskade däremot hastigheten framåt. Detta berodde på att roboten gled för mycket i sidled vid rörelserna då hjulen inte fick tillräckligt mycket grepp mot underlaget.

8 Slutsats

Detta kapitel besvarar de frågeställningar som ställdes i inledningen.

8.1 Hur skapas den slingrande rörelsen som förflyttar en orm framåt och hur kan den beskrivas?

Gemensamt för de vanligaste rörelsemönster som ormar använder sig utav är att ormen utnyttjar att den har olika friktion i olika riktningar, så kallad anisotrop friktion, för att röra sig framåt. Ormars kroppar består av överlappade fjäll vilka gör att friktionen i sidled och bakåt är mycket större än friktionen framåt. Om friktionen hade varit lika i alla riktningar hade ormen inte kunnat ta sig någonstans.

Vad gäller att beskriva den slingrande rörelsen kan detta göras med varierande svårighetsgrad. När ormar slingrar fördelar de inte sin tyngd jämnt över kontaktytan mot marken, men då detta är svårt att återskapa har inte roboten denna egenskap. Istället följer roboten en vanlig sinusfunktion med tyngden jämnt fördelad över marken.

8.2 Hur kan en ormliknande rörelse genereras?

En ormliknande rörelse kan genereras genom att förse servomotorer med signaler vilka representerar vinklar från en sinus-funktion. Funktionen är tidsförskjuten för varje servomotor beroende på dess position i roboten. Detta innebär att en period (eller flera) delas upp på antalet servomotorer. Detta resulterar i en funktion som går från -1 till 1. Denna funktion multipliceras sedan med den amplitud som man vill att servomotorerna ska röra sig med.

8.3 Hur ska ytan mot underlaget se ut för att roboten ska kunna röra sig framåt?

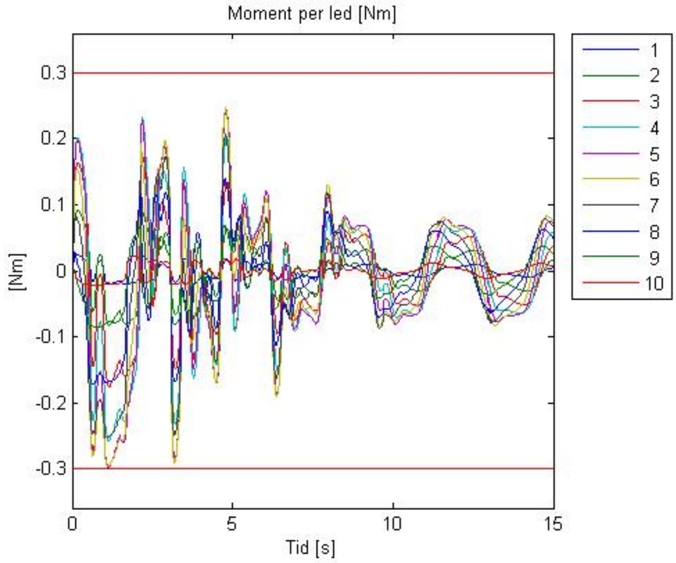
För att efterlikna ormars fjäll behövs ett material som har låg friktion framåt, hög friktion i sidled samt hög friktion bakåt. Den enklaste lösningen att efterlikna detta friktionsmönster är att montera hjul på roboten då dessa har hög friktion i sidled och låg friktion framåt. Dock har de lika låg friktion bakåt som framåt men detta visade inte sig ha någon märkbar inverkan. Ur en estetisk synpunkt hade dock ett hud/skinn/gummiliknande material varit att föredra, men då detta material är svårt att efterlikna samt att funktionalitet har högre prioritering valdes hjul.

8.4 Hur ska robotens stomme designas för att kunna generera nödvändiga rörelser och samtidigt få plats med de komponenter som krävs?

Stommens design hos roboten borde inte påverka robotens funktionalitet i någon större utsträckning, dock testades inte detta. Så länge stommen kan innehålla de komponenter som är nödvändiga samt kan böjas sektionsvis för att möjliggöra en ormliknande rörelse uppfyller stommen sina krav. Därför designades stommen utifrån storleken på de servomotorer som fanns att tillgå i början av projektet.

8.5 Hur starka motorer krävs för att orka röra roboten?

En enkel simulering med moduler som väger 30 g med 0,1 i friktionskoefficient fram och 0,5 åt sidorna gav ett moment på 0,1 Nm. Baserat på detta bedömdes det att motorerna på 0,3 Nm som fanns att tillgå i början av projektet borde räcka.



Figur 8.1: Simulering av 30g moduler med 0,1 friktion framåt, 0,5 åt sidorna. Varje linje är en moduls vridmoment.

8.6 Hur ska roboten styras och kan undvikning av hinder implementeras?

Det enklaste sättet att upptäcka hinder är att använda en sensor av något slag. I detta projekt valdes en ultraljudssensor som inte bara talar om ifall ett objekt befinner sig framför roboten utan även avståndet till detta. Nackdelen med denna typ av sensorer är risken för interferens om fler liknande sensorer är aktiva i närheten. Ifall avståndet är tillräckligt stort kan roboten undvika objektet, dock gjordes denna funktion på motsatt sätt då roboten följer objekt istället. Kommer den för nära ett objekt stannar roboten tills dess färdrikting är fri.

8.7 Vilken energikälla ska användas och hur ska den monteras i roboten?

Då roboten kräver elektricitet för att drivas monterades fyra seriekopplade AA-batterier på 1,2 V vardera att driva servomotorerna i roboten. Då de tog relativt stor plats placerades dessa i en egendesignad svansmodul. För att undvika spänningsfall över processorn gavs denna en egen spänningsskälla i form av två seriekopplade knappcellsbatterier på 3.0 V styck. Dessa placerades i en egendesignad huvudmodul. Till detta finns alternativa lösningar. Spänningsfall kan undgås med bättre kraftkablar till motorerna tillsammans med kondensatorer för att klara de största strömspikar som kan uppstå.

9 Referenser

[1] Gómez, Juan González. Modular robotics and locomotion: application to limbless robots.”(2008).

[2] Pettersen, Kristin Y, Øyvind Stavdahl, and Jan Tommy Gravdahl. Snake Robots: Modelling, Mechatronics, and Control. Springer, 2013.

[3] (2013). MakerBot Replicator 2X Desktop 3D Printer. Hämtad mars 8, 2014, från http://www.makerbot.com/replicator2x.

[4] (2013). Objet24 Desktop 3D Printer | Stratasys. Hämtad mars 8, 2014, från http://www.stratasys.com/3d-printers/design-series/precision/objet24.

[5] (2012). fmincon - MathWorks. Hämtad mars 9, 2014, från http://www.mathworks.se/help/optim/ug/fmincon.html.

[6] (2004). HS-81 Micro Servo - ServoCity. Hämtad mars 9, 2014, från http://www.servocity.com/html/hs-81_micro.html.

[7] (2013). “Mathematical model”, Wikipedia. Hämtad mars 9, 2014, från http://en.wikipedia.org/wiki/Mathematical_model

[8] (2014). “Simulation”, Wikipedia. Hämtad mars 9, 2014, från http://en.wikipedia.org/wiki/Simulation

[9] (2014). “Prototype”, Wikipedia. Hämtad mars 9, 2014, från http://en.wikipedia.org/wiki/Prototype

[10] (2014). “Iteration”, Wikipedia. Hämtad mars 9, 2014, från http://en.wikipedia.org/wiki/Iteration

[11] Gray, J. (1946). The mechanism of locomotion in snakes”. Journal of experimental biology 23 (2): 101–120. PMID 20281580.

Bilaga A Prototyper

Här beskrivs de två prototyper som efter den iterativa processen resulterade i slutprototypen vilken beskrivs i kapitel 4. Båda prototyper designades i CATIA och skrevs ut med en Makerbot Replicator 2X från Makerbot.

A.1 Prototyp 1

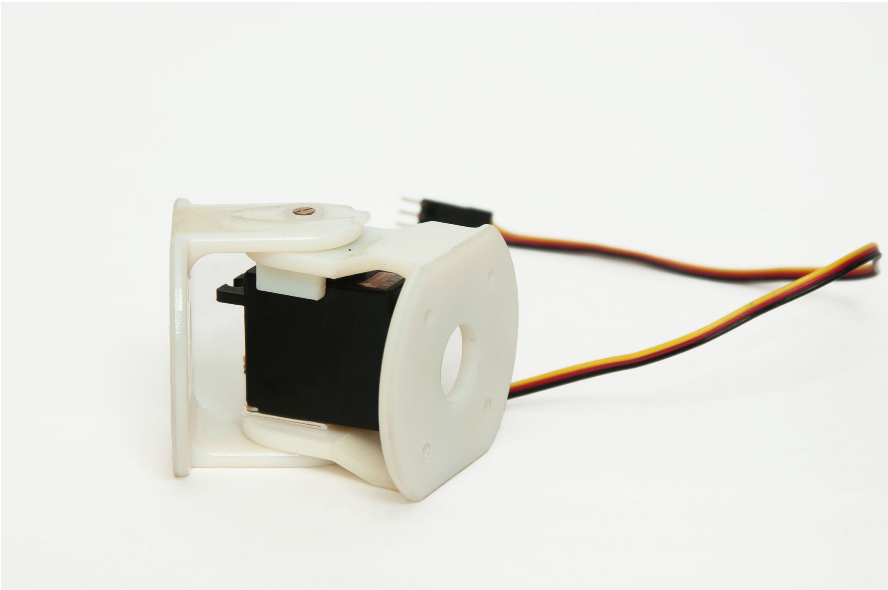
Storleken på moduldesignen grundar sig på de servomotorer som fanns tillgängliga vid projektets start, se figur A.1. För att minimera åtgånget material var det önskvärt att inte göra modulen större än nödvändigt. Servomotorernas storlek avgjorde därför hur stor modulen skulle göras. Den första prototypen av modulen gjordes väldigt simpel. Detta för att snabbt kunna skriva ut en enhet vilken sedan kunde analyseras och förbättras. Denna enkla design har fördelen att det är lätt att montera ihop flera moduler via de fyra små skruvhålen. Delarna har också ett större hål mellan skruvhålen för att kunna dra diverse kablar mellan modulerna. Ur ren estetisk synpunkt är bottenplattan gjord rund för att i större grad likna en verklig orm.

Figur A.1: Delarna till den första prototypen

Efter att modulen producerades analyserades den och en rad förbättringsförslag gjordes. Modulen hade en för liten godstjocklek, 2 mm, vilket gjorde den något ömtålig. Många håldimensioner behövde också ändras. Hålen för motorns axel blev för små och hålen för att fästa en modul i nästa blev för stora. Det fanns inget bra sätt att dels fästa motorn i den ena halvan och det fanns inget bra sätt montera ihop de båda halvorna.

A.2 Prototyp 2

Funktionsprincipen med de två U-formade delarna som bildar varje modul från den första prototypen har behållits i den andra prototypen men med en modifierad design, se A.2. På den nya designen sitter den ena modulhalvans "armar" innanför den andres, de är alltså inte längre identiska. Anledningen till detta var för att motorn skulle få ett fäste i den ena och att axeln skulle få ett fäste i den andra. Godstjockleken har ökats från 2 mm till 3 mm. På den del som motorn ska sitta på har två kanter monterats som bildar ett spår att hålla fast motorn i. På liknande sätt har den andra halvan fått en form som en av de medföljande hornen till motorn kan ligga i.

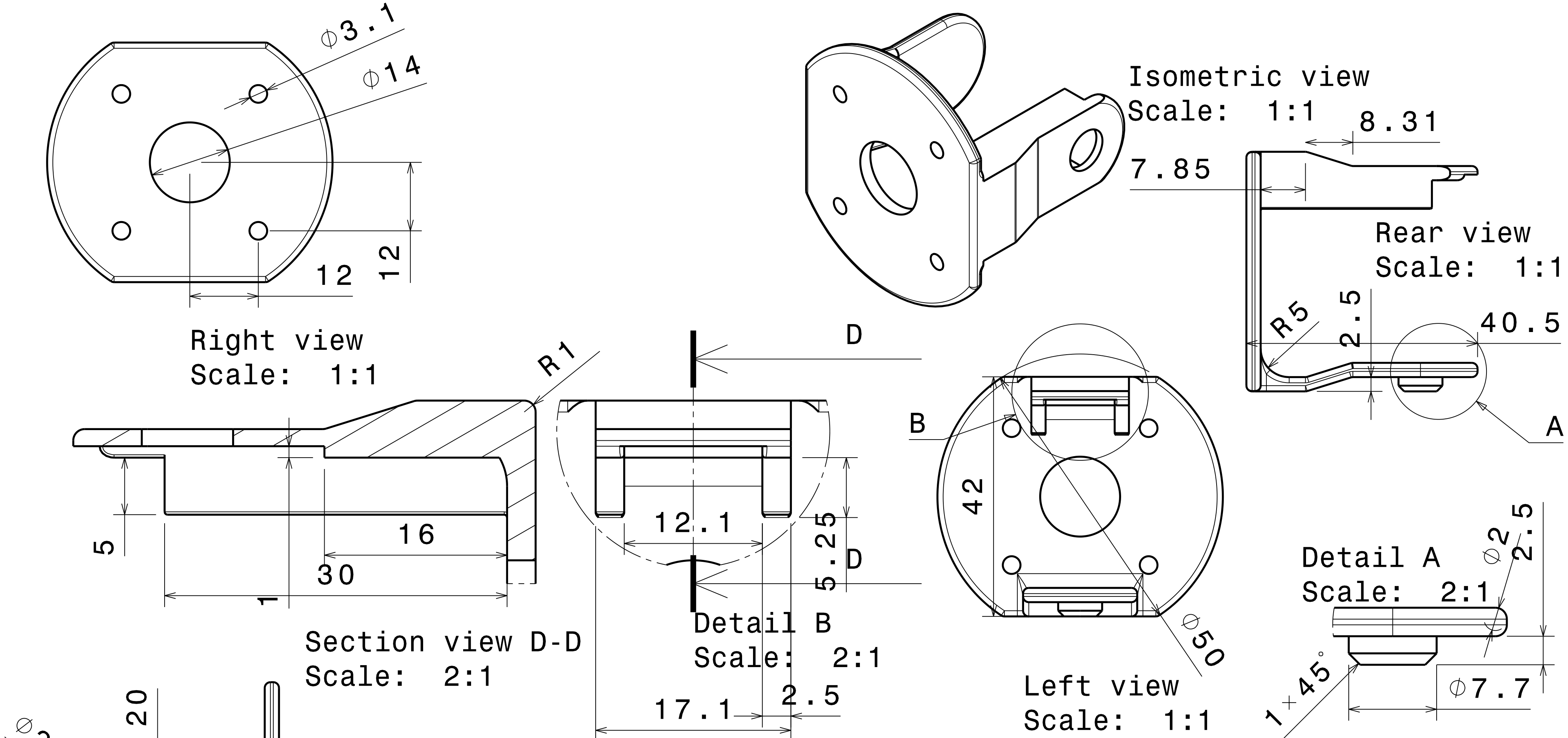


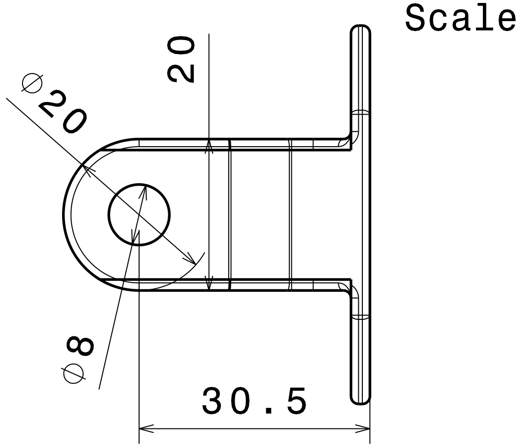
Figur A.2: Prototyp 2 hopmonterad

Modulen blev väldigt robust och kunde haft en något mindre godstjocklek för att spara material och vikt. Modulen kunde även haft skenor (se 3.1.3 Skenor) för att undersöka hurdan friktion skenor skapar mot olika underlag.

Bilaga B Ritningar

Ritningarna för modulerna är avsedda för servomotorer av typen HS-81 och därför måste delarna designas om för att passa andra servomotorer. Delarna är dessutom designade för att passa produktion genom 3D-skrivare. Vid annat tillverkningssätt bör designen göras om.

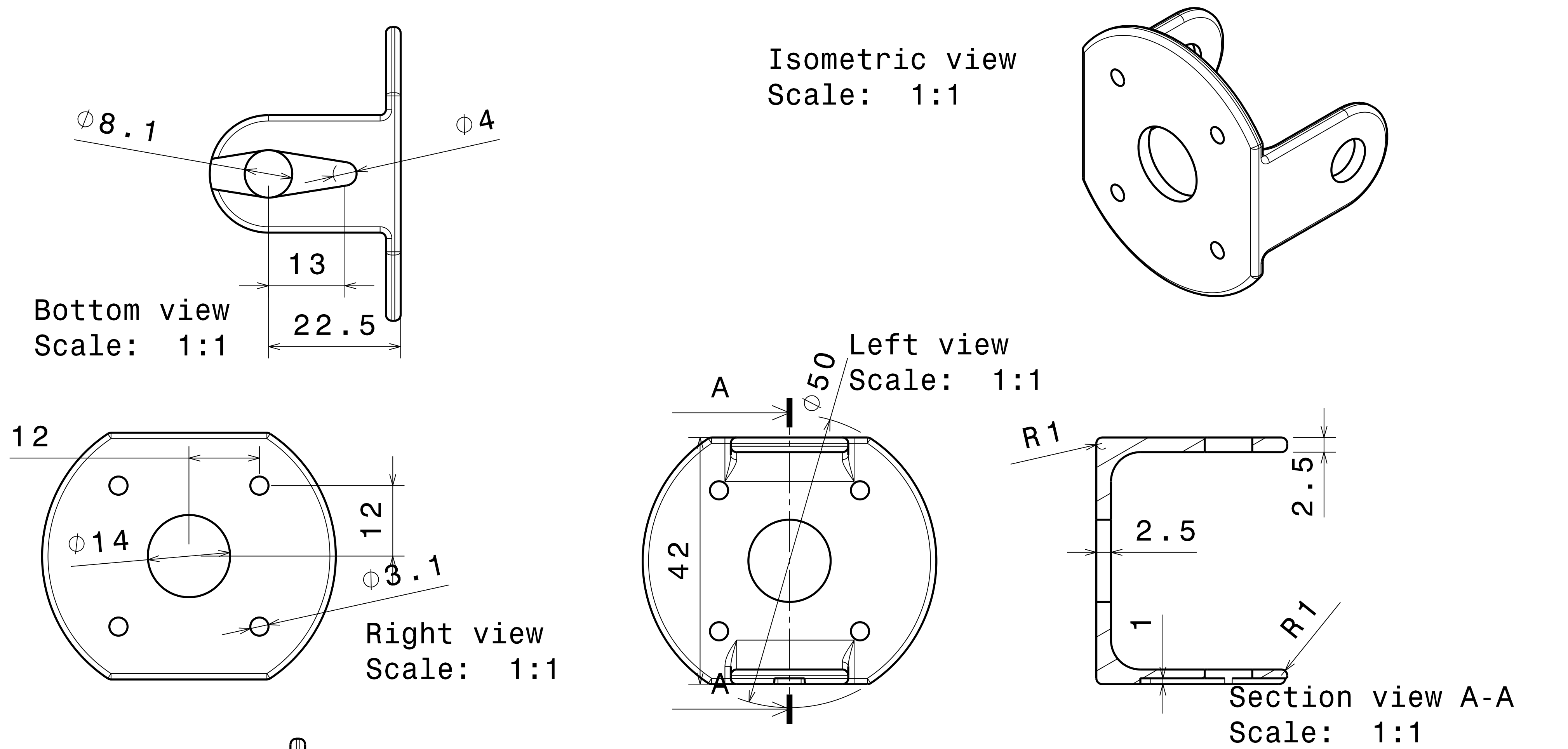


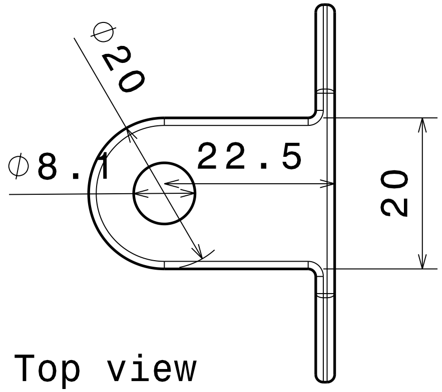


DESIGNED BY:	Marcus Karlsson
DATE:	05/05/2014
CHECKED BY:	XXX
DATE:	XXX
SIZE	A4
SCALE	1:1
WEIGHT (kg)	-
DRAWING NUMBER	Enaxlig modul a v5
SHEET	1/1

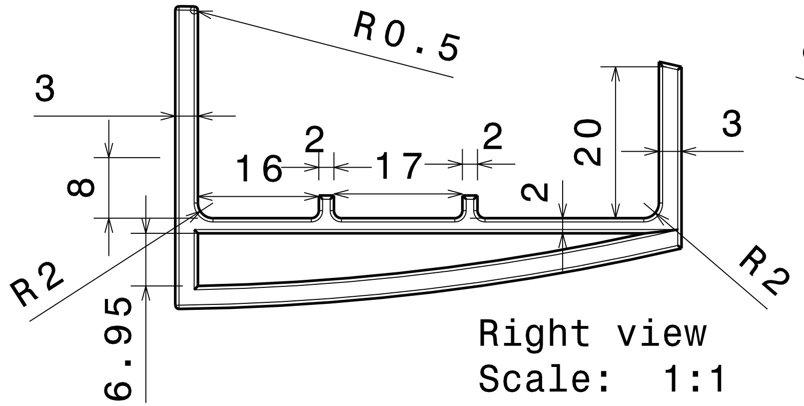
SnakeBot

Rev	
I	_
H	_
G	_
F	_
E	_
D	_
C	_
B	_
A	_

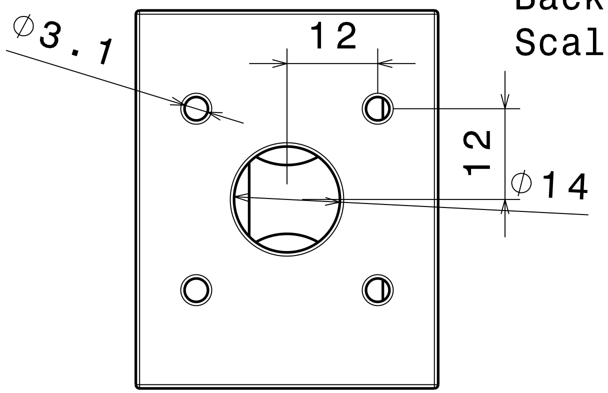




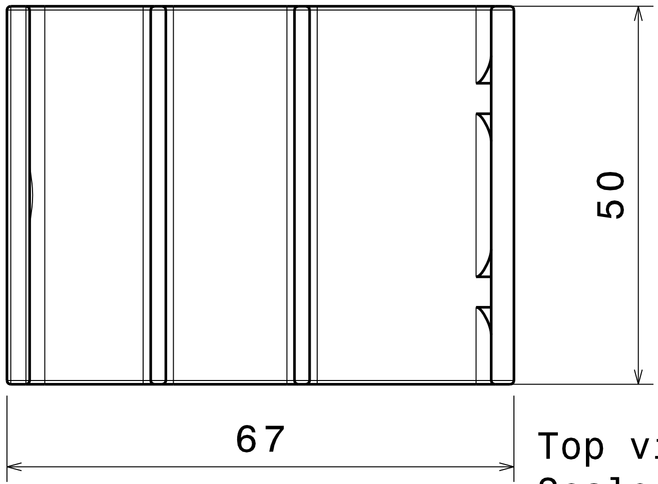
DESIGNED BY: Marcus Karlsson				I _
DATE: 13/05/2014				H _
CHECKED BY: XXX				G _
DATE: XXX				F _
SIZE A4		SnakeBot		E _
				D _
SCALE 1:1	WEIGHT (kg) -	DRAWING NUMBER Enaxlig Modul b v3	SHEET 1/1	C _
				B _
				A _



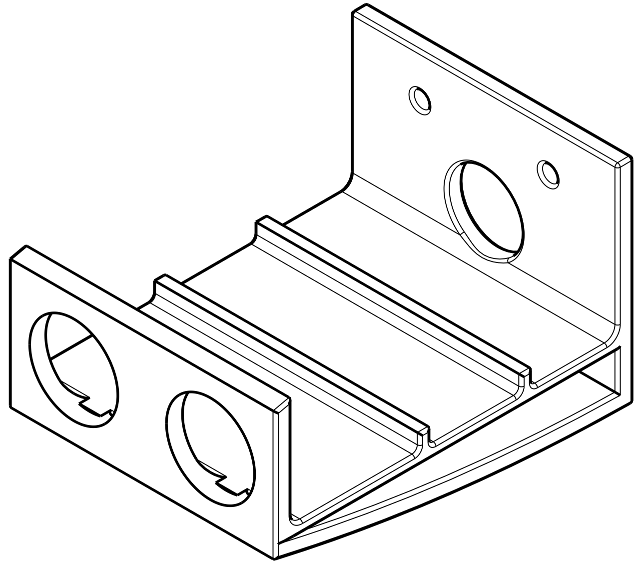
Right view
Scale: 1:1



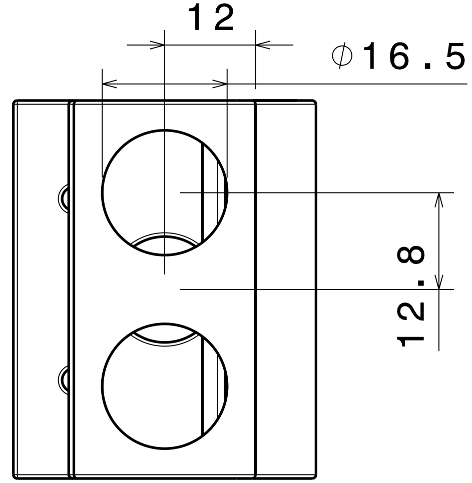
Back view
Scale: 1:1



Top view
Scale: 1:1



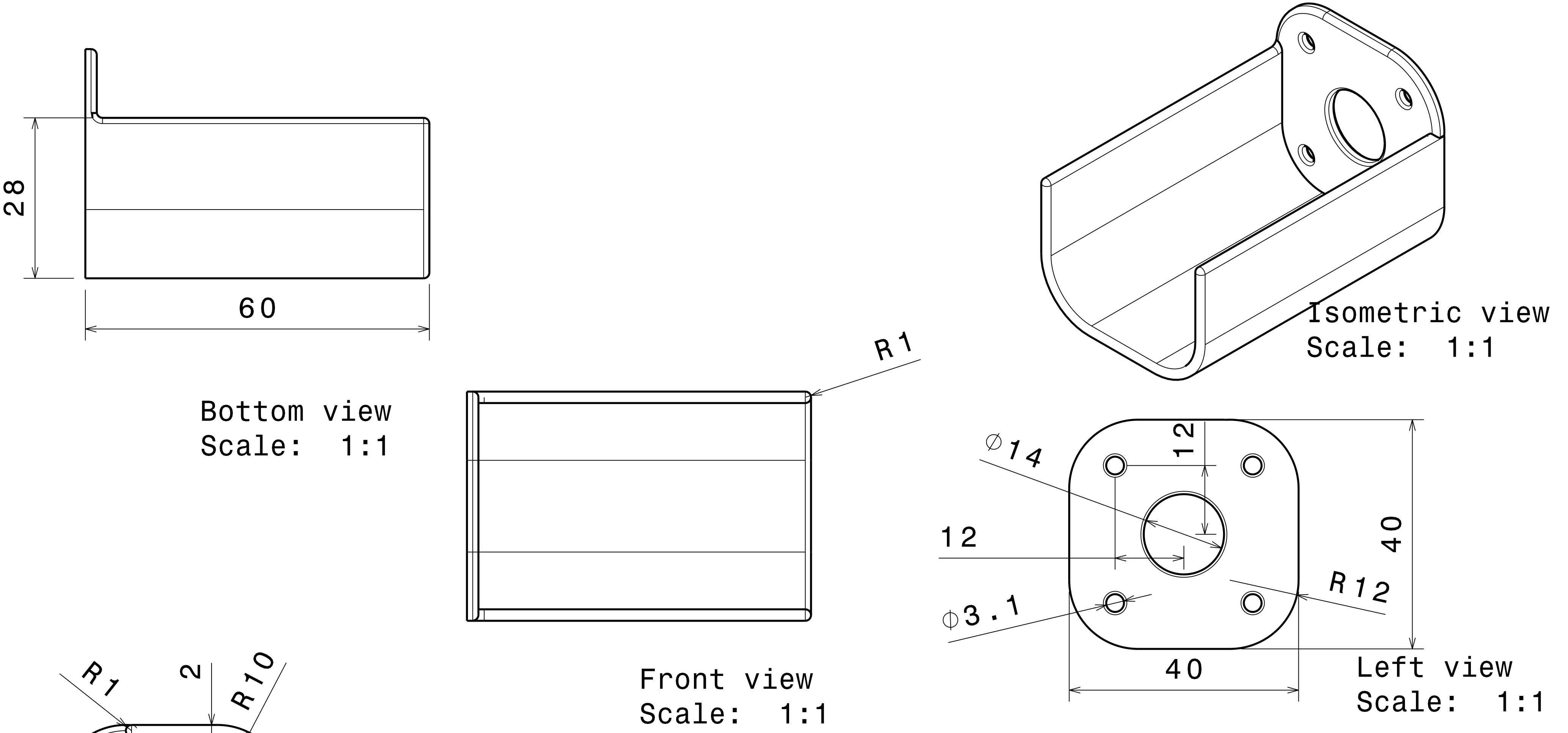
Isometric view
Scale: 1:1

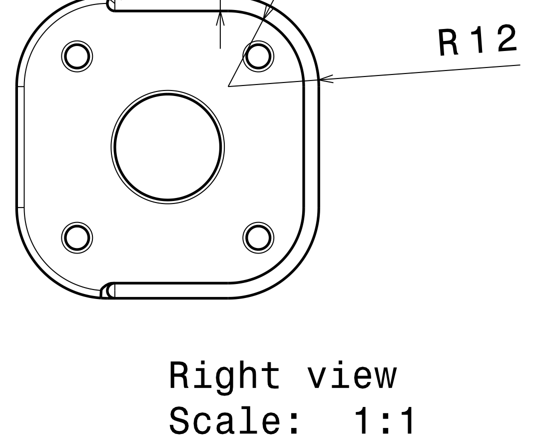


Front view
Scale: 1:1

DESIGNED BY: Marcus Karlsson DATE: 13/05/2014			I	_
CHECKED BY: XXX DATE: XXX			H	_
SIZE A4		SnakeBot	G	_
SCALE 1:1	WEIGHT (kg) -	DRAWING NUMBER Huvudmodul v3	SHEET 1/1	F	_

E	_
D	_
C	_
B	_
A	_





DESIGNED BY: Marcus Karlsson
DATE: 13/05/2014
CHECKED BY: XXX
DATE: XXX

SIZE: A4

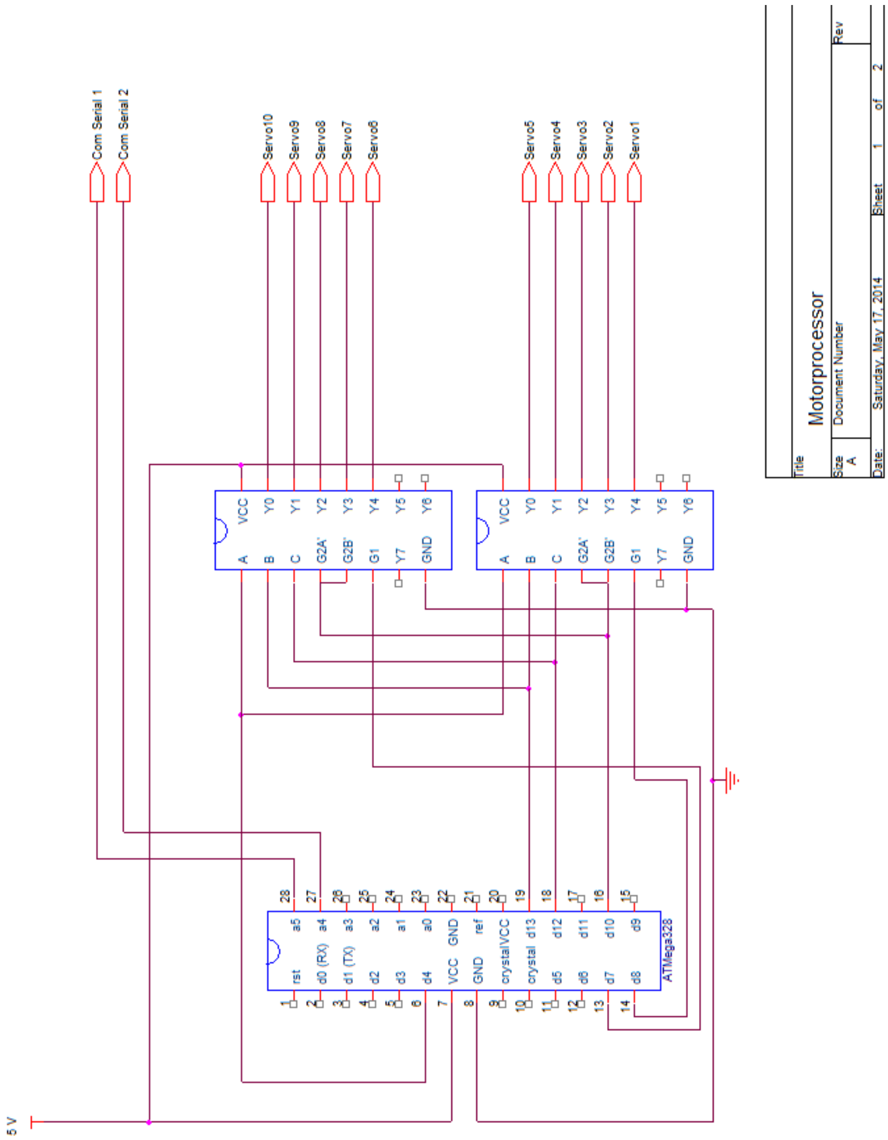
SnakeBot

SCALE	WEIGHT (kg)	DRAWING NUMBER	SHEET
1:1	-	Svansmodul v2	1/1

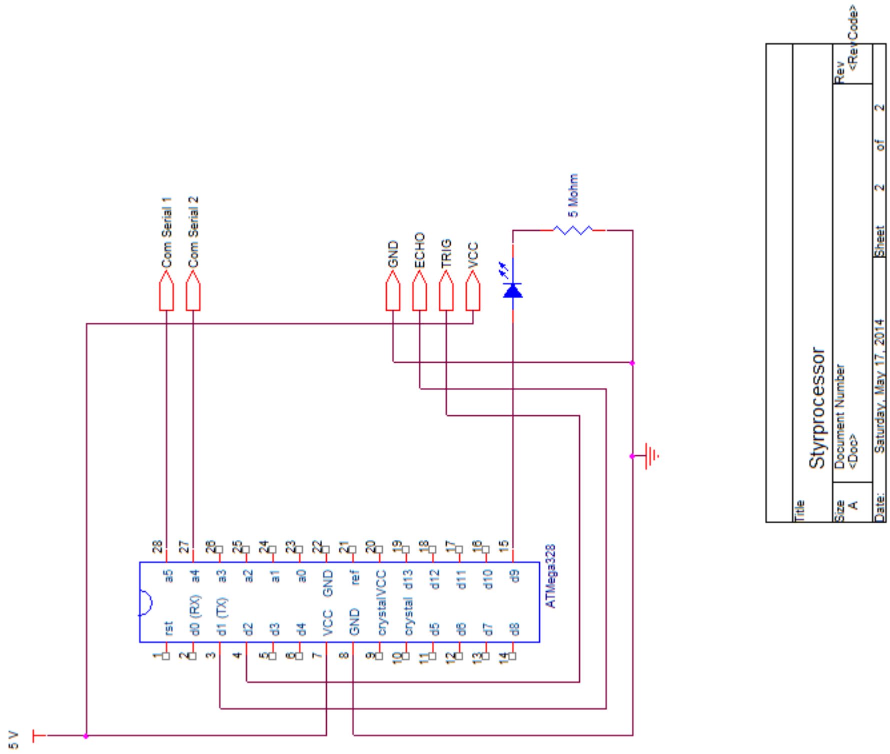
Rev	
I	_
H	_
G	_
F	_
E	_
D	_
C	_
B	_
A	_

Bilaga C Kretsschema

Kretsscheman för styrkortet



Figur C.1: Kretsschema över processor för motorer och dekodrar



Figur C.2: Kretsschema över processor för sensor

Bilaga D Matematisk Modell

D.1 Matematisk modell av robotorm

För att kunna göra en ordentlig analys på en robotorm behövs en matematisk modell. Förutom att modellen ger en grundläggande förståelse för robotormen är den nödvändig för att förstå hur styrningen ska fungera.

Detta kapitel är i hög grad baserad på kapitel 2 i referens [1]. För att enkelt kunna jämföra med källan kommer samma symboler användas och även ett liknande tillvägagångssätt. De generella ekvationer för robotormen som används i [1] är på matrisform (av naturliga skäl) vilket gör det svårt att förstå ekvationernas innebörd. Detta kapitel kommer därför också ge en del förenklade exempel vilket förhoppningsvis ska ge förståelse för de generella ekvationerna. Vissa mer avancerade steg och härledningar från [1] kommer även att utelämnas eller refereras till då nivån överstiger syftet med denna rapport. För vidare läsning rekommenderas också [1].

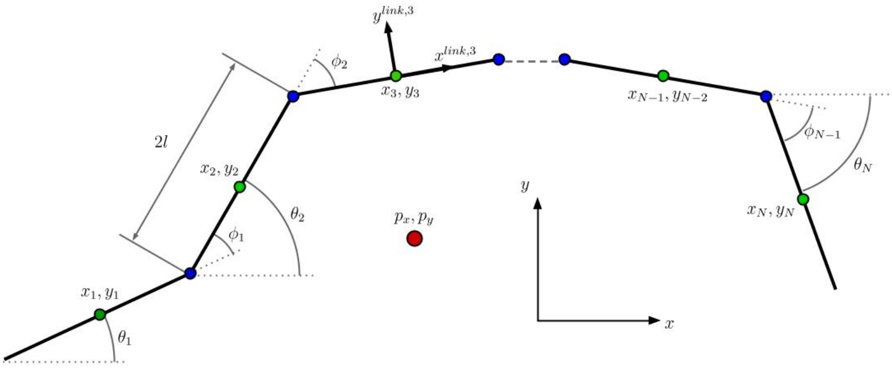
D.1.1 Mål för modellen

Målet med modellen är att få fram en tidsberoende tillståndsvektor för roboten där vektorns derivata är en funktion av dess nuvarande tillstånd och externa respektive pålagda krafter. De externa krafterna roboten utsätts för är friktionskrafter och de pålagda krafterna är momentet från motorerna i robotens leder. De slutgiltiga önskvärda tillstånden för roboten är ledernas vinklar och vinkelhastigheter och masspunktens position och hastighet. Detta då lederna är de frihetsgrader man direkt kan påverka med motorerna och robotens masspunkt det som bäst representerar robotens totala beteende.

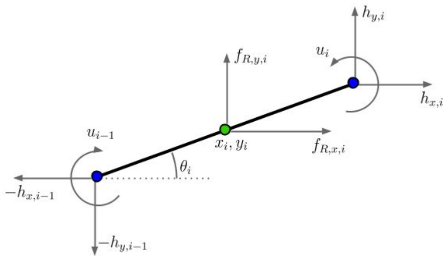
Kapitel D.1 tar fram en modell för en robotorm med N länkar och $N-1$ drivande leder där underlaget antas vara ett homogent och horisontellt. En rörelseekvation för robotormen med hänsyn till vinkelaccelerationen på länkarna, $\ddot{\boldsymbol{\theta}}$, och MC:s acceleration, $\ddot{\mathbf{p}}$, ges i kapitel D.1.6. I kapitel D.1.7 förklaras robotormens rörelse utifrån dess leder, $\ddot{\boldsymbol{\phi}}$, huvudets vinkelacceleration, $\boldsymbol{\theta}_N$, och MC:s acceleration, $\ddot{\mathbf{p}}$ vilket separerade drivande och ickedrivande frihetsgrader. Till slut förenklas modellen för att lättare kunna styra robotormen i kapitel D.1.8.

D.1.2 Parametrar för modell av robotorm

Modellen skapas i ett 2-dimensionellt horisontellt plan med homogent underlag. Robotormen är uppbyggd av N stela länkar vilka sitter i ihop med $N-1$ motoriserade leder. Varje länk har längden $2l$ och massan m. Bredden på varje länk räknas inte med i modellen. Då varje länk har homogen densitet hamnar MC (masscentrum) i mitten av varje längd. Tröghetsmomentet för varje länk är $J = \frac{1}{3}ml^2$ och den totala massa för roboten blir Nm. Modellens parametrar finns sammanställda i tabell 3 och illustrerade i figurerna D.1 och D.2.



Figur D.1: De kinetiska parametrarna för robotormen



Figur D.2: Krafter och moment på varje länk i robotormen (modifierad från [1])

Vektorerna som definieras i tabell 11 är kolonnvektorer med elementen, exempelvis $\mathbf{X} = [x_1,...,x_N]^T \in \mathbf{R}^N$. Ormens generella ekvationer kommer även använda sig av följande vektorer och matriser

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & \ddots \\ & & 1 & 1 \end{bmatrix} \in \mathbb{R}^{N-1 \times N}, \qquad \mathbf{D} = \begin{bmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & \ddots & \ddots \\ & & 1 & -1 \end{bmatrix} \in \mathbb{R}^{N-1 \times N},$$

$$\boldsymbol{e} = [1,...,1]^T \in \mathbb{R}^N, \qquad \dot{\boldsymbol{\theta}}^2 = [\dot{\theta}_1^2,...,\dot{\theta}_N^2] \in \mathbb{R}^N,$$

$$\sin\boldsymbol{\theta} = [\sin\theta_1,..., \sin\theta_N]^T\mathbb{R}^N, \qquad \cos\boldsymbol{\theta} = [\cos\theta_1,..., \cos\theta_N]^T\mathbb{R}^N,$$

$$\boldsymbol{S}_\theta = \begin{bmatrix} \sin\theta_1 & & & \\ & \ddots & & \\ & & \ddots & \\ & & & \sin\theta_N \end{bmatrix} \in \mathbb{R}^{N \times N}, \qquad \boldsymbol{C}_\theta = \begin{bmatrix} \cos\theta_1 & & & \\ & \ddots & & \\ & & \ddots & \\ & & & \cos\theta_N \end{bmatrix} \in \mathbb{R}^{N \times N}.$$

A och **D** är matriser som adderar respektive subtraherar gränsande element i en vektor,. Dessa används ofta då en länk beror på de angränsande länkarna. Vektorn $\boldsymbol{e}$ är till för att summera alla element i en N-dimensionell vektor. Resterande matriser definieras då de är vanligt förekommande i detta kapitel.

Tabell 11: Parametrar till modell av robotorm (modifierad från [1])

Symbol	Enhet	Förklaring	Vektor
N		Antalet länkar	
m	kg	Massa per länk	
l	m	Halva längden per länk	
J	kgm^2	Tröghetsmoment på varje länk	
θ_i	rad	Vinkel mellan länk i och x-axeln	$\boldsymbol{\theta} \in \mathbb{R}^N$
ϕ	rad	Vinkel på led i	$\boldsymbol{\phi} \in \mathbb{R}^{N-1}$
x_i,y_i		Kordinater för MC på länk i	$\mathbf{X}$,$\mathbf{Y} \in \mathbb{R}^N$
p_x,p_y		Kordinater för robotens MC	$\boldsymbol{p} \in \mathbb{R}^2$
u_i	Nm	Drivande moment på länk i från länk $i+1$	$\boldsymbol{u} \in \mathbb{R}^{N-1}$
u_{i-1}	Nm	Drivande moment på länk i från länk $i-1$	$\boldsymbol{u} \in \mathbb{R}^{N-1}$
$f_{R,x,i}$,$f_{R,y,i}$	N	Friktionskrafter på länk i	$\boldsymbol{f}_{R,x}$,$\boldsymbol{f}_{R,y} \in \mathbb{R}^N$
$h_{x,i}$,$h_{y,i}$	N	Ledens tvångkraft på länk i från länk $i+1$	$\boldsymbol{h}_x$,$\boldsymbol{h}_y \in \mathbb{R}^{N-1}$
$-h_{x,i-1}, -h_{y,i-1}$	N	Ledens tvångkraft på länk i från länk $i-1$	$\boldsymbol{h}_x$,$\boldsymbol{h}_y \in \mathbb{R}^{N-1}$

D.1.3 Modellens kinetik

Detta delstycke undersöker sambanden mellan robotens positionsparametrar. Det vill säga länkarnas positioner (x_i,y_i), länkarnas vinkel mot x-axeln θ_i, ledernas vinklar ϕ_i och robotens MC:s position (p_x,p_y). Även parametrarnas hastighet och dessa samband tas fram. Positionsparametrarna illustreras i figur D.1. Notera att robotormen består av $N+2$ frihetsgrader, vinklarna på varje länk och MC:s position i planet.

Ledernas vinklar relaterar till länkarnas vinklar enligt

$$\phi_i = \theta_i - \theta_{i+1} \tag{D.1}$$

och ormens MC $\boldsymbol{p}$ relaterar till länkarnas position enligt

$$\boldsymbol{p} = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} \frac{1}{Nm}\sum_{i=1}^N mx_i \\ \frac{1}{Nm}\sum_{i=1}^N my_i \end{bmatrix} = \frac{1}{N}\begin{bmatrix} \boldsymbol{e}^T\mathbf{X} \\ \boldsymbol{e}^T\mathbf{Y} \end{bmatrix}. \tag{D.2}$$

Då alla länkarna är homogena blir MC länkarnas medelposition.

Sambandet mellan länk i och länk $i+1$ i led i där länkarna är bundna till varandra ger en relation mellan länkarnas position (x_i,x_{i+1}) och länkarnas vinkel (θ_i,θ_{i+1}) enligt

$$x_{i+1} - x_i = l\cos\theta_i + l\cos\theta_{i+1}, \tag{D.3a}$$

$$y_{i+1} - y_i = l\sin\theta_i + l\sin\theta_{i+1}. \tag{D.3b}$$

Dessa ekvationer kan sammanställas i matrisform enligt

$$\mathbf{DX} + l\mathbf{A}\cos\boldsymbol{\theta} = \mathbf{0}, \tag{D.4a}$$

$$\mathbf{DY} + l\mathbf{A}\sin\boldsymbol{\theta} = \mathbf{0}. \tag{D.4b}$$

Notera additionen liksom subtraktionen av närstående länkar i (D.3) vilket är bakgrunden till skapandet av matriserna **A** och **D**.

Målet är att lösa ut varje länks position som en funktion av länkvinklarna och robotormens MC. Detta går att göra genom att kombinera (D.4) med (D.2). Härledningen är dock svår, därför ges först ett mer intuitivt exempel och därefter ges den generella ekvationen.

Exemplet går ut på att lösa ut x_i som en funktion av MC och länkvinklarna. Genom att använda p_x ur (D.2) och definiera alla x_i utifrån x_1 och länkvinklarn. Sedan löses x_1 ut

$$p_x = \frac{1}{N}\sum_{i=1}^{N} mx_i = \frac{1}{N}(x_1 + (x_1 + l\cos\theta_1 + l\cos\theta_2) + (x_1 + l\cos\theta_1 + 2l\cos\theta_2 + \cos\theta_3)...), \tag{D.5a}$$

$$x_1 = p_x - \frac{l}{N}\sum_{i=1}^{N}((2(N-1)-1)\cos\theta_i). \tag{D.5b}$$

Genom att studera (D.5) kan det inses att varje x_i kan skrivas som en funktion av p_x och de andra länkarnas position. De andra länkarnas position utgår från x_i och är en funktion av $\boldsymbol{\theta}$. Kortfattat beror alltså varje enskild länks position i x-led, x_i, av p_x och en vägning av resterande länkars $\cos\boldsymbol{\theta}$.

Från [1] ges nu den generella ekvationen utan vidare förklaring som

$$\mathbf{X} = -l\mathbf{K}^T\cos\boldsymbol{\theta} + \boldsymbol{e}p_x, \tag{D.6a}$$

$$\mathbf{Y} = -l\mathbf{K}^T\sin\boldsymbol{\theta} + \boldsymbol{e}p_y. \tag{D.6b}$$

Där $\mathbf{K} = \mathbf{A}^T(\mathbf{DD}^T)^{-1}\mathbf{D} \in \mathbb{R}(N \times N)$. **K** är den matris som väger $\cos\boldsymbol{\theta}$.

Genom att derivera (D.6) med avseende på tiden fås länkarnas hastigheter fram som funktion av länkarnas vinkelhastighet och MC:s hastighet fås enligt

$$\dot{\mathbf{X}} = l\mathbf{K}^T\boldsymbol{S}_\theta\dot{\boldsymbol{\theta}} + \boldsymbol{e}\dot{p}_x, \tag{D.7a}$$

$$\dot{\mathbf{Y}} = -l\mathbf{K}^T\boldsymbol{C}_\theta\dot{\boldsymbol{\theta}} + \boldsymbol{e}\dot{p}_y. \tag{D.7b}$$

Notera att $\boldsymbol{S}_\theta\dot{\boldsymbol{\theta}} = [\dot{\theta}_1\sin\theta_1,...,\dot{\theta}_N\sin\theta_N]$.

D.1.4 Friktionsmodell för robotormen

Som beskrivet i kapitel 2 är friktionen en viktig faktor för hur roboten rör sig framåt. Det förklaras också att en robotorm som rör sig med en sinusrörelse och konstant kontakt mot underlaget ska kunna ta sig framåt krävs att länkarna har anisotrop friktion. Det är alltså viktigt att en bra modell för friktionen tas fram. Den mest verklighetstrogna modellen, som är önskvärt ur simuleringssynpunkt, är Coulumbs modell där friktionen är proportionell mot länkarnas vikt.

Friktionskoefficienterna betecknas μ_t och μ_n som respektive är koefficient i länkens riktning, $x^{link,i}$ och koefficient i länkens normal, $y^{link,i}$ (se figur D.1). Friktionskraften $(f_{R,x,i},f_{R,y,i})$ är riktad i motsatt riktning mot länkens hastighet $(\dot{x}_i,\dot{y}_i)$. Detta leder till att en $Sign$ funktion (ger tecknet på dess parameter: $1{,}0,-1$) naturligt används då tecknet på friktionskrafterna är beroende av tecknet på hastigheterna. Notera att isotrop friktion kan skapas genom att sätta μ_t och μ_n lika.

Då friktionskoefficienterna följer länkarnas riktning medan länkarnas hastighet är i x:y planet blir de annars enkla friktionsekvationerna relativt svåra. Här ges därför direkt den generella ekvationen för friktionen på varje länk enligt

$$\boldsymbol{f}_R = \begin{bmatrix} \boldsymbol{f}_{R,x} \\ \boldsymbol{f}_{R,y} \end{bmatrix} = -mg \begin{bmatrix} \mu_t \mathbf{C}_\theta & -\mu_n \mathbf{S}_\theta \\ \mu_t \mathbf{S}_\theta & \mu_n \mathbf{C}_\theta \end{bmatrix} sign\left(\begin{bmatrix} \mathbf{C}_\theta & \mathbf{S}_\theta \\ -\mathbf{S}_\theta & \mathbf{C}_\theta \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\mathbf{Y}} \end{bmatrix} \right) \in \mathbb{R}^{2N}. \tag{D.8}$$

För analys av robotens rörelse och de krafter som friktionen påverkar ormen med är Coulumbs modell och ekvation D.8 komplicerad. Därför kommer även en viskös friktion modell introduceras från kapitel 2.5 i [1]. Denna modell är friktionen linjär mot hastigheten och tar därför bort behovet av $Sign$ funktionen. Den färdiga modellen visas i sin helhet och används vid analys av drivande krafter på robotormen i kapitel 3.3.1.

$$\boldsymbol{f}_R = \begin{bmatrix} \boldsymbol{f}_{R,x} \\ \boldsymbol{f}_{R,y} \end{bmatrix} = -\begin{bmatrix} c_t(\mathbf{C}_\theta)^2 + c_n(\mathbf{S}_\theta)^2 & (c_t - c_n)\mathbf{S}_\theta\mathbf{C}_\theta \\ (c_t - c_n)\mathbf{S}_\theta\mathbf{C}_\theta & c_t(\mathbf{S}_\theta)^2 + c_n(\mathbf{C}_\theta)^2 \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\mathbf{Y}} \end{bmatrix} \in \mathbb{R}^{2N}, \tag{D.9}$$

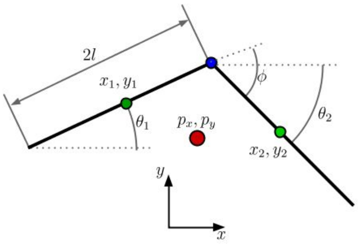
där c_t och c_n är nya friktionskoeficienter.

D.1.5 Dynamik för enkel modell med två länkar

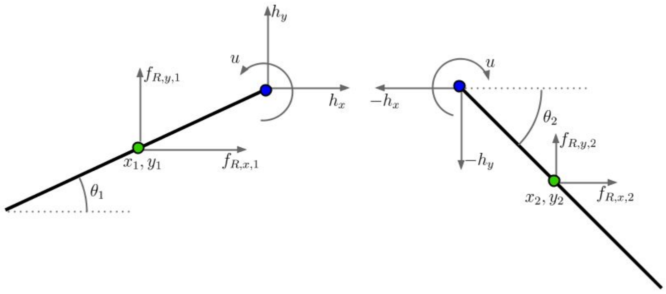
I detta kapitel kommer dynamiken för en enkel robot med två länkar tas fram. Detta för att få en mer intuitiv bild av nästa delkapitel, kap.D.1.6, där robotens generella dynamik beskrivs. Dynamiken för den enkla roboten beskriver länkarnas och MC:s acceleration $(\ddot{\theta}_1,\ddot{\theta}_2,\ddot{p}_x,\ddot{p}_y,)$ som en funktion av deras tillstånd $(\dot{\theta}_1,\theta_1,\dot{\theta}_2,\theta_2,\dot{p}_x,p_x,\dot{p}_y,p_y)$ och krafter på ormen. Dessa är friktionskrafterna $(f_{R,x,1},f_{R,y,1},f_{R,x,2},f_{R,y,2})$ och motorns kraft i leden (u).

Den enkla roboten har parametrar enligt figur D.3 vilket är en förenkling av figur D.1. Kraftmodellen från figur D.2 och friläggning av de två länkarna i den enkla roboten ger kraftförhållandena som ses i figur D.4.
Ur figur D.4 fås kraftbalansen för de två länkarna fram enligt

$$m\ddot{x}_1 = f_{R,x,1} + h_x, \tag{D.10a}$$
$$m\ddot{y}_1 = f_{R,y,1} + h_y, \tag{D.10b}$$
$$m\ddot{x}_2 = f_{R,x,2} - h_x, \tag{D.10c}$$
$$m\ddot{y}_2 = f_{R,y,2} - h_y. \tag{D.10d}$$



Figur D.3: Kinetiska parametrarna för enkel modell



Figur D.4: Friläggning av enkel modell

De två länkarnas acceleration kan även tas fram genom att först skriva om ekvation D.3 till den enkla modellen och derivera två gånger med avseende på tiden vilket ger

$$x_2 - x_1 = l(\cos(\theta_1) + \cos(\theta_2)) \Rightarrow \left\{\frac{d^2}{dt^2}\right\} \Rightarrow \tag{D.11a}$$

$$\ddot{x}_1 - \ddot{x}_2 = l(\ddot{\theta}_2 \sin(\theta_1) + \dot{\theta}_2^2 \cos(\theta_1) + \ddot{\theta}_2 \sin(\theta_2) + \dot{\theta}_2^2 \cos(\theta_2)), \tag{D.11b}$$

$$y_2 - y_1 = l(\sin(\theta_1) + \sin(\theta_2)) \Rightarrow \left\{\frac{d^2}{dt^2}\right\} \Rightarrow \tag{D.11c}$$

$$\ddot{y}_1 - \ddot{y}_2 = l(-\ddot{\theta}_1 \cos(\theta_1) + \dot{\theta}_1^2 \sin(\theta_1) - \ddot{\theta}_2 \cos(\theta_2) + \dot{\theta}_2^2 \sin(\theta_2)). \tag{D.11d}$$

MC:s acceleration fås fram genom att skriva om ekvation D.2 till den enkla modellen och derivera två gånger med avseende på tiden vilket ger

$$p_x = \frac{x_1 + x_2}{2} \Rightarrow \left\{\frac{d^2}{dt^2}\right\} \Rightarrow \ddot{p}_x = \frac{\ddot{x}_1 + \ddot{x}_2}{2}, \tag{D.12a}$$

$$p_y = \frac{y_1 + y_2}{2} \Rightarrow \left\{\frac{d^2}{dt^2}\right\} \Rightarrow \ddot{p}_y = \frac{\ddot{y}_1 + \ddot{y}_2}{2}. \tag{D.12b}$$

Eliminera kontaktkrafterna (h_x,h_y) genom att sätta in ekvation D.10 i D.12, detta resulterar i

$$\ddot{p}_x = \frac{1}{2m}(f_{R,x,1} + f_{R,x,2}), \tag{D.13a}$$

$$\ddot{p}_y = \frac{1}{2m}(f_{R,y,1} + f_{R,y,2}). \tag{D.13b}$$

Som förväntat resulterar detta i att accelerationen på MC är lika med de externa krafterna på roboten delat med dess massa.

Momentbalansen tas fram för de två länkarna utifrån figur D.4

$$J\ddot{\theta}_1 = u - l\sin(\theta_1)h_x + l\cos(\theta_1)h_y, \tag{D.14a}$$

$$J\ddot{\theta}_2 = -u - l\sin(\theta_2)h_x + l\cos(\theta_2)h_y. \tag{D.14b}$$

Kontaktkrafterna i D.14 tas bort genom att lösa ut (x_1,x_2,y_1,y_2) ur ekvation D.10 och stoppa in i D.11 enligt

$$\frac{f_{R,x,1} - f_{R,x,2} + 2h_x}{m} = l(\ddot{\theta}_2 \sin(\theta_1) + \dot{\theta}_2^2 \cos(\theta_1) + \ddot{\theta}_2 \sin(\theta_2) + \dot{\theta}_2^2 \cos(\theta_2)), \tag{D.15a}$$

$$\frac{f_{R,y,1} - f_{R,y,2} + 2h_y}{m} = l(-\ddot{\theta}_1 \cos(\theta_1) + \dot{\theta}_1^2 \sin(\theta_1) - \ddot{\theta}_2 \cos(\theta_2) + \dot{\theta}_2^2 \sin(\theta_2)). \tag{D.15b}$$

Lös ut h_x och h_y ur D.15

$$h_x = \frac{f_{R,x,2} - f_{R,x,1}}{2} + \frac{ml}{2}(\ddot{\theta}_2 \sin(\theta_1) + \dot{\theta}_2^2 \cos(\theta_1) + \ddot{\theta}_2 \sin(\theta_2) + \dot{\theta}_2^2 \cos(\theta_2)), \tag{D.16a}$$

$$h_y = \frac{f_{R,y,2} - f_{R,y,1}}{2} + \frac{ml}{2} l(-\ddot{\theta}_1 \cos(\theta_1) + \dot{\theta}_1^2 \sin(\theta_1) - \ddot{\theta}_2 \cos(\theta_2) + \dot{\theta}_2^2 \sin(\theta_2)). \tag{D.16b}$$

Avslutningsvis kan ekvation D.16 sättas in i ekvation D.14. $\ddot{\theta}_1$ och $\ddot{\theta}_2$ kan då lösas ut som en funktion av länkarnas tillstånd, de externa friktionskrafterna och den interna kraften u. Tillsammans med ekvation D.13 beskriver dessa den enkla modellens dynamik.

D.1.6 Generell dynamik för robotormen

I detta kapitel tas den generella dynamiken fram för robotormen. Tillvägagångssättet är detsamma som för den enkla modellen i kapitel D.1.5.

Dynamiken för roboten beskriver länkarnas och MC:s acceleration $(\ddot{\boldsymbol{\theta}},\ddot{\boldsymbol{p}})$ som en funktion av deras tillstånd $(\dot{\boldsymbol{\theta}},\boldsymbol{\theta},\dot{\boldsymbol{p}},\boldsymbol{p})$ och krafter på ormen, vilket är friktionskrafterna $(\boldsymbol{f}_{R,x},\boldsymbol{f}_{R,y})$ och motorernas krafter i lederna $(\boldsymbol{u})$.

Kraftbalansen för en generell länk, i, fås från figur D.2 enligt

$$m\ddot{x}_i = f_{R,x,1} + h_{x,i} - h_{x,i-1}, \tag{D.17a}$$

$$m\ddot{y}_i = f_{R,y,1} + h_{y,i} - h_{y,i-1}. \tag{D.17b}$$

$$\tag{D.17c}$$

Kraftbalansen för alla länkar kan uttryckas i matrisform som

$$m\ddot{\mathbf{X}} = \boldsymbol{f}_{R,x} + \mathbf{D}^T \mathbf{h}_x, \tag{D.18a}$$

$$m\ddot{\mathbf{Y}} = \boldsymbol{f}_{R,y} + \mathbf{D}^T \mathbf{h}_y. \tag{D.18b}$$

Länkarnas acceleration kan även tas fram genom att derivera ekvation D.4 två gånger med avseende på tiden vilket ger

$$\mathbf{D}\ddot{\mathbf{X}} = l\mathbf{A}(\mathbf{C}_\theta \dot{\boldsymbol{\theta}}^2 + \mathbf{S}_\theta \ddot{\boldsymbol{\theta}}), \tag{D.19a}$$

$$\mathbf{D}\ddot{\mathbf{Y}} = l\mathbf{A}(\mathbf{S}_\theta \dot{\boldsymbol{\theta}}^2 - \mathbf{C}_\theta \ddot{\boldsymbol{\theta}}). \tag{D.19b}$$

MC:s acceleration fås fram genom att derivera ekvation D.2 två gånger med avseende på tiden och sätta in ekvation D.18 vilket eliminerar kontaktkrafterna när de summeras. Resultatet blir

$$\ddot{\boldsymbol{p}} = \begin{bmatrix} \ddot{p}_x \\ \ddot{p}_y \end{bmatrix} = \frac{1}{N} \begin{bmatrix} \boldsymbol{e}^T \ddot{\mathbf{X}} \\ \boldsymbol{e}^T \ddot{\mathbf{Y}} \end{bmatrix} = \frac{1}{Nm} \begin{bmatrix} \boldsymbol{e}^T \mathbf{f}_{R,x} \\ \boldsymbol{e}^T \mathbf{f}_{R,y} \end{bmatrix} \tag{D.20}$$

och precis som beskrevs i den enkla modellen betyder detta, vilket är förväntat, att accelerationen på robotens MC är lika med de externa krafterna på ormen delat på dess massa.

Momentbalansen för en generell länk, i, ges av

$$J\ddot{\theta}_i = u_i - u_{i-1} - l\sin(\theta_i)(h_{x,i} + h_{x,i-1}) + l\cos(\theta_i)(h_{y,i} + h_{y,i-1}), \tag{D.21}$$

och momentbalansen för alla länkarna kan då skrivas som

$$J\ddot{\boldsymbol{\theta}}_i = \mathbf{D}^T\mathbf{u} - l\mathbf{S}_\theta\mathbf{A}^T\mathbf{h}_x + l\mathbf{C}_\theta\mathbf{A}^T\mathbf{h}_y. \tag{D.22}$$

För att få ett oberoende av kontaktkrafterna multipliceras ekvation D.18 med $\mathbf{D}$, resultatet sätts in i ekvation D.19, genom att lösa ut $\mathbf{h}_x$ och $\mathbf{h}_y$ fås

$$\mathbf{h}_x = (\mathbf{D}\mathbf{D}^T)^{-1}(ml\mathbf{A}(\mathbf{C}_\theta\dot{\boldsymbol{\theta}}^2 + \mathbf{S}_\theta\ddot{\boldsymbol{\theta}}) - \mathbf{D}\mathbf{f}_{R,x}), \tag{D.23a}$$

$$\mathbf{h}_y = (\mathbf{D}\mathbf{D}^T)^{-1}(ml\mathbf{A}(\mathbf{S}_\theta\dot{\boldsymbol{\theta}}^2 - \mathbf{C}_\theta\ddot{\boldsymbol{\theta}}) - \mathbf{D}\mathbf{f}_{R,y}). \tag{D.23b}$$

Genom att sätta in ekvation D.23 i D.22 och lösa ut $\ddot{\boldsymbol{\theta}}$ kan modellen för robotormen skrivar ut som

$$\mathbf{M}_\theta\ddot{\boldsymbol{\theta}} + \mathbf{W}\dot{\boldsymbol{\theta}}^2 - l\mathbf{S}_\theta\mathbf{K}\mathbf{f}_{R,x} + l\mathbf{C}_\theta\mathbf{K}\mathbf{f}_{R,y} = \mathbf{D}^T\mathbf{u}, \tag{D.24a}$$

$$Nm\ddot{\boldsymbol{p}} = \begin{bmatrix}\ddot{p}_x \\ \ddot{p}_y\end{bmatrix} = Nm\begin{bmatrix}\ddot{p}_x \\ \ddot{p}_y\end{bmatrix} = \begin{bmatrix}\boldsymbol{e}^T\mathbf{f}_{R,x} \\ \boldsymbol{e}^T\mathbf{f}_{R,y}\end{bmatrix}. \tag{D.24b}$$

Där de nya variablerna är

$$\mathbf{M}_\theta = J\mathbf{I}_N + ml^2\mathbf{S}_\theta\mathbf{V}\mathbf{S}_\theta + ml^2\mathbf{C}_\theta\mathbf{V}\mathbf{C}_\theta, \tag{D.25a}$$

$$\mathbf{W} = ml^2\mathbf{S}_\theta\mathbf{V}\mathbf{C}_\theta - ml^2\mathbf{C}_\theta\mathbf{V}\mathbf{S}_\theta, \tag{D.25b}$$

$$\mathbf{V} = \mathbf{A}^T(\mathbf{D}\mathbf{D}^T)^{-1}\mathbf{A}, \tag{D.25c}$$

$$\mathbf{K} = \mathbf{A}^T(\mathbf{D}\mathbf{D})^{-1}\mathbf{D}. \tag{D.25d}$$

Genom att introducera tillståndsvariabeln $\mathbf{x} = [\boldsymbol{\theta}^T, \boldsymbol{p}^T, \dot{\boldsymbol{\theta}}^T, \dot{\boldsymbol{\theta}}^T]^T \in \mathbb{R}^{2n+4}$, kan modellen skrivas om till en mer kompakt form med tillståndsformeln

$$\dot{\mathbf{x}} = \begin{bmatrix}\dot{\boldsymbol{\theta}} \\ \dot{\mathbf{p}} \\ \ddot{\boldsymbol{\theta}} \\ \ddot{\mathbf{p}}\end{bmatrix} = \mathbf{F}(\mathbf{x},\mathbf{u}), \tag{D.26}$$

där elementen i $\mathbf{F}(\mathbf{x},\mathbf{u})$ lätt kan hittas genom att lösa ut $\ddot{\boldsymbol{\theta}}$ och $\ddot{\mathbf{p}}$ i ekvation D.24.

D.1.7 Separering av drivande och ickedrivande dynamik

Den modell som togs fram i slutet på förra kapitlet, ekvation D.24, är svår att analysera och implementera styrning på. Målet är därför att skriva om modellen på en enklare form. *Partial feedback linearsation of underactuated system* är en metod vilken gör dynamiken linjär kring systemets drivande frihetsgrader. Det första steget för detta är att separera ickedrivande och drivande frihetsgrader vilket kommer gås igenom i detta kapitel. I nästa kapitel tas den linjära återkopplingen fram.

För att kunna separera drivande och ickedrivande frihetsgrader måste de definieras. Accelerationen på MC, $\ddot{\mathbf{p}}$, tillhör ickedrivande delen då den inte direkt kan styras av motorerna, det vill säga styrsignalen $\mathbf{u}$. Däremot består vinkelaccelerationen på länkarna, $\ddot{\boldsymbol{\theta}}$, av en ickedrivande frihetsgrad och $N-1$ drivande frihetsgrader. Detta då det finns N länkar ($\boldsymbol{\theta} \in \mathbb{R}^N$) och $N-1$ motorer, en i varje led ($\mathbf{u} \in \mathbb{R}^{N-1}$). Dock går det inte att dela upp ekvationen för $\ddot{\boldsymbol{\theta}}$ i en drivande och ickedrivande del då $\mathbf{u}$ direkt påverkar alla länkars vinkelaccelerationer. Därför söks en form där bara $N-1$ av länkarna direkt påverkas av $\mathbf{u}$. Genom att ändra de generella koordinaterna (detsamma som frihetsgraderna) för systemet från länkarnas vinklar, $\boldsymbol{\theta}$, och MC:s position ,$\mathbf{p}$, till

$$\mathbf{q}_\phi = \begin{bmatrix} \bar{\boldsymbol{\phi}} \\ \mathbf{p} \end{bmatrix} \in \mathbb{R}^{N+2}, \tag{D.27}$$

där $\bar{\boldsymbol{\phi}} = [\phi_1, \ldots ,\phi_{N-1},\theta_N]^T \in \mathbb{R}^N$ innehåller alla $N-1$ ledvinklar samt sista länkens vinkel. Transformationen mellan länkvinklarna och ledvinklarna ges av

$$\boldsymbol{\theta} = \mathbf{H}\bar{\boldsymbol{\phi}}, \quad \mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 & 1 \\ 0 & 1 & 1 & \ldots & 1 & 1 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \ldots & 0 & 1 \end{bmatrix} \in \mathbb{R}^{N\times N}. \tag{D.28}$$

Modellen med de nya koordinaterna fås fram genom att sätta in ekvation D.27 i D.24 vilket ger

$$\mathbf{M}_\theta \mathbf{H}\ddot{\bar{\boldsymbol{\phi}}} + \mathbf{W}diag(\mathbf{H}\dot{\bar{\boldsymbol{\phi}}})\mathbf{H}\ddot{\bar{\boldsymbol{\phi}}} - l\mathbf{S}_\theta \mathbf{K}\mathbf{f}_{R,x} + l\mathbf{C}_\theta \mathbf{K}\mathbf{f}_{R,y} = \mathbf{D}^T\mathbf{u}, \tag{D.29a}$$

$$Nm\ddot{\mathbf{p}} = \begin{bmatrix} \boldsymbol{e}^T\mathbf{f}_{R,x} \\ \boldsymbol{e}^T\mathbf{f}_{R,y} \end{bmatrix}. \tag{D.29b}$$

Där det utnyttjas att $\dot{\boldsymbol{\theta}}^2 = diag(\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} = diag(\mathbf{H}\dot{\bar{\boldsymbol{\phi}}})\mathbf{H}\dot{\bar{\boldsymbol{\phi}}}$. Genom att multiplicera ekvation D.29 med $\mathbf{H}^T$ fås en önskad form på matrisen framför $\mathbf{u}$ så att bara lederna, de $N-1$

första elementen i $\bar{\boldsymbol{\phi}}$, blir beroende av styrsignalen. Detta visas med

$$\mathbf{H}^T\mathbf{D}^T\mathbf{u} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & & & & \vdots & \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N-2} \\ u_{N-1} \\ 0 \end{bmatrix}. \tag{D.30}$$

Det är möjligt att skriva den färdiga modellen av roboten som

$$\overline{\mathbf{M}}(\bar{\boldsymbol{\phi}})\ddot{\mathbf{q}}_\phi + \overline{\mathbf{W}}(\bar{\boldsymbol{\phi}},\dot{\bar{\boldsymbol{\phi}}}) + \overline{\mathbf{G}}(\bar{\boldsymbol{\phi}})\mathbf{f}(\bar{\boldsymbol{\phi}},\dot{\bar{\boldsymbol{\phi}}},\dot{\mathbf{p}}) = \overline{\mathbf{B}}\mathbf{u}, \tag{D.31}$$

där

$$\mathbf{q}_\phi = \begin{bmatrix} \bar{\boldsymbol{\phi}} \\ \mathbf{p} \end{bmatrix}, \tag{D.32a}$$

$$\overline{\mathbf{M}}(\bar{\boldsymbol{\phi}}) = \begin{bmatrix} \mathbf{H}^T\mathbf{M}_\theta(\bar{\boldsymbol{\phi}})\mathbf{H} & \mathbf{0}_{N\times 2} \\ \mathbf{0}_{N\times 2} & Nm\mathbf{I}_2 \end{bmatrix}, \tag{D.32b}$$

$$\overline{\mathbf{W}}(\bar{\boldsymbol{\phi}},\dot{\bar{\boldsymbol{\phi}}}) = \begin{bmatrix} \mathbf{H}^T\mathbf{W}diag(\mathbf{H}\dot{\bar{\boldsymbol{\phi}}})\mathbf{H}\ddot{\bar{\boldsymbol{\phi}}} \\ \mathbf{0}_{2\times 1} \end{bmatrix}, \tag{D.32c}$$

$$\overline{\mathbf{G}}(\bar{\boldsymbol{\phi}}) = \begin{bmatrix} -l\mathbf{H}^T\mathbf{S}_{H\bar{\phi}}\mathbf{K} & l\mathbf{H}^T\mathbf{C}_{H\bar{\phi}}\mathbf{K} \\ -\mathbf{e}^T & \mathbf{0}_{1\times N} \\ \mathbf{0}_{1\times N} & -\mathbf{e}^T \end{bmatrix}, \tag{D.32d}$$

$$\overline{\mathbf{B}} = \begin{bmatrix} \mathbf{I}_{N-1} \\ \mathbf{0}_{3\times(N-1)} \end{bmatrix}, \tag{D.32e}$$

och där $\mathbf{S}_{H\bar{\phi}} = \mathbf{S}_\theta$ och $\mathbf{C}_{H\bar{\phi}} = \mathbf{C}_\theta$.

För den färdiga modellen, D.31, representerar de $N-1$ första ekvationerna dynamiken för ledvinklarna, det vill säga de styrbara frihetsgraderna för robotormen. De tre sista ekvationerna representerar i sin tur dynamiken för robotormens position och riktning, det vill säga de icke styrbara frihetsgraderna. Ekvation D.31 kan alltså delas upp en ekvation som representerar de styrbara frihetsgraderna och en för de ickestyrbara frihetsgraderna enligt

$$\overline{\mathbf{M}}_{11}\ddot{\mathbf{q}}_a + \overline{\mathbf{M}}_{12}\ddot{\mathbf{q}}_a + \overline{\mathbf{W}}_1 + \overline{\mathbf{G}}_1\mathbf{f}_R = \mathbf{u}, \tag{D.33a}$$

$$\overline{\mathbf{M}}_{21}\ddot{\mathbf{q}}_a + \overline{\mathbf{M}}_{22}\ddot{\mathbf{q}}_a + \overline{\mathbf{W}}_2 + \overline{\mathbf{G}}_2\mathbf{f}_R = \mathbf{0}_{3\times 1}, \tag{D.33b}$$

där $\mathbf{q}_a = [\phi_1, \dots, \phi_{N-1}]^T \in \mathbb{R}^{N-1}$ representerar de styrbara frihetsgraderna och $\mathbf{q}_u = [\theta_N, \dots, p_x, p_y]^T \in \mathbb{R}^3$ representerar de ickestyrbara frihetsgraderna. $\overline{\mathbf{M}}_{11} \in \mathbb{R}^{(N-1)\times(N-1)}$, $\overline{\mathbf{M}}_{12} \in \mathbb{R}^{(N-1)\times 3}$, $\overline{\mathbf{M}}_{21} \in \mathbb{R}^{3\times(N-1)}$, $\overline{\mathbf{M}}_{22} \in \mathbb{R}^{3\times 3}$, $\overline{\mathbf{W}}_1 \in \mathbb{R}^{N-1}$,$\overline{\mathbf{W}}_2 \in \mathbb{R}^3$, $\overline{\mathbf{G}}_1 \in \mathbb{R}^{(N-1)\times 2N}$ och $\overline{\mathbf{G}}_2 \in \mathbb{R}^{3\times 2N}$.

D.1.8 Linjär återkoppling av modellen

Utifrån de två uppdelade ekvationerna i D.33 kommer en förenkling av modellen göras genom linjär återkoppling. Tanken är att de styrbara frihetsgraderna ska kunna regleras. Förutom att förenkla modellen är reglering av ledernas vinklar även en önskvärd egenskap hos det riktiga systemet. Målet är att få de styrbara frihetsgraderna, $\ddot{\mathbf{q}}_a$, att vara direkt styrda av nya styrsignaler och de ickestyrbara frihetsgraderna, $\ddot{\mathbf{q}}_u$, vara en funktion av systemets tillstånd och de nya styrsignalerna.

Börja med att lösa ut $\ddot{\mathbf{q}}_u$ ur D.33b

$$\ddot{\mathbf{q}}_u = -\overline{\mathbf{M}}_{22}^{-1}(\overline{\mathbf{M}}_{21}\ddot{\mathbf{q}}_a + \overline{\mathbf{W}}_2 + \overline{\mathbf{G}}_2\mathbf{f}_R), \tag{D.34}$$

och stoppa in D.34 i D.33a ger

$$(\overline{\mathbf{M}}_{11} - \overline{\mathbf{M}}_{12}\overline{\mathbf{M}}_{22}^{-1}\overline{\mathbf{M}}_{21})\ddot{\mathbf{q}}_a + \overline{\mathbf{W}}_1 + \overline{\mathbf{G}}_1\mathbf{f}_R - \overline{\mathbf{M}}_{12}\overline{\mathbf{M}}_{22}^{-1}(\overline{\mathbf{W}}_2 + \overline{\mathbf{G}}_2\mathbf{f}_R) = \mathbf{u}. \tag{D.35}$$

Genom att skapa följande linjära styrkontroll

$$\mathbf{u} = (\overline{\mathbf{M}}_{11} - \overline{\mathbf{M}}_{12}\overline{\mathbf{M}}_{22}^{-1}\overline{\mathbf{M}}_{21})\bar{\mathbf{u}} + \overline{\mathbf{W}}_1 + \overline{\mathbf{G}}_1\mathbf{f}_R - \overline{\mathbf{M}}_{12}\overline{\mathbf{M}}_{22}^{-1}(\overline{\mathbf{W}}_2 + \overline{\mathbf{G}}_2\mathbf{f}_R), \tag{D.36}$$

där $\bar{\mathbf{u}} = [\bar{u}, \ldots, \bar{u}_{N-1}]^T \in \mathbb{R}^{N-1}$ är de nya styrsignalerna. Detta gör det möjligt att skriva om D.33 som

$$\ddot{\mathbf{q}}_a = \bar{\mathbf{u}}, \tag{D.37a}$$
$$\ddot{\mathbf{q}}_u = \mathcal{A}(\mathbf{q}_\phi, \dot{\mathbf{q}}_\phi) + \mathcal{B}(\mathbf{q}_a)\bar{\mathbf{u}}, \tag{D.37b}$$

där

$$\mathcal{A}(\mathbf{q}_\phi, \dot{\mathbf{q}}_\phi) = -\overline{\mathbf{M}}_{22}^{-1}(\overline{\mathbf{W}}_2 + \overline{\mathbf{G}}_2\mathbf{f}_R) \in \mathbb{R}^3, \tag{D.38a}$$
$$\mathcal{B}(\mathbf{q}_a) = -\overline{\mathbf{M}}_{22}^{-1}\overline{\mathbf{M}}_{21} \in \mathbb{R}^{3\times(N-1)}. \tag{D.38b}$$

Denna modell kan skrivas i standardform för ett så kallat *control-affine system* genom att definiera $\mathbf{x}_1 = \mathbf{q}_a$, $\mathbf{x}_2 = \mathbf{q}_u$, $\mathbf{x}_3 = \dot{\mathbf{q}}_a$, $\mathbf{x}_4 = \dot{\mathbf{q}}_u$ och $\mathbf{x} = [\mathbf{x}_1^T, \mathbf{x}_2^T, \mathbf{x}_3^T, \mathbf{x}_4^T]^T \in \mathbb{R}^{2N+4}$ som ger

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \\ \dot{\mathbf{x}}_3 \\ \dot{\mathbf{x}}_4 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_3 \\ \mathbf{x}_4 \\ \bar{\mathbf{u}} \\ \mathcal{A}(\mathbf{x}) + \mathcal{B}(\mathbf{x}_1)\bar{\mathbf{u}} \end{bmatrix} = \mathbf{f}(\mathbf{x}) + \sum_{j=1}^{N-1}(\mathbf{g}_j(\mathbf{x}_1)\bar{u}_j), \tag{D.39}$$

där

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{0}_{(N-1)\times 1} \\ \mathcal{A}(\mathbf{x}) \end{bmatrix}, \quad \mathbf{g}_j(\mathbf{x}_1) = \begin{bmatrix} \mathbf{0}_{(N-1\times 1)} \\ \mathbf{0}_{3\times 1} \\ \mathbf{e}_j \\ \mathcal{B}_j(\mathbf{x}_1) \end{bmatrix}. \tag{D.40}$$

$\mathbf{f}(\mathbf{x})$ kallas *drift vector field* och $\mathbf{g}_j(\mathbf{x}_1)$ kallas *control vector field*. För praktisk analys av modellens rörelse med en given styrsignal ska ekvation D.39 lösas och för att sedan undersöka det faktiska momentet på lederna, det vill säga på motorerna, ska ekvation D.36 användas.