Modellering och reglering av en oktakopter

Henrik Ohlsson
Hrvoje Corluka

LUNDS UNIVERSITET
Institutionen för Reglerteknik

Msc Thesis
ISRN LUTFD2/TFRT--5935--SE
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Media-Tryck
Lund 2014

Abstract

This report deals with the theory and identification out of a multi-rotor craft, particularly an OktokopterXL from Mikrokopter.de.

A model of the craft is constructed using Matlab Simulink and a PD controller is designed to control the attitude of the craft. The controller is auto generated from Matlab to C code which is then interwoven with the multi-rotor's source code. The designed controller works well and also the model corresponds well with reality.

The multirotor platform used is not very developer-friendly with hard to understand, uncommented code and where the navigation board source code is not open source. However, the price and performance still make it a very
interesting platform.

Sammanfattning

Det här examensarbetet behandlar teori för identifiering och reglering av en multirotorfarkost, närmre bestämt en Octokopter XL från Mikrokopter.de.

En modell av farkosten konstrueras med hjälp av Matlab Simulink och med hjälp av denna så designas en PD-regulator för att styra attityden av farkosten. Regulatorn kodgenereras från Matlab till C-kod som sedan integreras med farkostens källkod. Regulatorn som designas fungerar bra och även modellen visar sig stämma bra överens med verkligheten.

Plattformen som oktakoptern bygger på är dock inte så utvecklarvänlig med allmänt svårbegriplig okommenterad kod och där navigationskortets källkod inte är öppen. Däremot så gör priset och grundprestandan det till en mycket intressant plattform.

Tack

Vi vill först tacka Jan Ohlsson, Torbjörn Crona och Per Jonsson som gav oss möjligheten att få detta examensarbete.

Därefter vill vi tacka alla på Guidance, Navigation & Control avdelningen på Saab Dynamics där vi utfört arbetet, fått hjälp och handledning. Speciellt tack till Anders Peterson, Stefan Thorstensson, Carl Nordheim, Björn Johansson, Franz Hoffman, Fredrik Neregård, Henrik Jonsson och Ulla-Mona Mattsson.

Ett stort tack till Karl Erik Årzen som har varit examinator.

Sist men inte minst ett stort tack till Jolon och Tina som har hjälpt oss med boende, bil med mera när det krisat.

Ordlista

Attityd – farkostens vinklar i luften
Borstlös motor – i en borstlös-motor är permanentmagneten roterande och spolarna sitter i statorn. Detta innebär att man slipper mekaniskt slitage och oftast får man även lättare motorer med bättre vridmoment och högre effektivitet.
DIY – Do It Yourself, gör det själv paket
ESC – Electronic Speed Controller
GPS- Global Positioning System
Hovra – sväva över en punkt
Klinometer – mätverktyg för att mäta vinklar
Kvaternion - fyrdimensionellt talsystem definierat av W.R. Hamilton 1843

Innehåll

1. Introduktion

Examensarbetet ”Styrning och reglering av Oktakopter” har utförts på SAAB Dynamics GNC (Guidance, Navigation and Control) i Linköping med Lunds Universitet som ansvarigt lärosäte.

1.1 Bakgrund

SAAB Dynamics system- och styrningsavdelning beslutade sig för att köpa in en multirotorfarkost för att använda som en exjobbs- och experiment-plattform.

En multirotorfarkost är en farkost med fler än två rotorer. Fler rotorer ger upphov till ökad stabilitet, robusthet och ökad lyftkapacitet. De vanligast förekommande är quadrokopter (fyra rotorer), hexakopter (sex rotorer) och oktakopter (åtta rotorer). Multirotorfarkoster är väldigt populära sedan några år tillbaka, professionellt används de till att fotografera och spela in video där de är betydligt billigare och mycket smidigare än att hyra en helikopter för att t.ex. få en flygvy över ett utomhusevenemang.

Ett annat användningsområde är för att inspektera svåråtkomliga platser och föremål som till exempel vindkraftverk där man med realtidsbilder från kameran kan flyga och inspektera vindkraftverket från marken.

Hobbyfantaster är en annan stor användargrupp där man bygger ihop egna lösningar. Där finns många DIY (Do It Yourself) paket där man utgår från grundkomponenter och bygger egna ramar och designar egna regulatorer.

Plattformen har flera intressanta delar som tilltalar hemmabyggare då det ses som en kul utmaning och där man i slutändan bygger ihop en väldigt rolig leksak.

1.2 Syfte/Mål

Grundmålet med examensarbetet var att ta fram en Simulinkmodell av farkosten och en egen regulatordesign. Detta skulle göras genom att först modellera farkosten i Simulink och sedan autogenerera kod för just styrningsregulatorn till farkosten.

I Matlab finns det ett kodgenereringspaket och genom att bygga Simulinkmodellen på ett visst sätt kan man generera C-kod direkt från modellen. Den genererade koden skulle sedan integreras med befintlig kod och verifieras så att de modellframtagna parametrarna i regulatorn fungerade även då farkosten flög. Utvärdering av arbetsmetodiken låg i SAABs intresse för att undersöka om de i framtiden skulle kunna använda sig av automatgenererad C kod vilket skulle kunna spara mycket utvecklingstid.

1.3 Individuell uppdelning

Både Hrvoje och Henrik inledde arbetet med litteratursökning och fortsatte sedan med modellframtagning. Förseningar med leveransen av oktakoptern medförde att en större uppdelningen inte kunde göras till en början. När farkosten kom byggde Henrik ihop den, Hrvoje fokuserade på att få all mjukvara som följer med att fungera. Henrik fokuserade sedan på att förstå den medföljande koden medan Hrvoje återgick till modellbygge och vinkelregulatordesign. Generellt fokuserade Hrvoje mer på simulering i Simulink medan Henrik fokuserade på testning i den verkliga farkosten. Henrik designade hastighetsregulatorn och båda hjälptes åt att skriva rapporten.

De problem som stöttes på under arbetets gång medförde att vi fick hjälpas åt väldigt mycket och det underlättade att man var två som kunde hjälpas åt när man satt fast.

1.4 Rapportens struktur

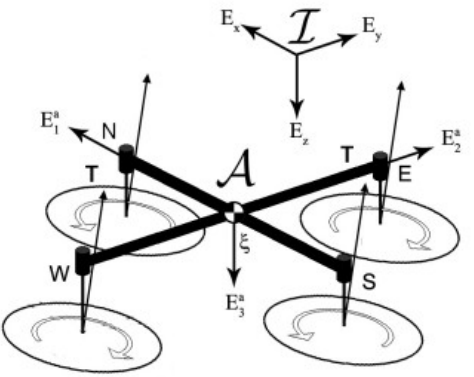
Rapporten börjar med att beskriva teorin kring multirotorer, sensorer och regulatorn i kapitel 2. Plattformen och farkosten som användes under detta examensarbete presenteras i kapitel 3. Kapitel 4 handlar om modellen som togs fram i Simulink, och regulatorerna som designades med hjälp av modellen. Resultatet presenteras och analyseras i kapitel 5 och rapporten avslutas med diskussion och slutsatser i kapitel 6.

2. Teori

2.1 Representation

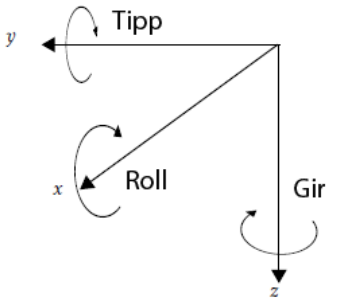
När man beskriver en kropps rörelse och position så gör man det lämpligen relativt något inertiellt (stillastående). Om det inte finns bestämda definitioner på olika referenspunkter kan det lätt uppstå missförstånd.

Ett inertiellt koordinatsystem är ett fixt koordinatsystem till vilket en farkosts position och attityd kan mätas relativt. I flygsammanhang väljs detta system som ett nord-öst-ned koordinatsystem, där x-axeln pekar positivt mot norr, y-axel pekar positivt mot öst och z-axeln pekar positivt ner mot jordens mittpunkt. Inertiella koordinatsystemet betecknas { } där är riktningen på gravitationen och { } är origo utryckt i det kroppsfasta koordinatsystemet { } som farkosten representeras med. I detta koordinatsystem är positiv i farkostens riktning framåt, är positiv i högerriktning och positiv i riktning nedåt. (Bermes 2010) (Hua, o.a. 2013) (Pounds 2007)



Figur 2.1 En quadrokopters representation

Att beskriva en farkosts rotation med avseende på det inertiella koordinatsystemet kan göras på olika sätt men konventionen inom flyg är Eulervinklar i roll-, tipp-, och gir-vinklar. Detta görs genom 3 rotationer kring koordinataxlarna i farkostens koordinatsystem. Genom att först utföra rotationen runt z-axeln, sedan rotationen runt y-axeln och till sist rotationen kring x-axeln så erhålls Eulervinklarna, . När dessa transformeras från inertiella till kroppsfasta koordinatsystemet så kallas de gir( ), tipp( ) och roll ( ). Figur 2.2 illustrerar rotationerna runt de tre axlarna.



Figur 2.2 Gir(ψ), tipp(θ) och roll (ϕ), x är framåt på farkosten

Då en vektor ska transformeras från att vara uttryckt i ett inertiellt koordinatsystem till ett kroppsfast gäller följande:

( ) ( ) (2.1)

där är produkten av 3 olika rotationsmatriser (en för varje axel)

( ) ( ) (2.2)

( ) ( ) (2.3)

( ) ( ) (2.4)

( ) ( ) ( )

( ) (2.5)

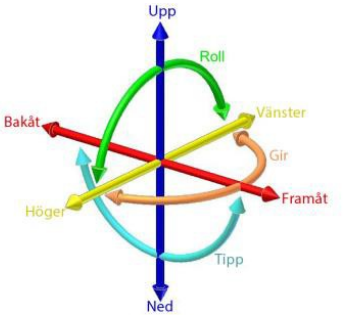
För att gå från det andra hållet, dvs. transformera från det kroppsfasta till det inertiella koordinatsystemet så inverteras . I och med att alla tre rotationsmatriser, (2.2),(2.3),(2.4), är ortonormerade så är även ortonormerad. Inversen till en ortonormerad matris är samma som matrisens transponat.

( ) ( ) (2.6)

Eulervinklar är inte fullständiga, de är odefinierade för −. Detta kan lösas genom att använda kvaternioner men resultatet med kvaternioner blir svårtolkat då däremot Eulervinklar är väldigt intuitiva. Inga planer på att göra loopar med oktakoptern fanns, dvs. inga vinklar nära −så därför användes Eulervinklar.

2.2 Multirotorer - flygdynamik

En multirotor utgör ett underaktuerat system då den direkt kan styra endast 4 av 6 frihetsgrader. De olika frihetsgraderna illustreras i figur 2.3. Det är möjligt att styra roll- tipp- och gir-vinklarna samt positionen i höjdled. Det är däremot inte möjligt att direkt förflytta sig i horisontalplanet.



Figur 2.3 Illustration av de 6 frihetsgraderna

Att styra farkosten i horisontalled utförs genom att ändra attityden (vinkla farkosten) i roll och tippled. Farkosten hålls hovrande genom att grundgas ges

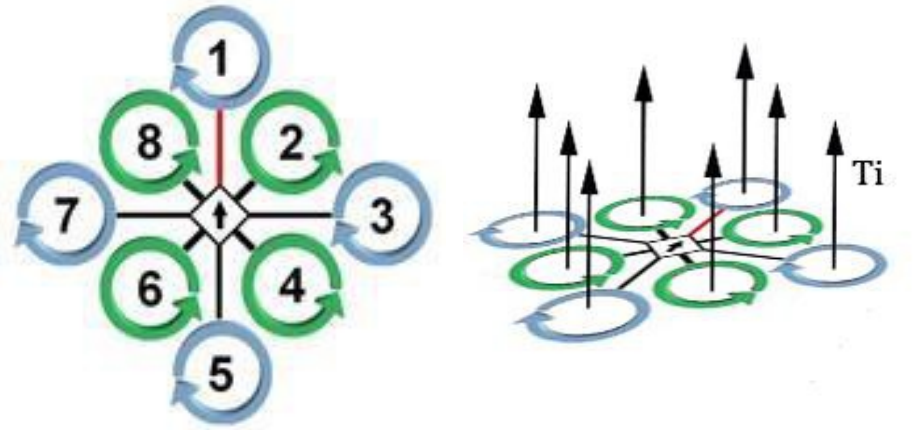
som motverkar gravitationskrafter. Då attityden i roll- och tippled ändras så kommer acceleration i sidled att genereras av den då vinklade, snett uppåtriktade, kraften.

Exempel: För att farkosten ska förflytta sig framåt så låter man de 3 främre rotorerna rotera lite långsammare, de 3 bakre rotera lite snabbare och de 2 på sidorna vara oförändrade. Detta medför att farkosten vinklas och samtidigt uppstår en kraft i horisontalled vilket i sin tur leder till förflyttning i horisontalplanet. Valet att låta de bakre rotorerna rotera snabbare samtidigt som de främre roterar långsammare görs för att erhålla en snabbare vinkeländring och samtidigt ha oförändrad kompensering för gravitationskraften. Skulle man låta bara de främre rotera långsammare skulle man snabbt tappa höjd och om man bara skulle låta de bakre rotera snabbare skulle man först få en ökning i höjdled följt av att även där tappa i höjd. När man låter både de främre och de bakre samverka får man ett snabbare system för att uppnå önskad vinkel, höjdtappet finns kvar men blir inte lika kraftigt som om man endast minskar rotorerna.

Dynamiken hos en multirotor leder till att det mest naturliga sättet att styra farkosten är att styra den med 2 spakar som kan röra sig i både x- och y-led.
Exempel: Man låter höger spak motsvara farkostens tipp- och rollvinkel. Förs spaken framåt så kommer farkosten luta framåt, och rör man spaken i sidled så kommer den att luta i sidled. Farkosten lutar proportionellt mot spakrörelsen och rör sig därmed snabbare i horisontalled med ökat utslag. Den andra spaken styr summan av varvtalet på rotorerna, vilket motsvarar acceleration upp och ner. Vänster- och högerförskjutning på den spaken motsvarar ändring i girvinkel.

Figur 2.4 Motorerna på farkosten roterar åt olika håll

Farkosten modelleras som en stelkropp och flygdynamiken blir ett samband av den lyftkraft i varje motor genererar, det luftdrag som bildas från motorerna i motsatt riktning och tyngdkraften på själva farkosten. För att få balans i farkostens girmoment roterar motorerna åt olika håll enligt figur 2.4



Figur 2.5 Motorernas rotationshåll och numrering samt illustration av dragkraftsvektorn Ti, där i är motorns nummer.

Varje rotor bildar en dragkraftsvektor som är riktad uppåt. Vektorns storlek för varje motor *i*, där *i* motsvarar motorns numrering med början på den motor som är riktad framåt, se figur 2.5, beskrivs enligt formeln:

(2.7)

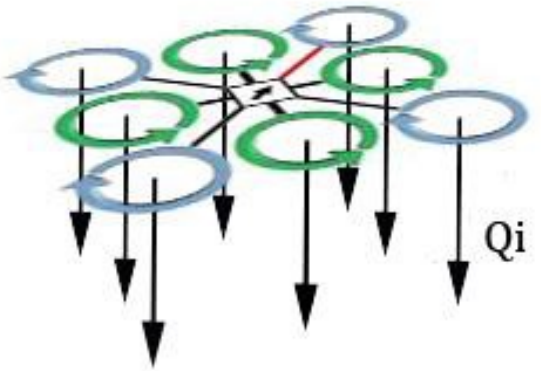
där är rotorhastigheten och är en lyftkonstant som förklaras mer i detalj senare. (Hossain, Rideout och Krouglicof 2010) (Hamel, o.a. 2002) (Mahony, Kumar och Corke September 2012) (Luukkonen 2011)

Den translationella dynamiken i inertiella koordinatsystemet kan beskrivas med följande formler

( ) ( ) (2.8)

där är den totala massan för farkosten, är farkostens hastighet i inertiella koordinatsystemet, är gravitationskonstanten och är den totala lyftkraften uppåt från rotorerna. Den första termen är gravitationen som är riktad

ned i inertiella koordinatssystemet och den andra termen är den totala lyftkraften roterad till samma koordinatsystem.



Figur 2.6 Vektorn Qi för varje motor.

Vridmomentet för varje motor motverkas av luftmotståndet i propellern, se figur 2.6, och kan beskrivas enligt:

(2.9)

där är en konstant som likt förklaras senare.

För att förändra farkostens attityd varieras varvtalet på motorerna för att på så vis skapa ett vridmoment runt respektive axel. Styrning i x-led dvs. framåt och bakåt görs genom att styra vinkeln mot y-axeln, tippvinkeln. Det vridmoment runt y-axeln som behövs beräknas enligt formeln:

( ) ( )
(( )) (2.10)

där är avståndet mellan rotor och fakostens masscentrum Motsvarande beräkningar utförs för roll, runt x-axeln.

( ) ( )
(( )) (2.11)

För att uppnå en rotation runt Z-axeln (gir) ökas respektive minskas varvtalet på de motorer som roterar åt samma håll, detta för att inte påverka den totala lyftkraften.

( ) (2.12)

För att generera ytterligare moment i girled och därmed erhålla snabbare svar på givet kommando är motorerna tiltade 3 grader. De motorer som roterar medurs åt ett håll och de som roterar moturs åt det andra. Detta gör ett utfört girkommando mer kraftfullt.

Lyftkraften från varje motor som behövs för att farkosten ska hovra följer formeln:

√ (2.13)

där är rotorhastigheten, farkostens totala vikt, gravitationskonstanten, antalet rotorer och lyftkonstanten.

För att styra höjden ökas respektive minskas varje motors varvtal. Varje enskild motors totala bidrag består av upp-, roll- tipp- och gir-bidrag. För att påverka höjden är det endast upp-bidraget som ändras. Vid hovring är upp-bidraget det enda som påverkar och det normala är att börvärdet för upp-ner-spaken i neutralt läge är just det som krävs för att farkosten ska hovra.

Eulers rörelseekvation beskriver rotationsaccelerationen av kroppen enligt

(2.14)

där är en 3x3 tröghetsmomentsmatris, är vinkelhastighetsvektorn { } och ( ) . Vinkelhastighetsvektorn bestämmer sedan Eulervinklarnas derivata, för små vinklar gäller att , , och .

Det totala sambandet mellan vridmoment, krafter och rotorhastighet ges av:

( ) ( ) (2.15)

( )

där är avståndet mellan rotor och farkostens masscentrum samt är rotorhastigheten i kvadrat.

Efter omskrivning kan ett samband för önskat varvtal tas fram. I implementationen skickas däremot inte kommandon för ett varvtal utan en spänning. Sambandet har därför skrivits om för att motsvara den önskade spänningen.

( )

(2.16)

( ) ( )

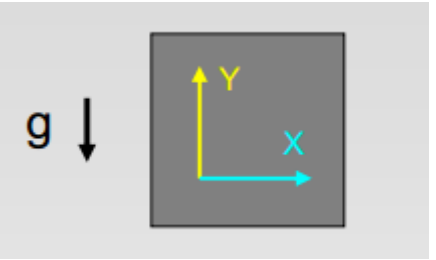
2.3 Sensorer

Som för de flesta balanserande plattformar så är huvudsensorerna accelerometrar och gyroskop. På FlightCtrl-kortet så finns tre accelerometrar och tre gyroskop, en för varje dimension. Regulatorn behöver aktuell vinkel och vinkelhastighet för att räkna ut styrkommando.

För att mäta detta så används gyroskop och accelerometer. I en ideal värld så är värdena som läses av från sensorerna helt korrekta utan fel. I verkligheten så är det inte så utan båda sensorerna har fel och går inte att lita på blint.

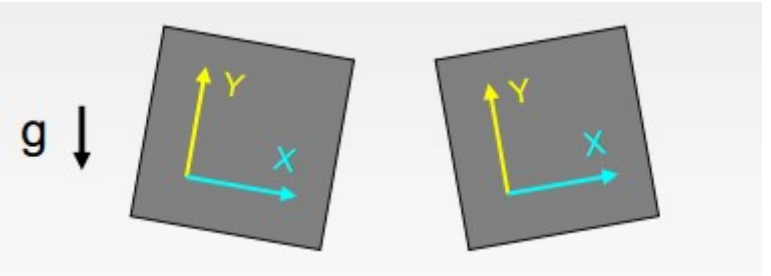
Accelerometer

En accelerometer mäter acceleration, eller egentligen kraft delat med massa enligt Newtons formel: —



Figur 2.7 Mätning i neutralt läge

I figur 2.7 så mäter accelerometern 0 i x-led och den negativa gravitationsaccelerationen i y-led.

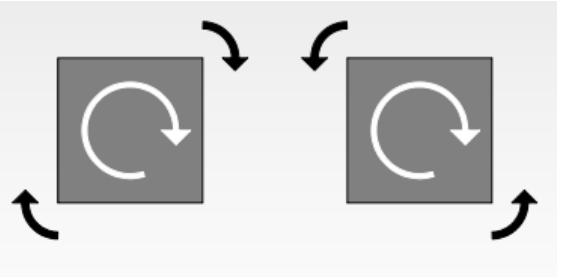


Figur 2.8 Mätning vid rotation

I figur 2.8 märks gravitationen i x-axeln och den är något positiv i figuren till vänster på bilden och något negativ i figuren till höger. Detta gör att man med hjälp av trigonometri kan räkna ut vinkeln. Värdet för y-axeln är egentligen inte intressant då den inte ger någon information om vinkeln är positiv eller negativ.

Gyroskop

Gyroskop mäter vinkelhastighet eller rotationshastighet, snurrar man åt ett håll så ger den ett positivt värde, snurrar man åt andra hållet så ger den ut ett negativt värde och när farkosten är stilla så ger den värdet 0. Figur 2.9 visar en principskiss över ett gyroskop som bara ger utslag när den roterar.



Figur 2.9 Mätning med gyroskop

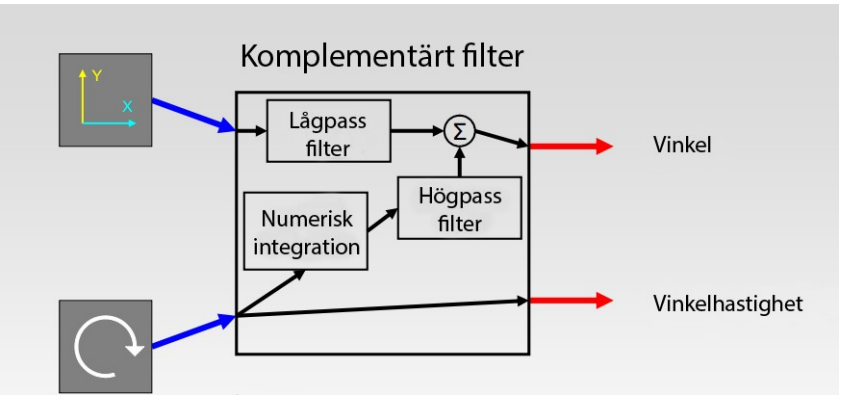
Det mest uppenbara är att använda data från accelerometern till att skatta vinkel- och gyroskopsdata för att skatta vinkelhastighet. Problemet med detta är att rådata är väldigt brusig och x-axeln mäter av all form av ändring i horisontell acceleration som ändring på vinkel. Har man en plattform som står horisontellt och har motorer som får den att accelerera framåt så kan inte accelerometern urskilja detta från gravitationen. Det finns lite olika sätt att lösa dessa problem, man kan t.ex. sätta ett lågpassfilter på gyroskopsdata för ta bort brus. Då får man däremot en fördröjning av vinkelskattningen pga filtreringen, större fördröjning desto mer man filtrerar och fördröjningar i system leder till ett mer komplext system.

Ett annat sätt är att bara integrera signalen från gyroskopet för att skatta vinkeln, men då får man problem med drift. Ett riktigt gyroskop returnerar aldrig 0 när det står still och då kommer detta lilla fel att adderas till vinkeln vilket i sin tur kommer medföra att skattningen till slut kommer att vara långt ifrån den riktiga vinkeln.

Kalmanfilter och komplementära filter

Kalmanfilter använder sig av en matematisk modell av systemet. Då man känner till insignalerna till systemet och mäter utsignalerna med sensorerna så kan man använda sig av flera mätningar för att skatta tillstånd så som vinkel och vinkelhastighet. Detta ger en mycket bättre skattning än bara en mätning av en brusig sensorsignal.

Kalmanfiltret skattar det nya tillståndet med ett viktat medelvärde med hjälp av den nya sensorsignalen och vad den matematiska modellen tror vinkeln är. Viktningen räknas fram av kovariansen som är ett mått på hur osäker skattningen är, om en sensorsignal är väldigt brusig så litar man mer på den matematiska modellen men om den är mindre brusig så får själva sensorsignalen en större inverkan på skattningen. Nackdelen med Kalmanfilter är att det är mer komplext och kräver tyngre matematisk beräkning och ofta är inbyggda system plattformar processorsvaga vilket gör att beräkningarna kan bli för krävande.



Figur 2.10 Illustration av ett komplementärt filter

Ett komplementärt filter använder sig av både informationen från gyroskopet och accelerometern för att skatta vinkeln. Figur 2.10 visar en illustration av ett komplementärt filter. Integrering av gyroskopsdata fungerar bra för skatta vinklar under kort tid, men över längre tid så vet man att gyroskopsskattningen kommer driva iväg. För att hantera detta använder man accelerometern till att skatta vinkeln och sedan återställer man gyroskopet genom att förutsätta att accelerationen är låg.

Enklaste formen på ett komplementärt filter kan se ut på formen nedan

( ) ( ) (2.17)

där är en designparameter. Uttrycket inom första parentesen i formeln är integration av gyroskopsdata tillsammans med föregående vinkel dvs. vad gyroskopet tror att vinkeln är och är vad accelerometern tror vinkeln är. Bidraget från de båda sensorerna viktas sedan till ett skattat värde där man då inte kommer ha problem med drift och samtidigt kommer ha en bättre vinkelskattning pga. gyroskopets acceptans av externa krafter till skillnad från accelerometern. (Colton 2007) (Van de Maele u.d.)

När komplementära filter implementeras testas ofta ifall värdet från accelerometern är inom rimligt intervall i närheten av gravitationen. Om det inte är det så använder man bara gyroskopsskattningen för det tidsteget.

Kalman- och komplementära filter är praxis att använda vid sensorfusion. Kalmanfiltret är rent teoretiskt bättre men lämpar sig inte alltid till inbyggda system som kan vara processorsvaga. I koden till FlightCtrl kortet så har de inte

använt sig av något av de kända sätten utan de har gjort en egendesignad lösning som fungerar bra.

2.4 PID-regulatorn

PID-regulatorn i dess olika varianter (P,PI,PD etc) är den absolut mest använda regulatorn i olika sammanhang. Den är enkel att förstå och det finns tumregler för hur man ska trimma in den.

Namnet PID kommer från regulatorns tre beståndsdelar där P står för proportionell, I för integrerande och D för deriverande del. På matematisk form skrivs den enligt nedan:

$$( ) \quad ( ) \quad \int ( ) \quad - ( ) \tag{2.18}$$

Här ser man att styrsignalen ( ) beräknas genom en viktad summa av det aktuella reglerfelet (börvärde − ärvärde, ( )), det ackumulerade reglerfelet och det predikterade framtida reglerfelet. är designparametrar som väljs för att uppnå önskvärd prestanda och robusthet för systemet. (Hägglund 2008)

Rotationsdynamiken för en multirotor kan enkelt beskrivas som ett andra ordningens system.

$$\frac{( )}{( )} \quad \frac{}{} \tag{2.19}$$

där är aerodynamisk dämpning som oftast är ganska liten. I och med att det är ett andra ordningens system ska man kunna reglera det med PD-regulator, man har då 2 designparametrar. (Corke 2011)

Valet av PD-regulator baserades dels på hur den befintliga mjukvarans uppbyggnad såg ut samt rådgivning med handledare på SAAB. Detta i kombination med att det i litteraturstudien framkommit att en PID-regulator står sig bra mot andra typer som en LQ-regulator för denna typ av farkoster (Bouabdallah, Noth och Siegwart 2004). Eftersom piloten kommer att vara med i loopen och ta bort stationära fel så visade det sig att en PD-regulator var tillräcklig för att kunna flyga farkosten i de fall som testades.

3. Oktakoptern

MK Okto XL som visas i figur 3.1 är den kraftfullaste multirotorn som Mikrokopter erbjuder. Detta är den farkost som har modellerats och reglerats. I detta kapitel beskrivs plattformen och farkosten som användes under examensarbetet.

Figur 3.1 MK Okto XL

MK Okto XL Specifikationer

- Mått: 73x73x36cm (BxLxH)
- Max last: 2500g
- Max höjd: >200 meter
- Max distans: >200 meter
- Flygtid: 18-28 minuter

3.1 Val av plattform

Efter att ha undersökt olika alternativ på marknaden beställdes en MK Okto XL från Mikrokopter.de. Detta var den kommersiella multirotorfarkost som hade högst lyftkapacitet. Lyftkapaciteten gjorde den mest intressant då det finns planer på att utöka farkosten med fler funktioner såsom att bära laster som kameror etc. De omdömen som fanns var att det även var en väldigt stabil plattform. Enligt hemsidan så skulle även det mesta vara öppen källkod vilket skulle göra det enkelt att modifiera och lägga till egna styr- och navigeringsfunktioner i mjukvaran efter hand.

3.2 Plattformen Oktokopter XL

Företaget Mikrokopter startades av två tyskar år 2006. Efter ungefär sex månader hade de lyckats med att skapa en flygande quadrokopter. Utveckling av deras mjuk- och hårdvara har fortskridit och nu finns det även hexakopter och oktakopter i deras utbud. Företaget startade som ett hobbyprojekt och har sedan blivit ett företag där funktioner och även fler farkoster har lagts till efterhand.

Mikrokopters farkoster är kända för att vara väldigt stabila. Det finns flera andra företag som bygger oktakoptrar och använder sig av Mikrokopters styr- och navigationskort men en annan ram. Fotografer byter ofta ram till en som är gjord av kolfiber. Den blir då lite mer modulär och man kan sätta ihop och ta isär oktakoptern snabbt för att enklare kunna frakta den. Kamerafästen kan köpas till och monteras på farkosten. Det finns allt från enklare varianter där kameran bara hänger till mycket mer avancerade med inbyggd stabilisering. SAAB valde att köpa in en mindre variant som gjorde det möjligt att fästa en systemkamera och filma under flygning. Kamerafästet har inbyggda funktioner för stabilisering i tip- och roll-led. Stabiliseringen gör att kamerans attityd behålls även om farkostens attityd ändras.

Oktakoptern består av en ram med 8 stycken aluminiumrör som fästs samman med hjälp av 2 kolfiberskivor. Längst ut på aluminiumrören sitter 8 stycken borstlösa motorer. Motorerna drivs med hjälp av ett Li-jon batteri och en ESC som omvandlar batteriets likspänning till en trefasspänning och gör det möjligt att variera varvtalet på motorerna. På motorerna sitter rotorblad monterade, under testflygning har en plastvariant använts och vid tyngre last används en kolfibervariant som ger högre prestanda.

3.3 Kretskort

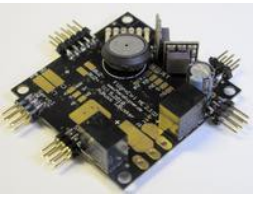
BL Control

BL Control är det tidigare nämnda ESC kort som används för att reglera varvtalen på motorerna. Kortet är specialdesignat av Mikrokopter för just de motorer som används i deras farkoster. Källkoden till detta kort är inte öppen.

Flight Control V2.1 ME (FlightCtrl)

Flight Control är huvudkretskortet. Kortet innehåller i princip alla sensorer och all kod som är nödvändig för att flyga en multirotorfarkost. Det är även på detta kort som regulatorn implementeras.

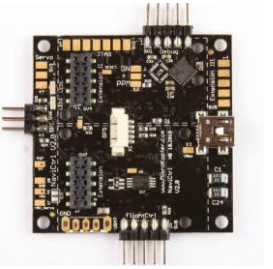
Källkoden till detta kort är öppen och processorn som gör alla beräkningar är en ATMEGA 1284P. De sensorer som sitter på kortet är accelerometer, gyro, kompass och en trycksensor för att mäta höjd.



Figur 3.2 FlightCtrl

Navigation Control (NaviCtrl V2.0)

NaviCtrl är ett tillvalskort som har en gps och därmed ger möjlighet till extrafunktioner för oktakoptern. Brytpunktnavigering (waypoint navigation) är ett exempel på funktion som är möjlig med detta kretskort. På grund av plagiering har Mikrokopter tyvärr valt att inte göra koden öppen längre.



Figur 3.3 NaviCtrl

3.4 Koptertool

KopterTool är en programvara som används för att läsa av och logga mätvärden i realtid. Här finns även möjligheter att ändra styrparametrar och kalibrera olika funktioner som används. KopterTool används även för att installera ny programvara till de olika kretskorten. Farkosten kommunicerar med en dator via seriellanslutning eller, då man flyger, via Bluetooth.

3.5 Sändare, Graupner MC-32

Sändaren som används är en Graupner MC-32 HoTT, en påkostad sändare med 16 kanaler som kan tilldelas olika kommandon. Förutom själva styrningen av farkosten så kan kanalerna användas till att styra eventuellt kameraställ och exempelvis använda en av knapparna till att ta bilder med kameran. Enligt hemsidan till oktakoptern så är denna sändare den optimala för att styra farkosten och den var även väldigt enkel att para ihop med farkosten och få alla funktioner att fungera.

Figur 3.4 Sändare

3.6 Simulator AeroSimRC

Ett simulatorprogram köptes in för möjligheten att öva flygning på dator innan första riktiga flygturen. Programmet hade en modell av just den inköpta oktakoptern. Denna kunde styras genom att koppla den inköpta sändaren till datorn med en medföljande kabel. Detta är smidigt och väldigt lärorikt och det rekommenderas starkt till alla som ska flyga en multirotorfarkost att, för att undvika krascher, börja med att flyga i en simulator.

3.7 Befintlig mjukvara

Den befintliga mjukvaran består av ca 7000 rader C kod med tyska variabelnamn. (Sa, Queensland University of Technology 2011) Tyvärr är koden väldigt dåligt kommenterad vilket ibland gör den svår att förstå. Speciellt svårt att förstå blir det då koden inte uppenbart följer teorin rakt av. Allt eftersom plattformen blivit populärare så har man börjat blanda in engelska också vilket egentligen bara gör det svårare att leta i koden.

Farkosten styrs av ett antal PID-regulatorvarianter. Tipp och roll regleras av varsin PID-regulator medan gir regleras av en PI-regulator. Tillståndsestimering utförs av en egendesignad algoritm som är varken kalmanfilter eller komplementärt filter. En forskargrupp i Australien har gått igenom koden och kommit fram till att tillståndsestimeringen inte liknar något de sett tidigare men det fungerar och farkosten flyger bra. (Sa och Corke, Estimation and Control for an Open-Source Quadcopter 2011)

En styrka med plattformen är att de flesta parametrarna enkelt går att justera, så som exempelvis PID parametrarna, via programmet KopterTool. Nackdelen är

att deras PID parametrar har visat sig inte vara klassiska PID-parametrar utan snarare en egendesignad variant där de bytt namn på de olika delarna.

Om man fäster en kamera eller annan last på farkosten så ändras dess flygegenskaper, men eftersom man enkelt kan ändra många parametrar gör det att man kan testa sig fram till vilka inställningar som passar ens egna flygpreferenser bäst. Alla ändringar görs genom det medföljande programmet KopterTool som har ett enkelt och användbart GUI.

Navigationskortet tillför en mängd funktioner som underlättar flygningen. Några av dessa är:

- *altitude hold*, vilken innebär att farkosten håller angiven höjd automatiskt.
- *position hold* som gör att farkosten håller sig kvar på en och samma position.
- *coming home* vilken piloten kan aktivera när han vill att farkosten automatiskt ska flyga tillbaka till sin startplats. Denna funktion aktiveras även om man skulle tappa kontakten mellan farkost och sändare.
- *Brytpunktnavigering* inom en viss radie.

En mängd parametrar kan ställas in så som på vilken höjd farkosten ska flyga till startplatsen, max- och min-varvtal och egna PID-värden, alla för autonom flygning. Problemet med navigationskortet är att all kod är stängd och det inte går att använda informationen från dess GPS till att utveckla egna funktioner.

4. Modellering

En bra modell av farkosten är baserad på bra kunskap om den teori som är aktuell och i så noga specifikationer av farkosten som möjligt.

När modellen skulle påbörjas erhölls ett par delar av SAAB, airframe och dess initieringskod. En stor del av tiden ägnades åt att förstå alla de delar som erhölls för att på bästa sätt kunna utveckla bra kompatibla block.

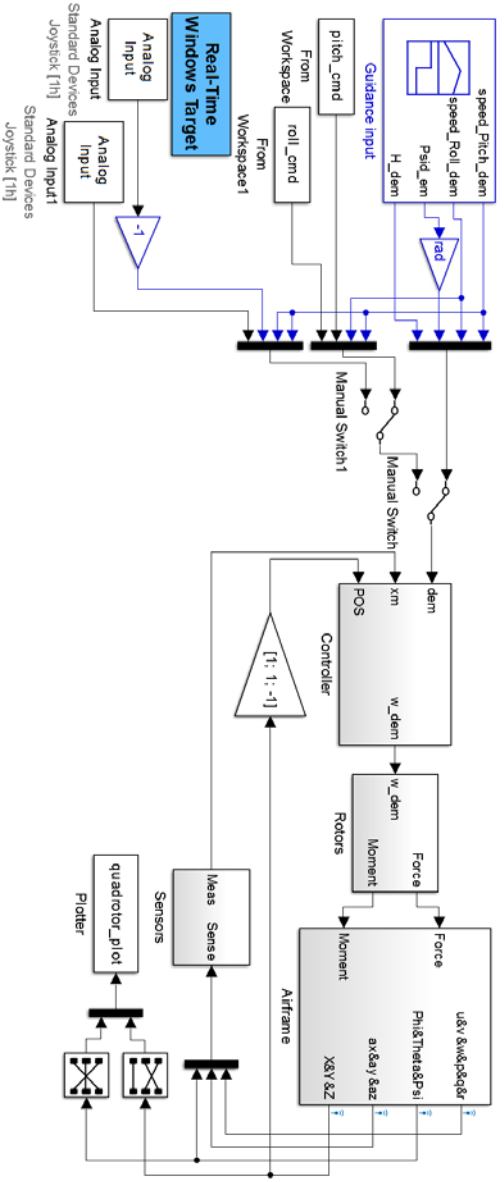
Till en början gjordes en modell i Simulink baserad på allmän teori om multirotorfarkoster. Detta kom senare att anpassas för att uppnå en bättre beskrivning av den verkliga farkosten. Fokus lades enbart på stelkroppsdynamiken och inte aerodynamiken då SAAB under många år redan tagit fram en väldigt bra modell för den.

I Simulink utvecklades en vinkelregulator samt en hastighetsregulator. Att styra en multirotorfarkost med hjälp av att reglera dess Euler-vinklar är det vanligaste sättet och det krävs inte heller någon positionsdata för att fungera. Insignaler på önskad vinkel, som motsvarar användarens kommandon från sändaren, jämförs med den aktuella vinkeln. Vinkelregulatorn kaskadkopplas sedan med en hastighetsregulator. Hastighetsregulatorn var ett önskemål från SAAB och är i sin tur beroende av just hastighetsdata. I hastighetsregulatorn representerar insignalerna istället önskad hastighet i x- och y-led och därefter beräknas en vinkel för att farkosten ska hålla angiven hastighet.

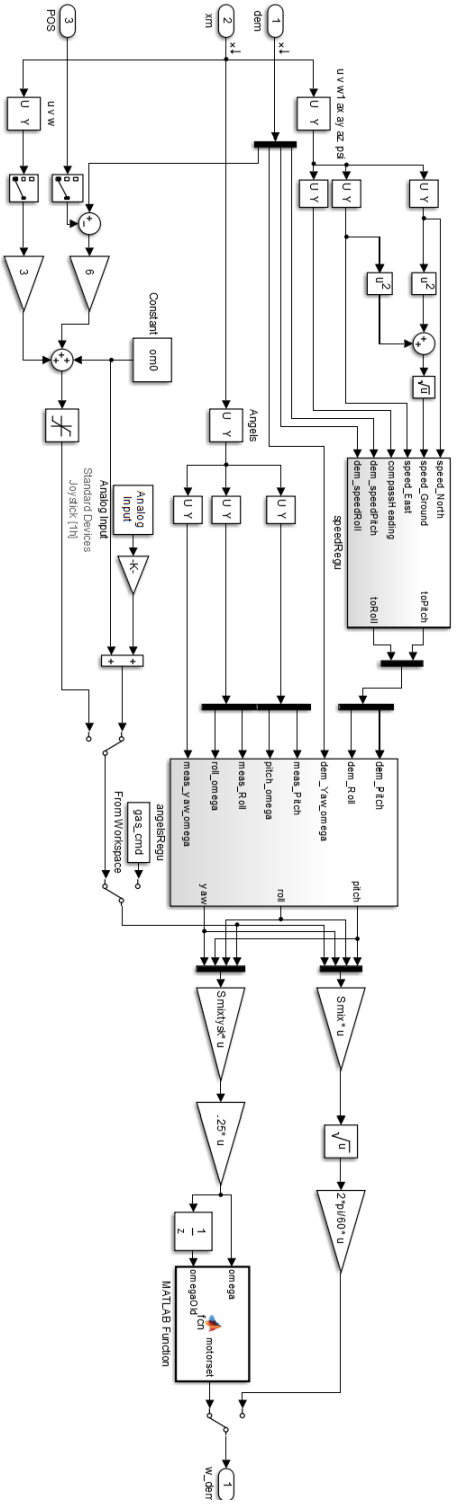
Regulatorblocken anpassades för autogenerering av C-kod. Det gjordes genom att enbart det mest väsentliga ligger i blocken samt att in- och utsignaler är anpassade för att enkelt fungera med den befintliga koden.

4.1 Simulinkmodell

I figur 4.1 och 4.2 visas en översikt av simulinkmodellen där de enskilda blocken beskrivs i senare avsnitt. Via manuella switchar så kan man växla mellan olika typer av insignaler (signalgenerator, loggdata, joystick) och även mellan teoretiska utsignaler eller signaler anpassade för den aktuella mjukvaran.



Figur 4.1 Överblick av simulinkmodell

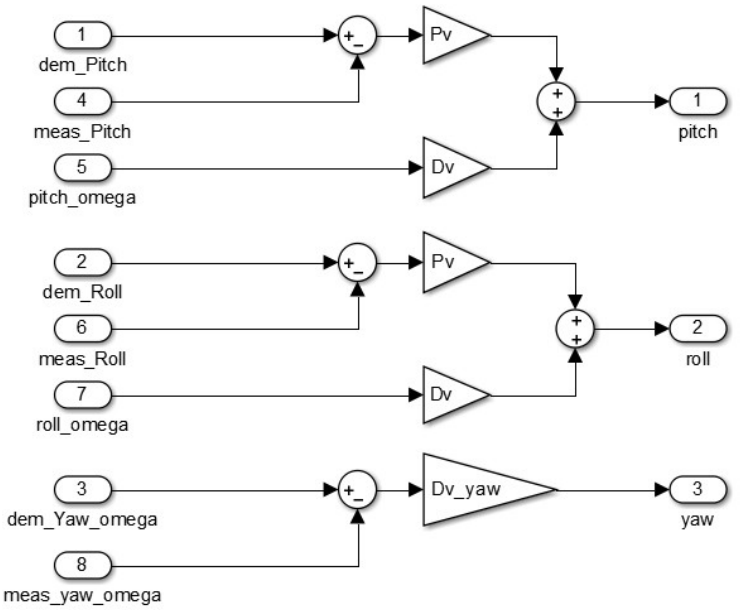


Figur 4.2 Overblick av regulator delen "Controller"

4.2 Insignaler

För att representera användarens styrkommandon används ett par olika alternativ. Vid test av enklare signaler används signalgeneratorn och för jämförelse med tidigare gjorda testflygningar används Matlabs *from workspace* där styrsignalerna är importerade ifrån testflygningarnas loggar. Det finns även en möjlighet att använda en joystick som insignal och då kan modellen flygas i Matlab med till exempel en Xbox-kontroll. För modell med enbart vinkelregulator motsvarar signalerna önskad vinkel för farkosten medan styrsignalerna för modell då även hastighetsregulatorn är inkopplad motsvarar önskad hastighet.

4.3 Vinkelregulator



Figur 4.3 Vinkelregulatorblockets innehåll

Vinkelregulatorn reglerar farkosten till önskad vinkel, det vill säga att en förändring på spaken är proportionell mot den vinkel man önskar för farkosten.

För att uppnå den önskade vinkeln beräknas först ett önskat vridmoment i x-, y- och z-led ut. Via ytterligare matrisberäkningar erhålls ett önskat varvtal på varje enskild motor. En förändring av vinkeln leder till en hastighet i xy-planet.

För gir är det lämpligare att låta spaken på kontrollen styra en vinkelhastighet i gir-led. Ifall den skulle styra en vinkel så skulle farkosten återvända till sin ursprungsvinkel så fort man släppte spaken till sitt nolläge. Detta är normalt inte det man vill ska ske när man flyger, utan då vill man att farkosten ska behålla sin gir-riktning när man släpper på spaken.

Vid användning av endast vinkelregulator kommer farkosten fortfarande ha en avklingande hastighet i sida när den önskade vinkeln återställs till neutral position och för att få farkosten att snabbt stanna måste användaren själv kompensera för detta.

Regulatorn använder sig av aktuell vinkel och jämför med den önskade och vinkeln regleras sedan med hjälp av en PD-regulator. Insignaler till regulatorn är önskade vinklar, aktuella vinklar och aktuella vinkelhastigheter.

En allmän beräkning av vridmomentet för tipp-vinkeln med en PD-regulator ges av

( ) ( ) (4.1)

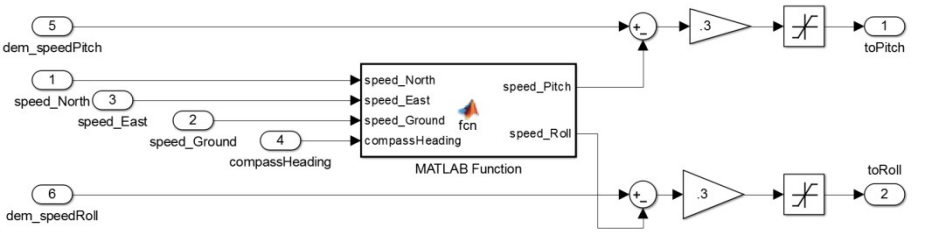
där och är designparametrar, är den önskade vinkeln, den aktuella vinkeln, önskad vinkelhastighet och aktuell vinkelhastighet. Eftersom målet är att hålla en önskad vinkel är den önskade vinkelhastigheten 0 och därför är

Motsvarande görs för roll-vinkeln och för gir används en D-regulator för att erhålla en hastighet i gir-led, dvs

( ) (4.2)

med som designparameter, önskad vinkelhastighet och aktuell vinkelhastighet. Figur 4.3 visar vinkelregulatorblocket från Simulink.

4.4 Hastighetsregulator



Figur 4.4 Hastighetsregulatorblockets innehåll

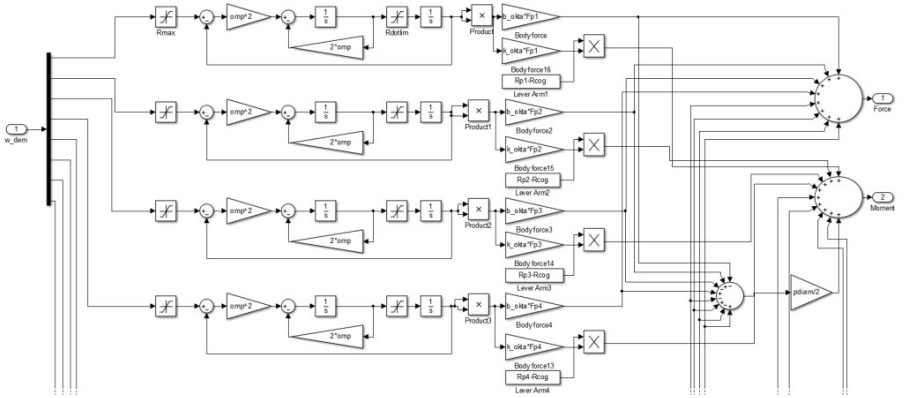
Målet med hastighetsregulatorn är att en viss position på spaken ska motsvara en viss hastighet i xy-led. Insignaler är önskad hastighet i x- och y-led (sett till farkosten) och med hjälp av en P-regulator beräknas en önskad tipp- och rollvinkel. Den stora fördelen med en hastighetsregulator är att den automatiskt stannar när man släpper spaken till nolläget. För att denna ska fungera krävs det att man har tillgång till GPS-data med aktuella hastigheter.

Den önskade tipp-vinkeln beräknas enligt:

( ) (4.3)

där är designparameter, den önskade hastigheten i x-led och är den aktuella hastigheten i x-led. Motsvarande beräkning görs för rollvinkel, då med hastigheter i y-led. Figur 4.4 visar hastighetsregulatorblocket från simulinkmodellen.

4.5 Rotorblock



Figur 4.5 Rotorblockets insida

Modellen följde till en början teorin och använde sig av varvtal. Då varvtalen inte kunde identifieras så har modellen anpassats till att hantera en spänning istället. Grunden är densamma med skillnaden att det som modellen tror är varvtal istället är ett värde på spänningen för ett visst varvtal. Rotorblocket är en modell av motorerna med hänsyn till deras maxhastighet och tidskonstant, därefter beräknas kraft och moment. I airframe utförs de teoretiska beräkningarna för luftdynamiken och aktuella Eulervinklar samt hastigheter avläses. Den för regulatorn intressanta mätdata återkopplas till modellen och delar av den skickas även till ett plottblock.

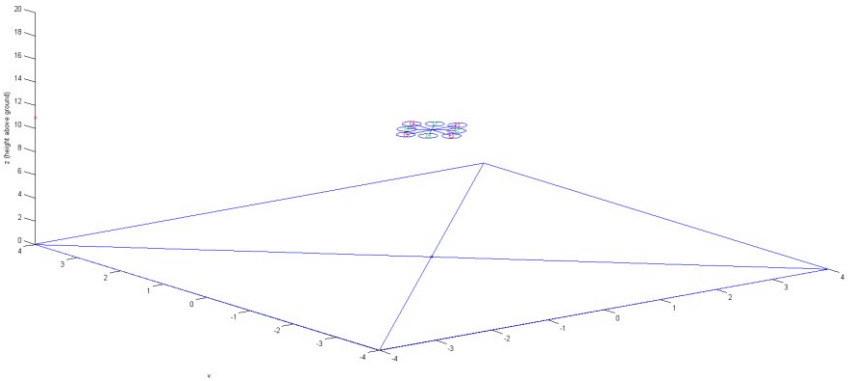
Motorerna antogs vara ett andra ordningens system

$$(\ ) \quad \text{————————} \tag{4.4}$$

där parametern trimmades in mot verklig flygdata. Figur 4.5 visar rotorblockets implementation i Simulink.

4.6 Plottblock

Peter Corke har till sin bok (Corke 2011) en medföljande simulinkmodell där han även använder en av honom framtagen 3D-realtidsplott. Detta ansågs mycket hjälpsamt vid simuleringar både för egen del och framförallt för mindre insatta personer. Koden modifierades därför för att passa vår modell och farkost. Figur 4.6 visar hur 3D-realtidsplotten ser ut när man simulerar.



Figur 4.6 3D-realtidsplott ifrån simulering i Simulink

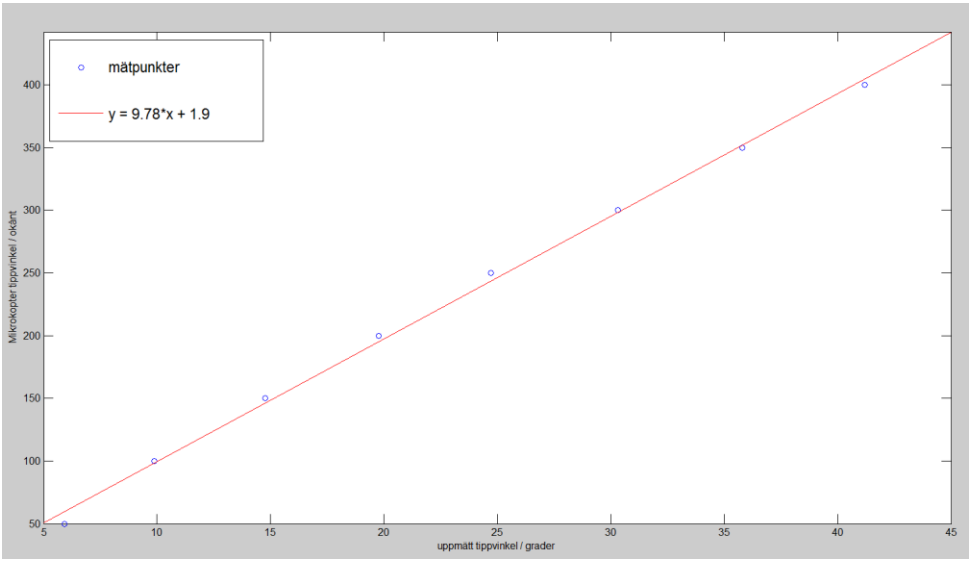
5. Resultat och analys

Här följer analys av resultaten från modellering och flygningar. De regulatorer som tagits fram och hur bra de fungerar diskuteras samt hur utförandet av parameteridentifiering utfördes.

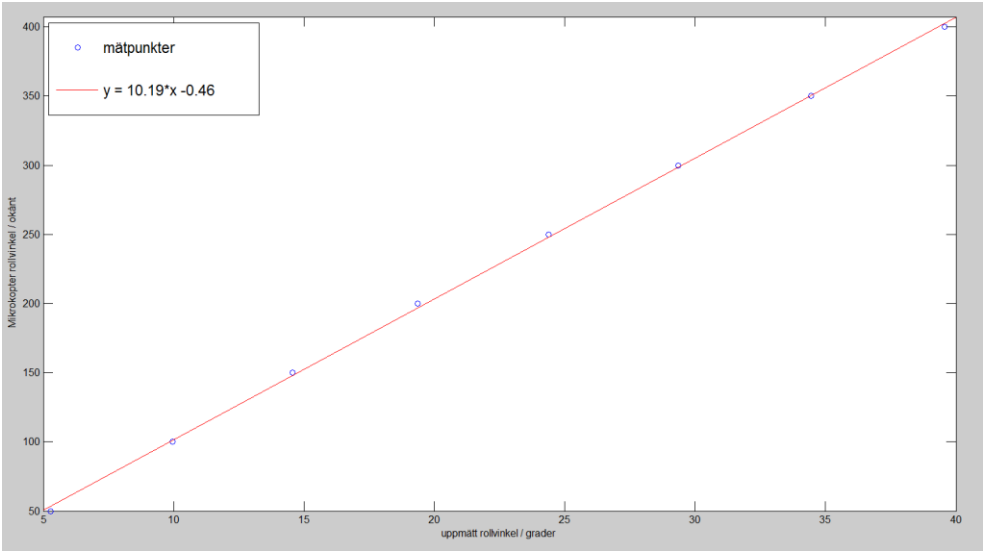
5.1 Parameteridentifiering

För både accelerometrar och gyron finns det flera möjliga felkällor. Det kan vara att de inte är placerade exakt ortogonalt mot varandra eller ett skalfaktorfel. De flesta signaler filtreras av Mikrokopter själva, men för att säkerhetsställa de signaler som används identifierades dessa och eventuella felkällor.

Att identifiera vinkelparametern gjordes genom att använda en klinometer. Den fästes på fram- respektive sidoarm av farkosten och sedan lästes de erhållna värdena av i steg om 5 grader. Utifrån testet kunde, upp till en lutning på 65 grader, en faktor tio avläsas. Figur 5.1 och 5.2 visar plottar över vinkelskattningen. För att inte riskera att vinkla farkosten för mycket valdes en begränsning av börvärdet till 1 radian (ca 57 grader) och därmed gjordes inga ytterligare mätningar. Beräkning av skalningsfaktorn på vinkelhastigheten gjordes med loggvärden ifrån flygningar och egna testfall där farkostens attityd ändrades med konstant vinkelhastighet. Graferna avgränsades till de områden där konstant vinkelhastighet kunde avskiljas och ett medelvärde av vinkelhastigheten beräknades. Vid konstant vinkelhastighet så deriverades vinkelplotten för samma tidpunkt och på grund av att vinklarna där är kända så kunde vinkelhastigheten beräknas. Detta gav oss en skalfaktor på ca 18,3.



Figur 5.1 Resultat av klinometer mätningar för tipp



Figur 5.2 Resultat av klinometer mätningar för roll

En ideal rotors dragkraft och vridmoment har ett kvadratiskt samband:

(5.1)

(5.2)

(5.3)

(5.4)

där och båda beror på luftens densitet, propellrarnas dynamik och konstanterna och . Dessa konstanter är enhetslösa och kan bestämmas med propellerns vingprofildata. Deras relation är

√ (5.5)

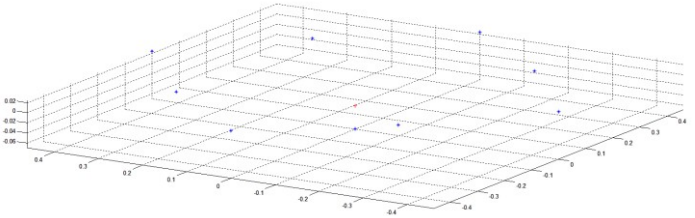
Då det inte fanns tillgång till vingprofilsdata så gjordes en skattning av som baserades på värden då farkosten hovrade. Enligt formeln för lyftkraften (2.13) kan vid hovring allt utom identifieras och därmed kan beräknas. Det går nu att härleda vilket gör det möjligt att med (5.4) och (5.5) beräkna och .

(5.6)

En första skattning av tröghetsmomentet utfördes med ett Matlabscript. I figur 5.3 visas resultatet för beräknat masscentrum för farkosten, den röda punkten. För att göra beräkningen approximerades viktfördelningen av farkosten med punktmassor, de blå punkterna. Dessa användes sedan till att räkna ut masscentrum och därefter tröghetsmomentet. Axlarna visar avstånd i meter.

I 5.7 ser vi den tröghetsmatris som erhölls:

( ) ( ) (5.7)



Figur 5.3 Graf ifrån tröghetsmomentsberäkning

5.2 Jämförelse modell/verklig farkost

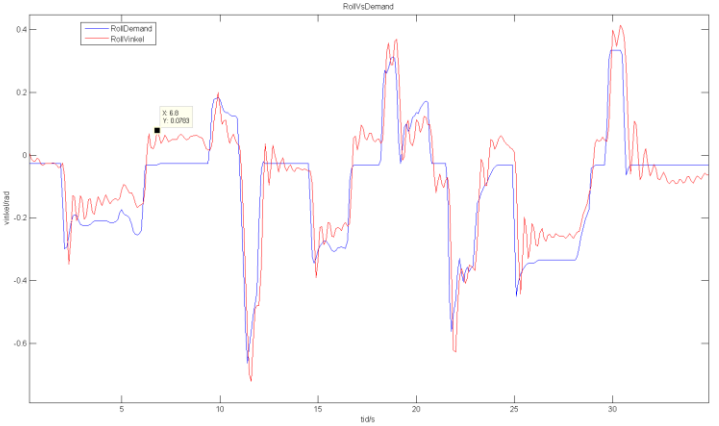
För att jämföra hur modellen avspeglar den riktiga farkosten användes loggar från testflygningar. De första försöken gav väldigt dåliga resultat både beroende på att modellen inte var så bra som förväntat och att de skalningar som fanns i C-koden inte hade identifierats tillräckligt bra. När farkosten flög första gången med egenframtagen regulator gav det mycket information, den flög inte speciellt bra och var knappt kontrollerbar men det gav mycket nyttig data.

De loggade styrsignalerna från flygningen matades in som styrsignal i modellen med samma regulator som användes i testflygningen. Av detta kunde man då se att den modell som tagits fram inte överensstämde helt med verkligheten.

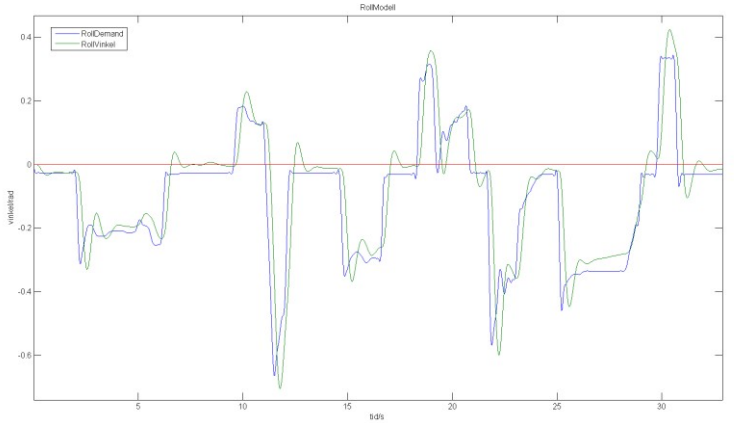
I figurerna i resten av detta kapitel visas börvärde mot ärvärde. För att jämföra hur modellen skiljer sig mot verkligheten jämförs graferna från modellen med grafer utifrån de loggade värdena.

Den data som loggas ger ut 10 värden per sekund. Eftersom att modellen är tidskontinuerlig och värdena mäts betydligt snabbare än 10Hz i verkligheten interpoleras dessa värden innan de används som insignaler i modellen. Efter att modellparametrar justerats erhölls en modell som stämde väl överens med verkligheten. Störst förändringar var justering av tröghetsmomentet, tyngdpunkt och modell för rotorerna.

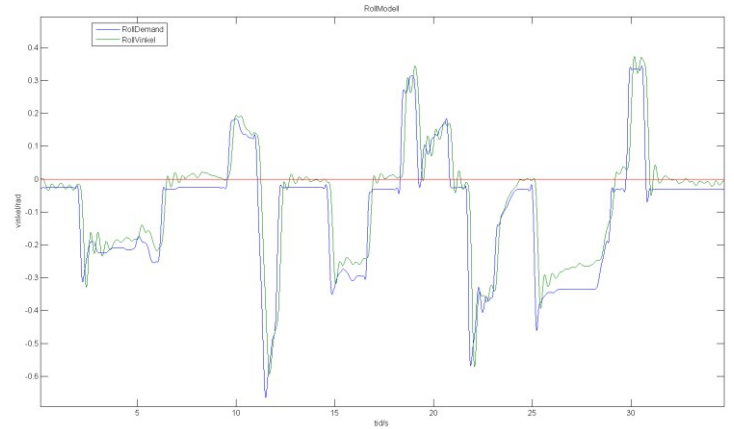
Efter att modellen finjusterats så kunde en ny regulator trimmas in och den regulatorn som trimmades in efter ett par försök fungerade riktigt bra även i verkligheten. Värdena på och är 194 och -37.



Figur 5.4 Resultat ifrån verklig data



Figur 5.5 Resultat ifrån simulering med felaktig modell



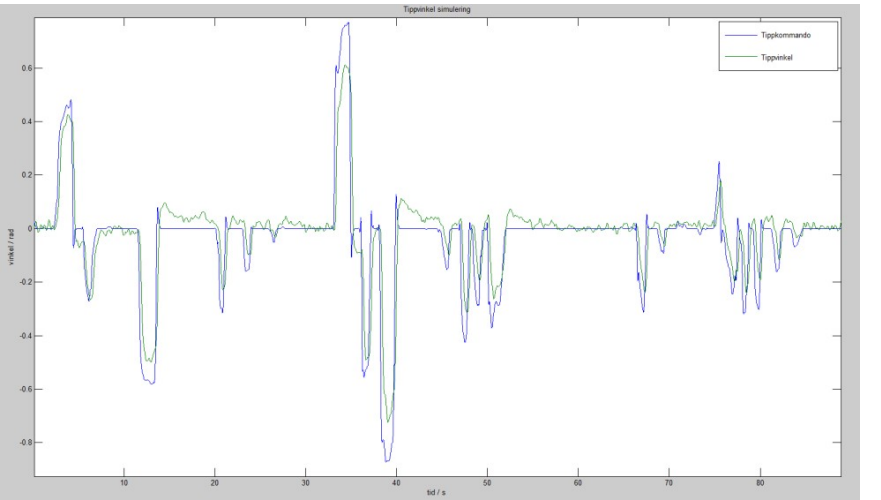
Figur 5.6 Resultat ifrån simulering med den slutgiltiga modellen

Första grafen, figur 5.4, visar att den verkliga flygningen har periodiska svängningar i stegsvaret som är mellan sekund 2 till 7. Vid samma intervall i figur 5.5, så hade första modellen inget av detta. När modellen sen ska gå tillbaka till sitt ursprungsläge så är inte överslängen och stationära felet lika stora som i verkligheten. Man kan också se att den första modellen har en längre dödtid vid första stegsvaret än i verkligheten. Tittar man däremot på den korrigerade modellen i grafen, figur 5.6 ser man den periodiska svängningen med i princip samma periodtid, det stationära felet överensstämmer lite bättre med den verkliga flygningen och även dödtiden stämmer bättre.

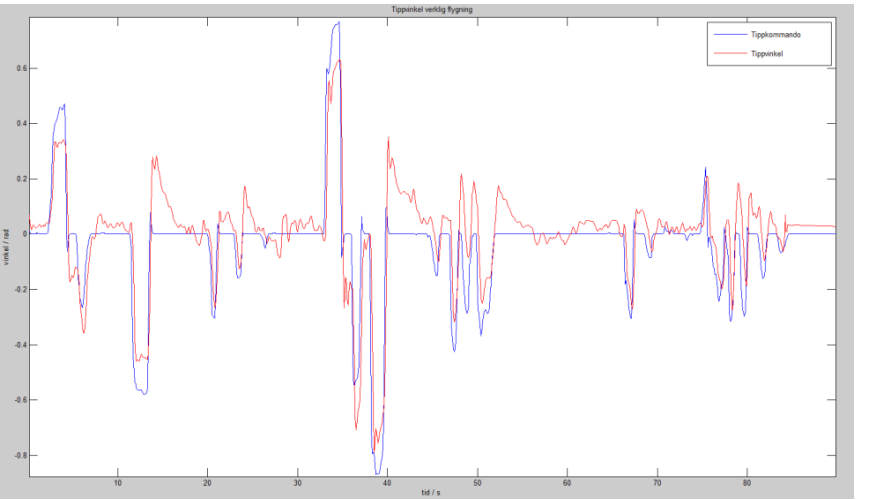
En sak som kan tyckas konstig är att stationära läget inte är 0 grader utan att den har en liten offset. Detta beror på att gyroskopen kalibreras varje gång man startar med ett nytt batteri och då lutar marken oftast lite. Det felaktiga nolläget kompenseras av användaren när man startat genom att justera bort offseten med fasta styrsignaler så att den står still och hovrar då spakarna är i neutralt läge. Det betyder att man i nolläget har en liten styrsignal hela tiden för att justera bort den lilla initiala lutningen.

5.3 Analys av regulatorprestanda

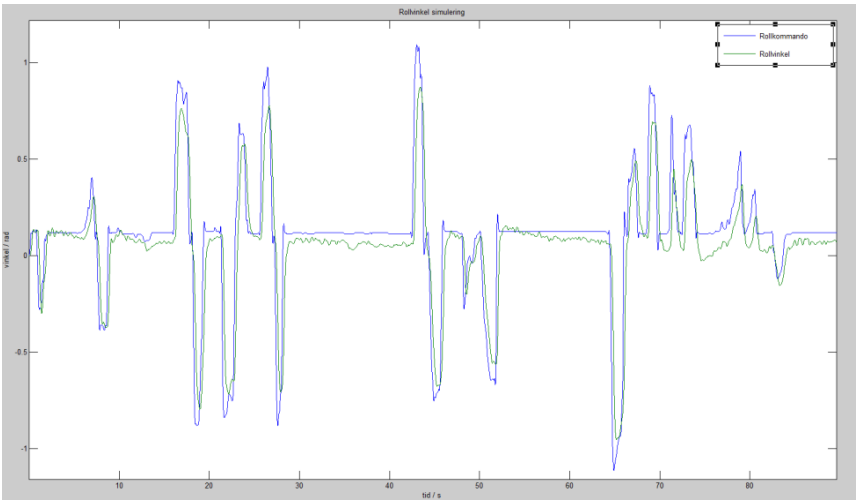
Graferna nedan figur 5.7 – 5.10 representerar en vanlig flygning, graferna visar tipp och roll vinklar för både en verklig (figur 5.8, 5.10) och simulerad flygning (figur 5.7, 5.9). Graferna visar att modellen är ganska bra och stämmer väl överens med verkligheten. Svagheten i modellen ligger i när man tippar och rollar samtidigt som t.ex. vid intervallet 45-50 sekunder. I det intervallet ser man att simuleringen skiljer sig en del från den verkliga flygningen. Simuleringen är bättre och följer börvärde bättre än vad som sker i verkligheten just när man tippar och rollar samtidigt, medan bör- och ärvärde överensstämmer väldigt bra för de delarna när man utför roll- eller tippändringar separat.



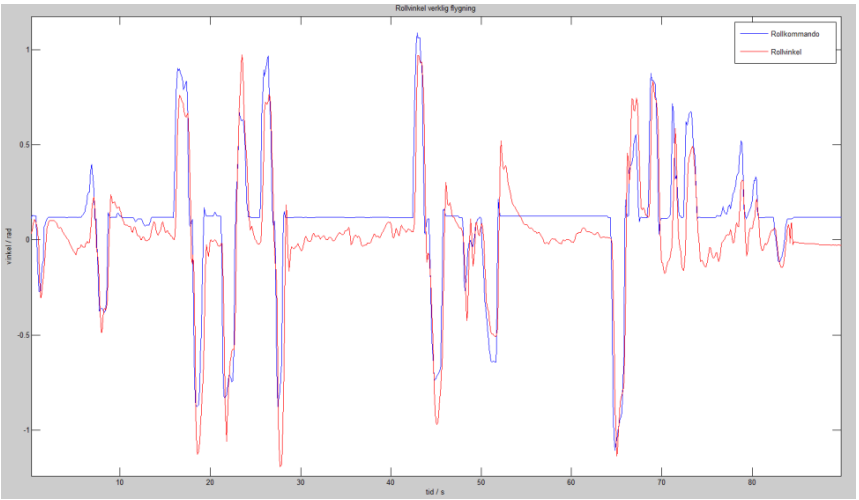
Figur 5.7 Simulering av vanlig flygning för tippvinkel



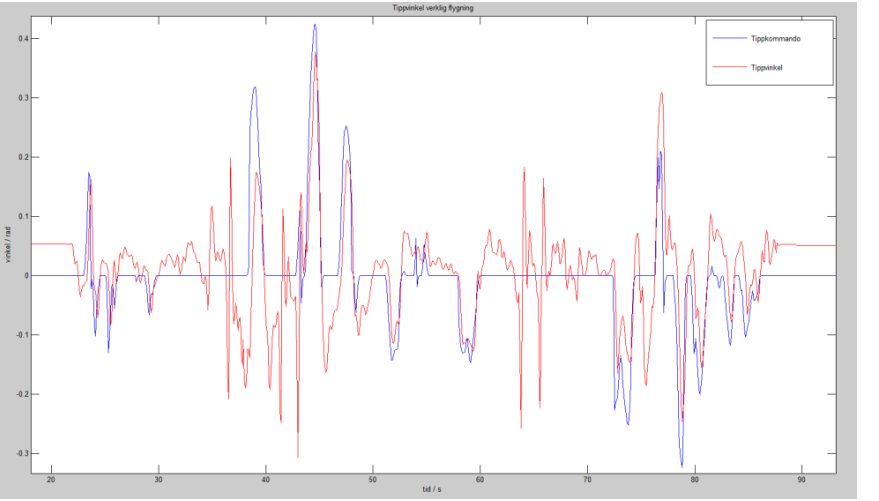
Figur 5.8 Plott av vanlig flygning från verklig data för tippvinkel



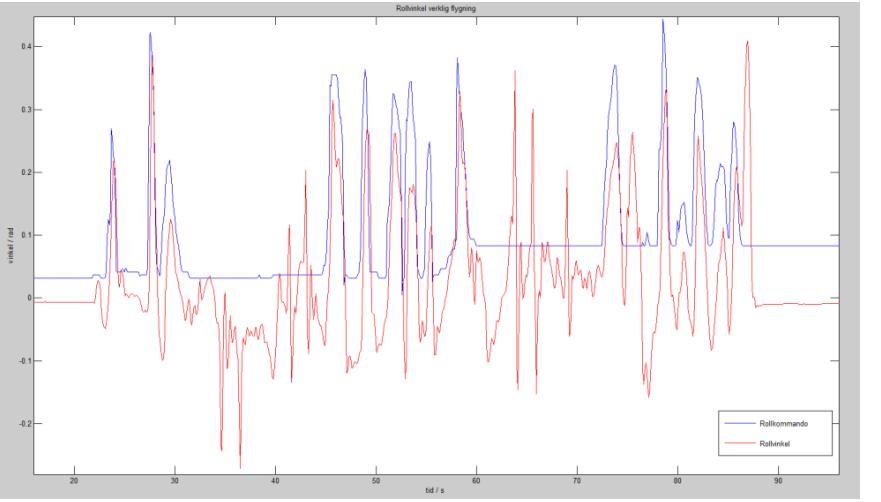
Figur 5.9 Simulering av vanlig flygning för rollvinkel



Figur 5.10 Plott av vanlig flygning från verklig data för rollvinkel



Figur 5.11 Graf av tippvinkel under en flygning med laststörningar

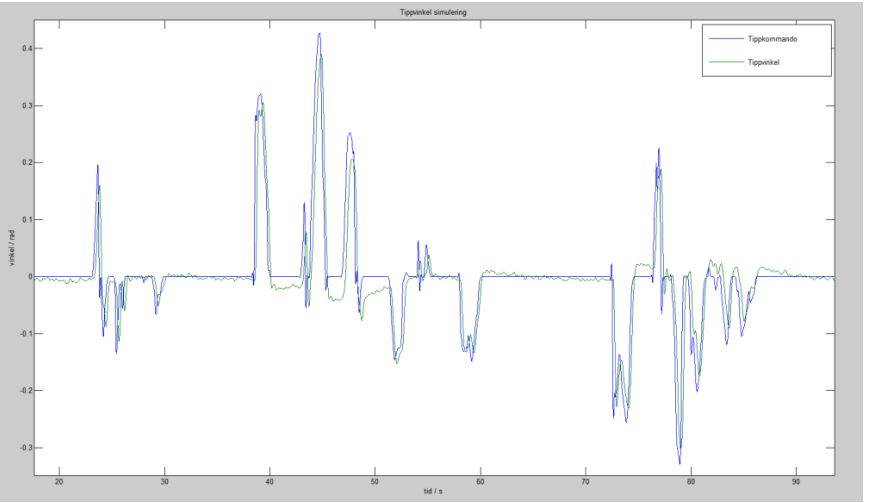


Figur 5.12 Graf av rollvinkel för en flygning med laststörningar

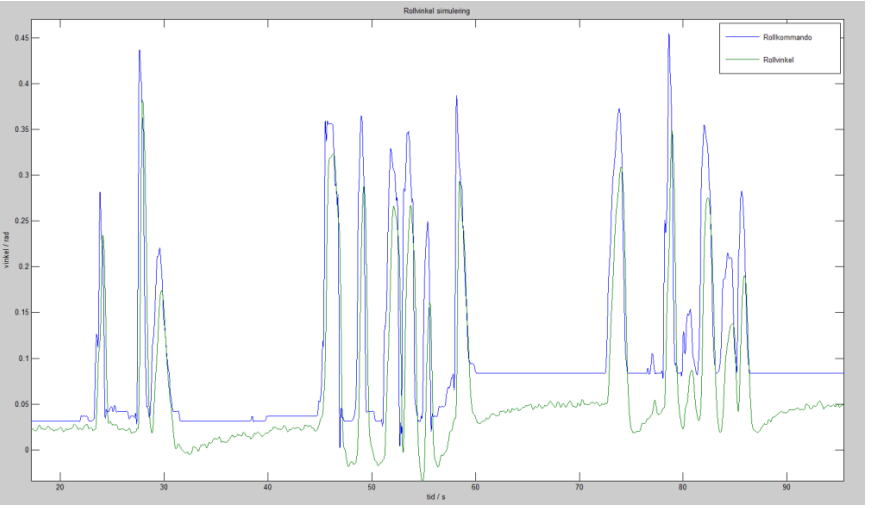
Graferna i figur 5.11 och 5.12 visar tipp- och rollvinklarna när farkosten utsätts för laststörningar. Dagen då testerna utfördes blåste det en del sidvind så man ser en offset också pga. kompensation för vinden. Laststörningarna inducerades genom att fästa ett snöre i koptern och sen dra i det flera gånger då koptern hovrade. Man ser störningarna mellan i intervallen vid ca 30-45s och 60-70s, snöret rycktes hårt

men trots det så vinklades farkosten som max ca 17 grader. Den hade inga problem att kompensera för det och kändes aldrig instabil under den perioden.

Graferna i figur 5.13 och 5.14 visar simuleringen av samma data som vid laststörningen utan att införa laststörningar i modellen. Genom att jämför graferna med figur 5.11 och 5.12 så kan man lättare se när farkosten utsattes för laststörningar.



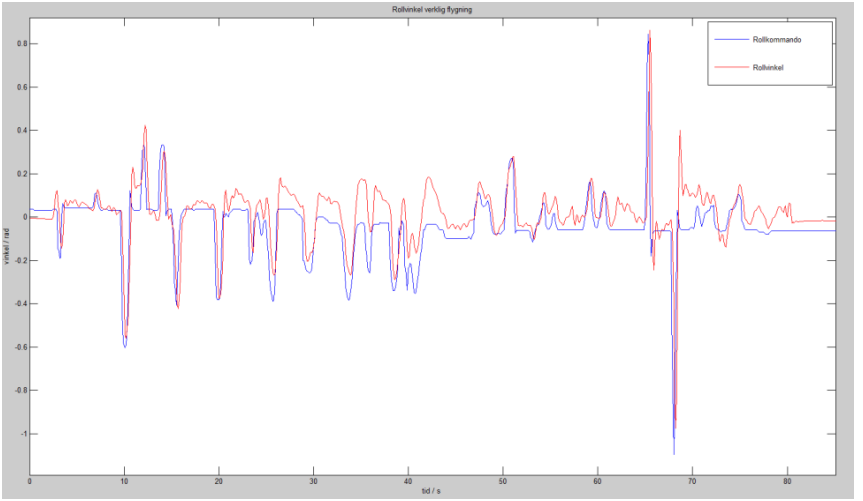
Figur 5.13 Simulering av tippvinkel från laststörningsflygning



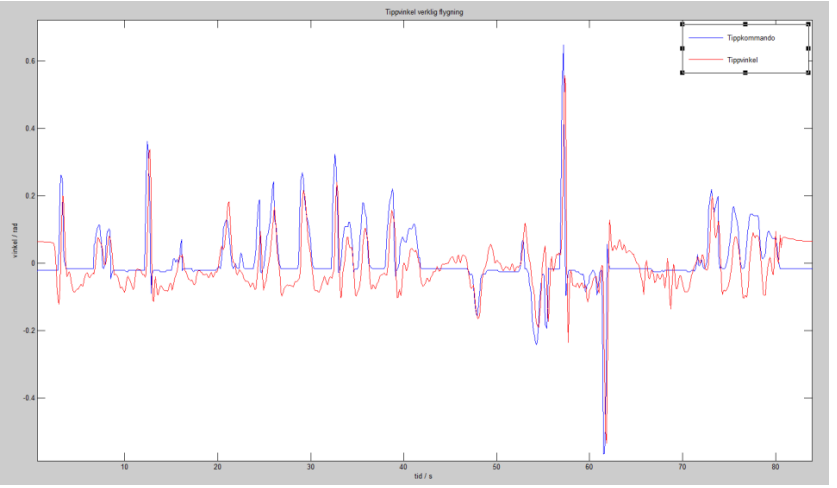
Figur 5.14 Simulering av rollvinkel från laststörningsflygning

Modellfel

I figur 5.15 och 5.16 har ett modellfel inducerat genom att fästa en vattenflaska på ca 0.5kg på farkostens ena landningsben. Detta flyttar tyngdpunkten på farkosten i och med att flaskan inte är i mitten under farkosten. Det blir en sned belastning på farkosten men regulatorn klarade det utan problem. Under flygningen så var det knappt märkbart att flaskan var påkopplad.



Figur 5.15 Graf över rollvinkeln under flygning med flaska



Figur 5.16 Graf över tippvinkeln under flygning med flaska

Av resultatet kan man se att vinkelregulatorn ger mycket goda resultat och även när extremfall testas så är resultaten bra.

Den framtagna regulatorn är snabbare och det känns även vid styrning av oktakoptern då den uppnår börvärde snabbare och samtidigt även känns aningen stelare än den ursprungliga.

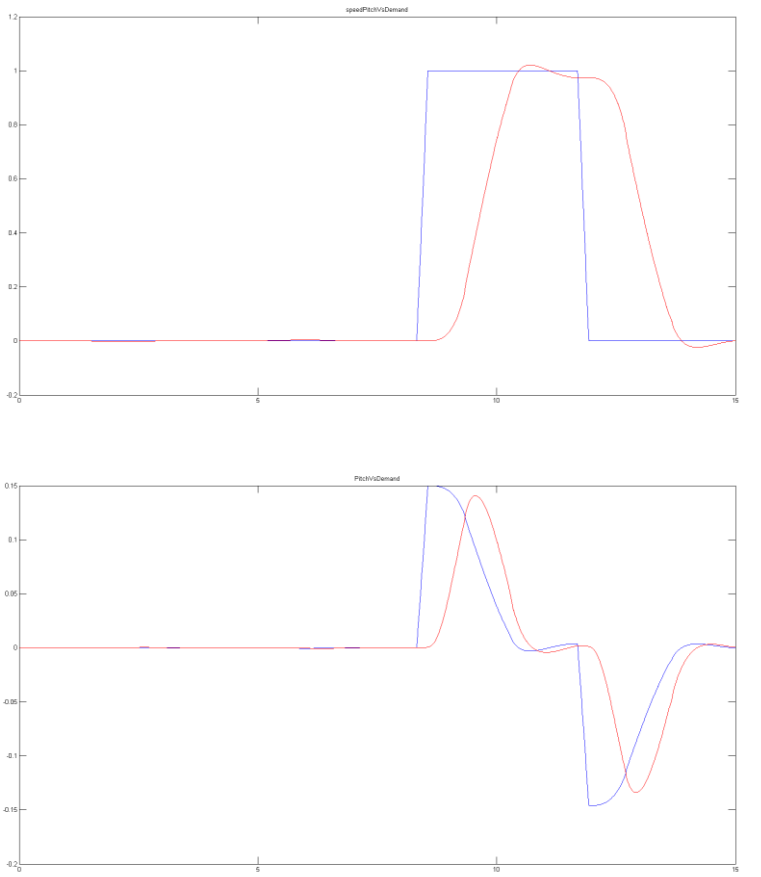
5.4 Implementering av autogenererad kod

Den autogenererade koden skapar ett antal nya c-filer. I ordinarie kod utförs hänvisning till dessa genom variabeldefinitioner. De aktuella signalerna mappas till de signaler som de autogenererade filerna använder. Efter det exekveras huvudfunktionen i den autogenererade koden och när den har körts så mappas de styrsignaler som beräknats till de aktuella styrsignalerna för farkosten.

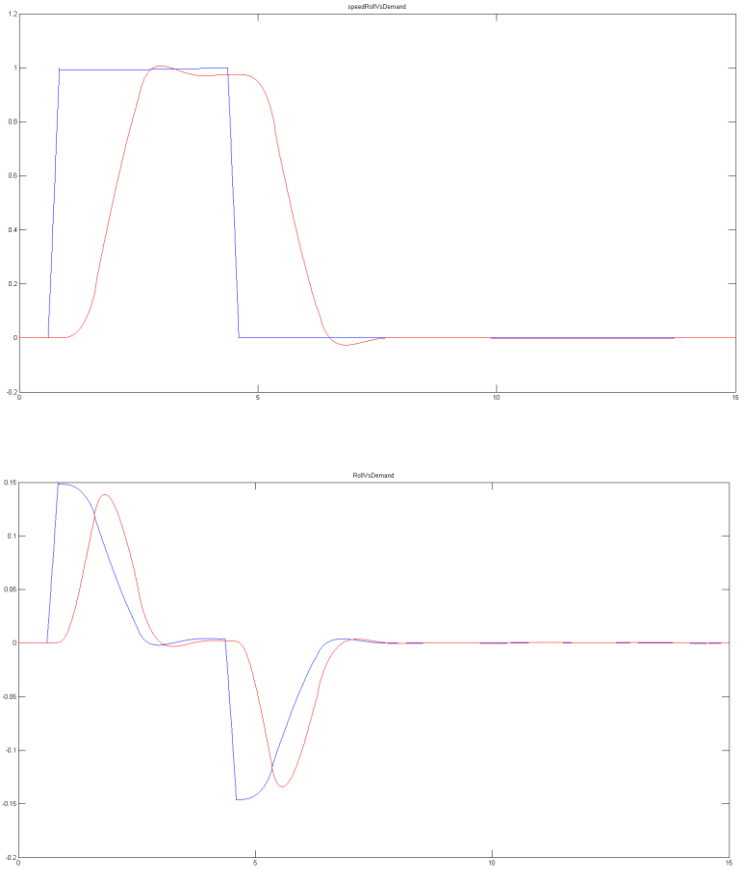
Integreringen mellan befintlig kod och egen kod resulterade inte i någon märkbar ökning av beräkningstid och den ordinarie koden för styrsignalen körs parallellt för möjligheten till jämförelse.

5.5 Hastighetsregulatorn

När det framkom att det saknades åtkomst till GPS-data togs beslutet att inte vidareutveckla hastighetsregulatorn. Den regulator som tagits fram i Simulink är förberedd för autogenerering och fungerar bra i modellen. I figur 5.17 och 5.18 ser man hur en önskad hastighet i tipp- respektive roll-led regleras med hjälp av en beräknad vinkel för tipp respektive roll. Man ser att regulatorn automatiskt korrigerar med en vinkling åt motsatt håll när hatighetskommandot är 0 för att få stopp på farkosten.



Figur 5.17 Hastighetsregulator med önskad hastighet i tipp-led samt det tipp-kommando som vinkelregulatorn erhåller.



Figur 5.18 Hastighetsregulator önskad hastighet i tipp-led samt det tipp-kommando som vinkelregulatorn erhåller.

6. Diskussion och slutsats

I detta kapitel diskuteras resultat, lärdomar och förslag på fortsatt arbete.

6.1 Matlab/Simulink

Att ta fram regulatorer som fungerade tillräckligt bra tog inte så lång tid utan den mesta tiden gick åt till att få alla block att verkligen fungera enligt teorin.

Efter att allt såg ut att stämma så kunde mer tid läggas på att optimera regulatorerna för att uppnå de mål som var satta, att uppnå liknande eller bättre reglering jämfört med de regulatorer som redan fanns i farkosten.

Koden för Mikrokopters egen regulator var väldigt rörig och det var svårt att förstå hur den fungerade/var designad. Detta medförde att vi själva inte lyckades implementera deras regulator i Matlab modellen. Hade detta varit möjligt för oss så hade det varit betydligt enklare att trimma in modellen direkt mot verklig data.

Ifrån de första testflygningarna med vår regulator erhöll vi väldigt nyttig flygdata. Nu hade vi loggar från en verklig flygning med en helt känd regulator, så då var det bara till att jämföra denna med hur modellen betedde sig.

6.2 Jämförelse av flygupplevelse med de olika regulatorerna

Den regulator som medföljer har optimerats efter många år av finjustering och fungerar riktigt bra. För att säkerställa om vår regulator fungerade överhuvudtaget vid de första försöken tränades flygkunskapen upp med den befintliga koden. Detta för att vår brist på flygrutin inte skulle påverka flygningen utan att vi enkelt skulle kunna känna skillnaden på hur farkosten agerade i luften med vår regulator jämfört med originalet.

Med den regulator som togs fram upplevs en skillnad jämfört med originalet där den framtagna regulatorn även den är enkel att flyga med och farkosten svarar bra mot de börvärden som skickas. Skillnaden ligger framförallt i aggressivitet. Den framtagna regulatorn är lite mer aggressiv, medan originalet upplevs som

långsammare och mjukare. Detta är helt enkelt bara två olika regulatorer och vilken som föredras beror helt på användarens preferenser och på det flyguppdrag som ska utföras.

6.3 Problem

Det största problemet som stöttes på under arbetets gång var att källkoden inte var så öppen som förväntat. I princip all kod är även skriven på tyskinfluerad form och även okommenterad, vilket gjorde att det gick åt en hel del tid till att förstå hur allt var sammankopplat.

Just navigeringslösningen låg planerad en bit in i projektet vilket medförde att upptäckten att delar av GPS-koden inte var tillgänglig kom sent i projektet. Detta ledde till begränsningar i tänkt implementation och då det inte fanns tid, och heller inte innefattades av projektet, att skriva all gps-kod på egen hand, som annars hade varit en möjlig lösning. Utifrån de förutsättningarna fattades beslut om att inte implementera hastighetsregulatorn i verkliga farkosten men istället att undersöka och beställa ett annat gps-kort som skulle kunna fungera för framtiden. Resultatet blev att ett NGUAVP-kort beställdes och möjlighet för gpsfunktioner nu finns för framtida projekt.

Förseningar av leveransen

Det var flera olika saker som bidrog till förseningar. Ett av problemen var att det uppkom en hel del komplikationer vid leverans av den beställda oktakoptern. Det ledde i sin tur till att projektet blev haltande redan i starten.

Till en början kunde all tid ägnas åt inläsning av teori angående multirotorer och modellering, vilket var positivt, men endast möjligt till en viss gräns. Till slut behövs den faktiska farkosten och uteblivandet av den var kännbart. Utan att ha den faktiska hårdvaran var det svårt att sätta sig in i hur allt hängde ihop.

Bakgrund C-kod, utbildning

Arbetet visade sig kräva mycket mer C-programmering än vad som från början var förväntat. Ingen av oss hade någon bakgrund i C utan den programmering som vi hade med oss ifrån utbildningen var i princip enbart JAVA.

I efterhand har vi kommit fram till att det förmodligen hade varit mer gynnsamt för projektet om inte båda exjobbarna haft främst reglerutbildning utan istället att en hade haft inriktning mot reglerteknik och en inriktning mot programmering, gärna med erfarenhet från C och inbyggda system.

Bristande information från Mikrokopter

Projektet blev stillastående ett par gånger då information för vissa viktiga metoder inte gick att få fram ifrån den bristfälliga hemsidan som ska fungera som manual. Det gjordes försök att få kontakt med grundarna utan framgång och inte heller de svenska återförsäljarna hade kunskap för att lösa dessa problem. Efter mycket undersökning på diverse forum och även en del chansningar så lyckades lösningar utvecklas.

Väderförhållanden

En annan faktor som påverkade projektets gång var väderförhållandet. Farkosten är mycket stabil i luften men påverkas betydande av olika väderförhållanden. Främst positionen påverkas mycket av den aktuella vinden.

För att få så liten påverkan som möjligt på de avslutande jämförelsetesterna gjordes dessa när väderförhållandena var som bäst. Detta gav resultat där farkosten kunde manövreras utan större kompensation för vind. Testerna med de olika regulatorerna kunde därför utföras med i princip samma förhållanden och i sin tur ge bra jämförbar data.

6.4 Möjligt framtida arbete

Efter att grundmålet för projektet har uppnåtts finns det flera möjliga påbyggnadsprojekt, som t.ex. brytpunktsnavigering och integrering av ny MEMS-IMU och GPS på separat processorkort.

Mikrokopters plattform med Flight Control och Navigation Control är det ingen större mening att försöka utveckla mer. Den näst intill obefintliga dokumentationen och att koden till Navigation Control är stängd utgör ett stort hinder.

Ett möjligt projekt skulle däremot kunna vara att skriva om den kod som finns och utveckla en helt egen kod. Just för GPS-delen köptes ett par andra kort in från ett helt open-source-projekt. Detta projekt grundades av ett antal entusiaster som tröttnade på Mikrokopters dåliga dokumentation och bristfälliga kod. Det är helt open-source och det finns dokumentation och guider till hur man ska göra för att själv implementera olika regulatorer utan att behöva ändra något i operativsystemskoden. Här hade det varit intressant att implementera flera olika regulatorer förutom en vanlig PID regulator för att jämföra dessa. Med tillgång till ett navigationskort och dess data skulle egna navigeringslösningar kunna utvecklas och även implementation av hastighetsregulatorn skulle vara möjlig.

En utvecklingsmöjlighet för Simulinkmodellen är att integrera sändaren så att man direkt kan använda input ifrån den och får en ännu bättre uppfattning om hur farkosten kommer bete sig.

6.5 Slutsats

Att modellera en farkost utifrån teorin är inte speciellt svårt. Utmaningen ligger i att göra en modell som beter sig som en verklig farkost, där är inte alltid teori och verklighet samma sak. När modellen väl är utvecklad fungerar Simulinks autogenereringsmetoder mycket bra. Integreringen mellan den autogenererade koden och originalkod görs enkelt med grundläggande programmeringskunskaper.

7. Bibliografi

Bermes, Christian. *Design and dynamic modeling of autonomous coaxial micro helicopters.* PhD Thesis ETH NO. 18847, Zürich: ETH, 2010.

Bouabdallah, Samir, Andre Noth, och Roland Siegwart. ”PID vs LQ Control Techniques Applied to an Indoor Micro Quadrotor.” *IEEE/RSJ International Conference on Intelligent Robots and Systems.* New Orleans, 2004. 2451-2456.

Colton, Shane. den 25 Juni 2007. http://web.mit.edu/scolton/www/filter.pdf (använd den 5 November 2013).

Corke, Peter. *Robotics, Vision and Control.* 73. Springer, 2011.

Hamel, Tarek, Robert Mahoney, Rogelio Lozano, och James Ostrowski. ”Dynamic modelling and configuration stabilization for an X4-flyer.” *15th IFAC World Congress.* Barcelona, 2002.

Hossain, Raju, Geoff Rideout, och Nicholas Krouglicof. ”Bond Graph Dynamic Modeling and Stabilization of a Quad-Rotor Helicopter.” *Spring Simulation Multiconference.* Orlando: 219-227, 2010.

Hua, Minh-Duc, Tarek Hamel, Pascal Morin, och Claude Samson. ”Introduction to Feedback Corntrol of Underactuated VTOL Vehicles.” *IEEE Control Systems Magazine*, den 17 Januari 2013: 61-75.

Hägglund, Tore. *Reglerteknik AK Föreläsningar.* Lund: KFS sigma, 2008.

Luukkonen, Teppo. *Modelling and control of quadcopter.* Independent research project in applied mathematics, Espoo: Aalto University, 2011.

Mahony, Robert, Vijay Kumar, och Peter Corke. ”Multirotor Aerial Vehicles.” *IEEE Robotics & Automation magazine*, September 2012 : 20-32.

Mallikarjunan, Srinath, Bill Nesbitt, Evgeny Kharisov, Enric Xargay, och Naira Hovakimyan. ”L1 Adaptive Controlle for Attitude Control of Multirotors.” *AIAA Guidance, Navigation and Control Conference.* Minneapolis, 2012.

Pounds, Paul Edward Ian. *Design, Construction and Control of a Large Quadrotor Micro Air Vehicle.* PhD Thesis, Canberra: Australian National University, 2007.

Sa, Inkyu. ”Queensland University of Technology.” den September 19 2011. https://wiki.qut.edu.au/download/attachments/145528940/MK_Doc.pdf?version=1&modificationDate=1339475905000 (använd den 5 November 2013).

Sa, Inkyu, och Peter Corke. ”Estimation and Control for an Open-Source Quadcopter.” *Australasian Conference on Robotics and Automation.* Melbourne, 2011.

Van de Maele, Pieter-Jan. u.d. http://www.pieter-jan.com/node/11 (använd den 4 November 2013).

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
February 2014

Document Number
ISRN LUTFD2/TFRT--5935--SE

Author(s)
Henrik Ohlsson
Hrvoje Corluka

Supervisor
Torbjörn Crona, Saab Dynamics
Karl-Erik Årzén, Dept. of Automatic Control, Lund University, Sweden (examiner)

Sponsoring organization

Title and subtitle
Modeling and Control of an Octacopter

Abstract

This report deals with the theory and identification out of a multi-rotor craft, particularly an OktokopterXL from Mikrokopter.de.

A model of the craft is constructed using Matlab Simulink and a PD controller is designed to control the attitude of the craft. The controller is auto generated from Matlab to C code which is then interwoven with the multi-rotor's source code. The designed controller works well and also the model corresponds well with reality.

The multirotor platform used is not very developer-friendly with hard to understand, uncommented code and where the navigation board source code is not open source. However, the price and performance still make it a very interesting platform.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
Swedish

Number of pages
1-46

Recipient's notes

Security classification