LUND UNIVERSITY

Projekt i Adaptiv Reglering VT92

Åström, Karl Johan

1992

Document Version:
Förlagets slutgiltiga version

ISSN 0280–5316
ISRN LUTFD2/TFRT--7484--SE

Projekt i Adaptiv reglering VT 92

Karl Johan Åström
Redaktör

Institutionen för Reglerteknik
Lunds Tekniska Högskola
Juni 1992

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
INTERNAL REPORT

Date of issue
June 1992

Document Number
ISRN LUTFD2/TFRT--7484--SE

Author(s)
Karl Johan Åström (Editor)

Supervisor

Sponsoring organisation

Title and subtitle

Projekt i Adaptiv Reglering VT92 (Projects in Adaptive Control)

Abstract

This report contains the project papers in the course on adaptive control given during spring 1992.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280–5316

ISBN

Language
Swedish

Number of pages
185

Recipient's notes

Security classification

Projekt i Adaptiv Reglering

VT 92

Karl Johan Åström
Redaktör

Förord

Kursen "Adaptiv reglering" innehåller en projektdel där teknologerna skall utföra ett projekt som beräknas ta en vecka. Dessa projekt är ett viktigt moment i institutionens målsättning att **utbilda elever med en stark teoretisk grund och god ingenjörsmässig förmåga**. Denna rapport innehåller redogörelser för de projekt som utfördes under vårterminen 1992. Som lärare har det varit utomordentligt tillfredsställande att se hur duktiga våra studenter är och hur effektivt de arbetar.

Teknologerna kan fritt välja projekt från ett antal förslag som vi presenterar. De kan också själva komma med förslag. Projekten kan utföras på många olika sätt. Man kan arbeta individuellt eller i grupp. I år varierade gruppstorleken från 1 till 5 personer. Varje grupp har åtminstone två handledare så att det alltid skall finnas någon tillgänglig. Projekt kan kombineras med ett projekt i kursen "Datorimplementering av reglersystem" som går i samma läsperiod. Flera av projekten är av detta slag. Flera projekt är mycket öppna, det kan röra sig om problem som vi ej vet lösningen på eller användande av system och programvara som ej är utprovad. En konsekvens av detta är att några projekt ej kan genomföras enligt den ursprungliga planen. Vi ser detta som en god illustration av vad som händer i verkliga livet. Projekten redovisas med en kort rapport, muntlig redogörelse och demonstrationer. Vi ger kritik på rapporten men kräver ej att den skall rättas efter kritiken, det får vara måtta på vad man skall åstadkomma på en vecka. I år genomfördes presentationerna i konferensform som en serie tiominuters föredrag. Både presentationerna och demonstrationerna var utmärkta.

Denna rapport är organiserad på följande sätt. Den aktuella listan över projekt presenteras i kapitel 1, därefter kommer de olika projektrapporterna. Kapitel 2 beskriver adaptiv reglering av en hårddisk. Detta projekt utfördes av Magnus Johansson, Mats Linde, Magnus Strandh och Anders Ströbeck. Problemet är att konstruera och implementera en adaptiv regulator som fungerar även om processen har variationer i förstärkningen. Regulatorn har implementerats i Modula 2 på ett VME system, processen simulerades på en analogimaskin.

Kapitel 3 beskriver ett projekt som utfördes av Anders Carlsson, Peter Fransson, Torbjörn Olsson, Anders Robertsson och Jörgen Svensson. Uppgiften var att implementera en snabb adaptiv regulator med en signalprocessor. Regulatorn implementerades i ett Macintosh system med ett signalprocessorkort från National Instruments. Signalprocessorn TMS320C30 från Texas Instruments användes. Detta projekt stötte på många oväntade svårigheter, bl.a. fel i en C-kompilator. Tack vare envishet och hårt arbete lyckades gruppen att få ett fungerande system i tid till demonstrationen. Detta projekt kommer att fortsättas i form av två examensarbeten.

Kapitel 4 beskriver ett projekt som utfördes av Per Lantorp. Uppgiften var att implementera en indirekt adaptiv regulator i LabView.

Vi har nyligen konstruerat en ny inverterad pendel som bygger på en konstruktion av Prof Furuta vid Tokyo Institute of Technology. Denna process hade attraherat två grupper som båda gjorde samprojekt med kursen datorimplementering. Regulatorerna implementerades på ett VME system som via Ethernet var kopplat till en arbetsstation där data kan beskådas i realtid med MATLAB.

Fredrik Hammar, Karl Henrik Johansson, Kristofer Ljunggren, och Johan

Nilsson implementerade ett Kalmanfilter med parameterstyrning. De undersökte också stabilisering av pendel i upprättstående läge med en linjär styrlag och en olinjär styrlag baserad på linjärisering med hjälp av återkoppling. Omfattande simuleringar med Simnon gjordes också för prov och verifikation. Det visade sig att skillnaderna i prestanda mellan den linjära och olinjära regulatorn var mycket liten, vilket framgår av redogörelsen i Kapitel 5.

Anders Kristenson och Magnus Wiklund implementerade också ett Kalmanfilter med parameterstyrning, pendel stabiliserades också med en linjär regulator. Som krona på verket utvecklade de en bra strategi för att få upp pendeln med ett kast. Strategin utvecklades med hjälp av intuition och simulering. Den implementerades sedan och demonstrerades med framgång. Detta projekt som berskrivs i Kapitel 6 belönades med "The 1992 Inverted Pendulum Award". Detta pris som sponsrats av institutionsveteranerna Bo Bernhardsson och Kjell Gustafsson tilldelas grupp som tre gånger lyckats kasta upp och stabilisera pendeln.

Det fanns också två teoretiskt inriktade projekt. Martin Strand läste en färsk översiktsartikel i Proc. IEEE och gjorde simuleringar i anslutning till detta. Hans projektrapport finns i Kapitel 7. Tomas Idofson och Lars Arvastson var intresserade av dual reglering och stokastiska adaptiva system. Det var svårt att finna ett projekt som kunde göras på en vecka. Därför gjordes en simuleringsstudie av insomningseffekten vid adaptiv reglering med försiktiga strategier. Redogörelsen i Kapitel 8 visar att effekten är olika för stabila och instabila system.

Vi hade fått ett industriellt problem från Anders Wallenborg (Tekn. Lic. LTH 1987) på TAIAB. Den bestod i att undersöka lämpligheten av adaptiv teknik för reglering av temperaturen hos tappvarmvatten i ett bostadshus. Jan Johansson och Mats Wennberg tog sig an denna uppgift. De väsentliga svårigheterna med problemet isolerades genom analy och simulering. Man visade att viss förbättring kunde uppnås med en enkel PI adaptiv regulator som justerar förstärkningen. Resultaten finns i Kapitel 9. En alternativ lösning till problemet har gjorts av Jörgen Malmborg. Hans lösning bygger på en intressant egenskap hos systemet. Jörgens rapport finns i kapitel 10.

Bland annat på grund av den publicitet som varit fanns flera teknologer som var intresserade av oskarp logik (Fuzzy Control). Stefan Gustavsson och Lars Hermansson undersökte möjligheterna att använda oskarp logik för att ställa in PI regulatorer. Detta var också ett samprojekt med kursen Datorimplementering av reglersystem. Ett skal från Togai som välvilligt ställts till vårt förfogand genom Pelton AB användes för att generera inställningsregler. C-kod som genererats av Togai systemet länkades in i ett reglersystem implementerat i Modula 2 med man-maskin kommunikation. Redogörelsen för detta finns i Kapitel 11.

I kapitel 12 beskriver Margot Tischbierek en metod att göra adaptiv motorstyrning. Metoden har analyserats och simulerats. Mattias Gyllerup har undersökt hur neuronnät kan användas för att göra en olinjär adaptiv regulator. Hans resultat presenteras i Kapitel 13.

Karl Johan Åström

Innehållsförteckning

1. Förslag till adaptiva projekt 1992

Projekten är beräknade att ta högst en vecka. Med andra ord gör det bästa Du kan på den tiden. Projekten skall redovisas med en kort rapport och en muntlig presentation på 15 minuter + 5 minuter för frågor. Den muntliga redovisningen sker i form av ett minisymposium onsdagen den 20 maj. Då skall också projektredovisningarna vara inlämnade. Jag redigerar sedan en institutionsrapport som innehåller alla redovisningar.

Projekt 1 — Läs och förstå färsk artikel om adaptiv reglering i decembernumret av IEEE Proc. och redovisa.

Projekt 2* — Adaptiv regulator för hårddisk.

Projekt 3 — Control Benchmark Example ACC90, ACC91 och ECC91.

Projekt 4† — Val av parametrar i GPC.

Projekt 5† — Nyttan av apriori-kunskap i en adaptiv regulator.

Projekt 6† — Adaptiv reglering av kopplade system. Kan man få interaktion mellan kretsarna? Kräver lite självständigt tänkande och idérikedom.

Projekt 7† — Implementering av adaptiv regulator i LabView.

Projekt 8*† — Snabb adaptiv regulator implementerad på signalprocessor.

Projekt 9*† — Implementering av man-maskinsnitt till adaptiv regulator i LabView.

Projekt 10† — Testning av automatinställare med hjälp av realtids-Simnon.

Projekt 11† — Automatinställning av process med två insignaler och två utsignaler. Kan man ställa in PI regulatorer för processen

$$\begin{pmatrix} \frac{k_{11}}{1+sT_{11}} & \frac{k_{12}}{1+sT_{12}} \\ \frac{k_{21}}{1+sT_{21}} & \frac{k_{22}}{1+sT_{22}} \end{pmatrix}$$

med relämetoden?
Kräver lite självständigt tänkande och idérikedom.

Projekt 12*† — Regelbaserad automatinställning.

Projekt 13*† — Parameterstyrning med Fuzzy teknik.

Projekt 14*† — Parameterstyrning med hjälp av återkopplingslinjärisering, med tillämpning på den inverterade pendeln.

Projekt 15† — Adaptiv reglering av tappvarmvatten. Industriprojekt.

Projekt 16†	Adaptiv reglering med neuronnät. Undersök möjligheterna att adaptera med hjälp av neuronnät. Baserat på en artikel i CSM. Klassificera stegsvar med neuronnät. Generera stegsvar för olika system. Klassifiera dem med hjälp av ett framkopplingsnät. Prova också att klassa svaren med hjälp av Kohonens nät. Använd verktygslåda i Matlab.
Projekt 17*†	Reglering av inverterad pendel. Många olika delar. Kalmanfilter, adaptivt Kalmanfilter. Linjäriserad stabilisering. Återkopplingslinjärisering. Uppkast av pendel baserad på energibetraktelser.
Projekt 18*†	Adaptiv motorstyrning.
Projekt 19†	Dual reglering.

---

* Samarbete med datorimplementering möjligt.
† Kan byggas ut till exjobb.

PROJEKT I
DATORIMPLEMENTERING AV REGLERSYSTEM OCH ADAPTIV REGLERING

Adaptiv reglering av hårddiskarm

Magnus Johansson, E88
Mats Linde, E87
Magnus Strandh, E87
Anders Ströbeck, F87

Lunds Tekniska Högskola
Institutionen för reglerteknik
April 1992

Sammanfattning

I denna rapport redovisas vårt kombinerade projekt i kurserna *Adaptiv reglering* och *Datorimplementering av reglersystem* vilka ges vid institutionen för reglerteknik vid Lunds tekniska högskola. K J Åström m.fl har publicerat en artikel om hur man implementerar en PID regulator på en DSP, för att styra läsarmen på en hårddisk. Vår uppgift bestod i att undersöka om man med en indirekt självinställande regulator (STR) beräknat på en förenklad processmodell, $G(s) = \frac{Kp}{s^2}$, kunde styra en modell av läsarmens dynamik. Simuleringarna har gjorts i Simnon och regulatorn har sedan implementerats i realtid. Som programmeringsspråk använde vi oss av Oregons Software's Modula-2 med realtidsprimitiver från institutionen. Slutsatsen av projektet är att det går bra att styra läsarmen med en indirekt STR. Vi har lyckats komma upp i ungefär samma hastighet som med en PID regulator implementerad på en DSP.

Innehåll

1 Inledning

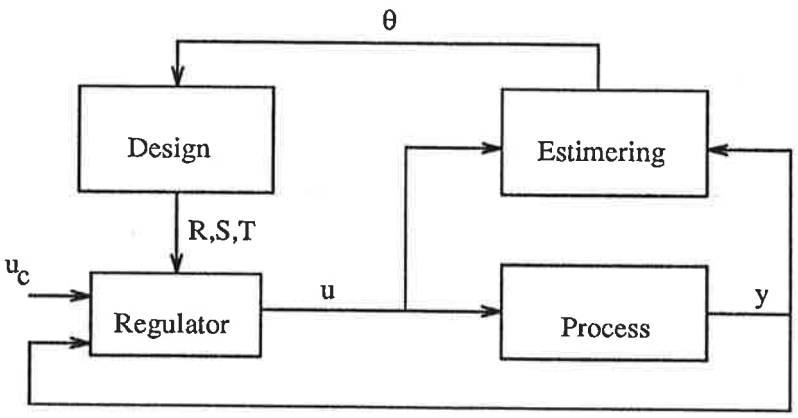
Vi har som uppgift att implementera en indirekt självinställande regulator (indirekt STR) för läsarmen på en hårddisk. Figur 1 visar en principiell uppbyggnad av en STR. Processen beskrivs som

$$G(s) = \frac{Kp}{s^2} \frac{\omega^2}{s^2 + 2\zeta\omega + \omega^2}$$

Som modell av processen i regulatorn används

$$G(s) = \frac{Kp}{s^2}$$

Förstärkningen Kp är okänd, och skall estimeras. Målet med projektet är att se hur snabb regulatorn för modellen kan bli utan att påverkas av ω i processen.



Figur 1: Principiell uppbyggnad av indirekt STR

2 Bakgrund

För att få en uppfattning hur estimeringen och regulatorn beter sig med olika parameterval simulerar vi först i Simnon. Regulatorn implementeras därefter i MODULA-2 på SUN arbetsstation. Processen simuleras på en analogimaskin. Processens förstärkning Kp är 72, egenfrekvensen ω är 2 kHz och ζ är 0.1.

Regulatorn påverkas av två störningar. En lågfrekvent laststörning som beror på tryck-drag som anslutningskablarna gör på armen. Denna störning

anses konstant vid ett visst läge på armen. Den andra störningen beror på att skivan är excentrisk, vilket leder till ett periodiskt följarfel. Felet har mycket liten amplitud, och kan beskrivas som en sinusvåg med frekvens 60 Hz.

För att kunna undersöka betendet hos processen nära ω krävs en samplingsfrekvens på cirka 10 kHz. På grund av begränsningar i arbetsstationerna kan vi inte sampla med mer än 100 Hz. Vi är tvungna att skala ner frekvensplanet med en faktor 200. Detta ger att
$\omega = 2kHz \Rightarrow \omega = 10Hz$
$60Hz \Rightarrow 0.3Hz$

3 Design av estimator

Vi använder rekursiv minstakvadratmetod (RLS) som estimeringsalgoritm för att skatta processparametern. Den är enkel i sin form och lätt att implementera. Algoritmen beskrivs i kapitel 3 i Adaptive Control av K J Åström, B Wittenmark
Vid estimeringen använder vi $G(s) = \frac{Kp}{s^2}$ som modell av processen. Endast Kp är okänd och skall skattas. Sampling av processen ger

$$H(q) = \frac{Kph^2}{2}\frac{q+1}{(q-1)^2}$$

Utsignalen y ges av

$$y(k) = \frac{Kph^2}{2}\frac{q+1}{(q-1)^2}u(k)$$

Kunskap om processen ger att u filtreras vilket ger v som

$$v(k) = \frac{q+1}{(q-1)^2}u(k)$$

$$v(k) = u(k-1) + u(k-2) - v(k-2) + 2v(k-1)$$

Sätt

$$\varphi = v(k)$$

och

$$\theta = \frac{Kph^2}{2}$$

vilket ger

$$y = \varphi\theta(RLS) \Rightarrow \hat{\theta}$$

RLS-algoritm för skattning av θ med exponentiell adaptionsparameter

$\varphi = u(k-1) + u(k-2) - \varphi(k-2) + 2\varphi(k-1)$

Beräkning av residual
$e = y - \varphi\hat{\theta}$

Uppdatera estimat
$k = P\varphi$
$den = \varphi^T k + \lambda$
$\theta = \theta + ke/den$

Uppdatera kovarians
$P = (P - k^2/den)/\lambda$

Uppdatera regressionsvektorn
$u(k-1) = u(k)$
$\varphi(k-2) = \varphi(k-1)$
$\varphi(k-1) = \varphi(k)$

4 Design av filter

För att estimeringen inte skall påverkas av laststörning och högfrekvent brus filteras u och y med ett bandpassfilter. Design av filtret gjordes i MATLAB med funktionen BUTTER. Brytfrekvenserna valde vi

$$\omega_1 = 0.02$$

och

$$\omega_2 = 0.1$$

av Nyqvistfrekvensen. Filtrets struktur är

$$H_f(q) = \frac{b_0 q^4 + b_2 q^2 + b_4}{q^4 + a_1 q^3 + a_2 q^2 + a_3 q + a_4}$$

med koefficienterna

$$b_0 = 0.0134$$
$$b_2 = -0.0267$$
$$b_4 = 0.0134$$
$$a_1 = -3.6113$$
$$a_2 = 4.9298$$
$$a_3 = -3.0190$$
$$a_4 = 0.7009$$

5 Regulatordesign

Regulatorn har en utsignal u, och två insignaler, referenssignalen u_c, och den mätta utsignalen y. En generell struktur för en regulator med dessa insignaler och utsignaler ges av

$$Ru = Tuc - Sy$$

R, S och T är polynom skrivna i operatorn q. För att undvika stationärt fel innerhåller R en integrator enligt

$$R = (q-1)R'$$

och för att undvika vikning så är

$$S = (q+1)S'$$

Önskat slutet system definieras av

$$Gm(s) = \frac{\omega^2}{s^2 + 2\zeta\omega + \omega^2} \Rightarrow Hm(q) = \frac{b_1 q + b_2}{q^2 + a_1 q + a_2} = \frac{Bm}{Am}$$

Modell av processen är enligt tidigare

$$G(s) = \frac{Kp}{s^2} \Rightarrow H(q) = \frac{Kph^2}{2}\frac{q+1}{(q-1)^2} = \hat{\theta}\frac{q+1}{(q-1)^2} = \frac{B}{A}$$

Modellens nollställe ligger på enhetscirkeln och bör inte förkortas. Ekvationen för det slutna systemet ger då

$$\text{AR+BS=AoAm}$$

På grund av att vi inte förkortar några nollställen i modellen måste B vara en faktor i Bm för lösbarhet

$$Bm = BBm'$$

Önskat slutet system blir då

$$Gm(q) = \frac{BBm'}{Am}$$

där Bm' väljs till en konstant så att stationära förstärkningen blir 1.

$$Bm' = \frac{2 + a_1 + a_2}{2\hat{\theta}}$$

För att lösningen skall vara kausal måste

$$degAo \geq 2degA - degAm - 1 = 1$$

Välj degAo=3, vilket ger att fem regulatorparametrar kan beräknas.

$$Ao = q^3 + a_0^o q^2 + a_1^0 q + a_2^0$$

Detta ger att R', S' och T kan väljas till andra ordningen som

$$R' = q^2 + r_1 q + r_2$$

$$S' = q^2 s_0 + q s_1 + s_2$$

$$T = Bm' Ao$$

Observera att R' är monisk.

Lösningen till det slutna systemet
$AR + BS = AoAm \Rightarrow$
$q^5 = q^5$
$q^4 : r_1 - 3 + \hat{\theta} s_0 = a_1 + a_0^o = \bar{a}$
$q^3 : r_2 - 3r_1 + 3 + \hat{\theta}(s_1 + 2s_0) = a_2 + a_0^o a_1 + a_1^o = \bar{b}$
$q^2 : 3r_1 - 3r_2 - 1 + \hat{\theta}(s_0 + 2s_1 + s_2) = a_0^o a_2 + a_1^o a_1 + a_2^o = \bar{c}$
$q : 3r_2 - r_1 + \hat{\theta}(s_1 + 2s_2) = a_1^o a_2 + a_2^o a_1 = \bar{d}$
$0 : -r_2 + \hat{\theta} s_2 = a_2^o a_2 = \bar{e}$

Skrivet i matrisform blir det

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ -3 & 1 & 2 & 1 & 0 \\ 3 & -3 & 1 & 2 & 1 \\ -1 & 3 & 0 & 1 & 2 \\ 0 & -1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r1 \\ r2 \\ \hat{\theta}s0 \\ \hat{\theta}s1 \\ \hat{\theta}s2 \end{pmatrix} = \begin{pmatrix} \bar{a} + 3 \\ \bar{b} - 3 \\ \bar{c} + 1 \\ \bar{d} \\ \bar{e} \end{pmatrix} \Rightarrow$$

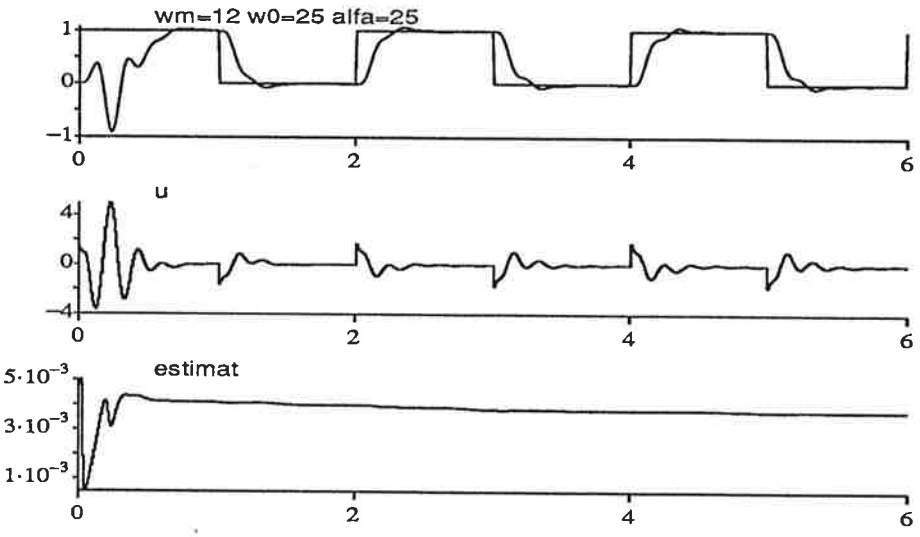
R- och S-polynomens koefficienter ges då av

$$\begin{pmatrix} r1 \\ r2 \\ \hat{\theta}s0 \\ \hat{\theta}s1 \\ \hat{\theta}s2 \end{pmatrix} =$$

$$\begin{pmatrix} 0.3125 & -0.1875 & 0.0625 & 0.0625 & -0.1875 \\ 0.1875 & -0.0625 & -0.0625 & 0.1875 & -0.3125 \\ 0.6875 & 0.1875 & -0.0625 & -0.0625 & 0.1875 \\ -0.6250 & 0.1250 & 0.3750 & 0.1250 & -0.6250 \\ 0.1875 & -0.0625 & -0.0625 & 0.1875 & 0.6875 \end{pmatrix} \begin{pmatrix} \bar{a}+3 \\ \bar{b}-3 \\ \bar{c}+1 \\ \bar{d} \\ \bar{e} \end{pmatrix}$$

6 Simulering i Simnon

Regulatorn och estimeraren implementerades diskret, men däremot valde vi att implementera processen som ett kontinuerligt system.
Regulatorn har parametrarna ω , ω_o, alfa och dämpningarna ζ och ζ_o. ζ och ζ_o valde vi til 0.707. Observerarens dynamik ω_o och alfa valde vi enligt strategin dubbelt så snabba som systemets dynamik. Efter ett antal simuleringar fann vi att
$\omega_o = 25$
alfa=25
$\omega = 12$
var det snabbaste vi kunde få systemet att svara på ett steg utan att påverkas av processens egenfrekvens. Simuleringen gjordes med en sinusstörning på u_c.



Figur 2: Stegsvar för det slutna systemet

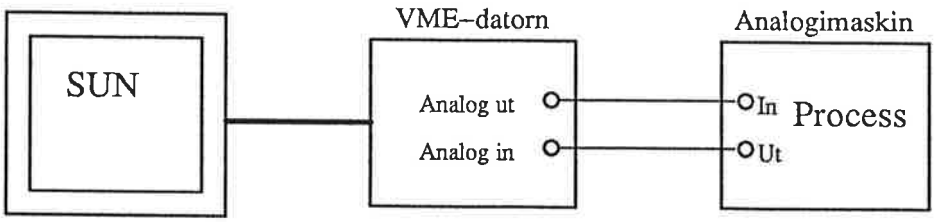
7 Realtidsimplementering

7.1 Inledning

Realtidsimplementeringen är gjord i Oregons Software's Modula-2 med realtidsprimitiver från institutionen för reglerteknik. Utvecklingen är gjord på Sun arbetsstationer i Unix miljö. Tidigare under kursen genomfördes ett övningsprojekt vilket resulterade i ett programpaket för reglering av ett DC-servo. Vi har använt nämnda programpaket som bas för projektet. Modulen Refgen har helt plockats bort. Modulen Regul används, men är av en helt annorlunda struktur. Vidare är modulen Opcom (med tillhörande OpcomLoop och Command) obetydligt förändrad och bara anpassad till aktuella processens kommando. Slutligen använder vi oss även av modulen Plotter. Plotter är uteslutande skriven av institutionen och vi använder den tacksamt.

7.2 Beskrivning av uppgiften

Hårddiskarmens dynamik, se kapitel 1 på sidan 1, simuleras och implementeras på en av institutionens gamla analogimaskiner. Analogimaskinen betraktas sedan som vår process, med en ingång och en utgång. Då realtidsegenskaperna i Unix-miljön inte räcker till för effektiva och snabba regleralgoritmer använder vi ett fristående målsystem kallat VME-datorn. Programmet utvecklas, kompileras och länkas under Unix och sedan laddas det ner i VME-datorn, via ett enkelt kommunikations-program (SIMCOM). VME-datorn finns tillsammans med AD/DA-omvandlare. När programmet har startat på VME-datorn fungerar arbetsstationen som en enkel lokalt ansluten operatörskonsol. Från konsolen styrs sedan processsen med hjälp av några kommando. För att kunna se hur processen uppför sig behöver vi någon form av grafisk representation. Vi använder oss därför av programmet Matlab för plottning.



Figur 3: Schematisk bild över utvecklingsmiljön

7.3 Användarhandbok

7.3.1 Tillgängliga kommando

Användaren kan från konsolen hantera vissa parametrar samt starta och stoppa regulatorn. Kommandona listas här med tillhörande beskrivning.

help ger en listning på tillgängliga kommandon

regpar listar regulatorns parametrar med nuvarande värde. Kommandot möjliggör också en ändring av parametervärde, se kapitel 7.3.3.

estpar hanterar estimatorn på samma sätt som *regpar* ovan.

diston lägger till en sinusstörning till fyrkantsreferensen.

distoff stänger av samma störning och återställer referensen till en fyrkantssignal.

restart nollställer estimatorn och regulatorn. *restart* ges för att regulatorn ska ta hänsyn till en parameterändring.

quit terminerar regulatorn och hela programmet.

7.3.2 Beskrivning av parametrarna

Det finns två uppsättningar parametrar, en för regulatorn och en för estimatorn. Nedan följer en listning med tillhörande beskrivningar på regulatorns parametrar. För att se vilka parametrar som finns ge kommandona *regpar* och *estpar*.

h är samplingsintervallet och ges i sekunder.

RefAmpl är amplituden på fyrkantreferensen. Värdet ska ligga i intervallet $[-1.0, 1.0]$. *Observera* att värdet måste anges med decimalpunkt även om det är ett heltal!

RefPeriod är fyrkantreferensens periodtid och anges i sekunder.

DistAmpl är sinusstörningens amplitud. Se *RefAmpl* ovan.

DistPeriod är sinusstörningens periodtid och anges i sekunder.

Slutligen ges en listning på estimatorns parametrar.

P är P-marisens initialvärde. Här är P bara en skalär.

lambda är adaptionskoefficienten och ges värde ≤ 1

zm är dämpningen för slutna systemet.

Wm är naturliga vinkelfrekvensen hos det slutna systemet.

ao är koefficienten till observerarens tredje pol.

zo är dämpningen för observeraren.

Wo är naturliga vinkelfrekvensen hos observeraren.

7.3.3 Syntax för ändring av parametervärde

Syntaxen för ett kommando ser ut som följer

kommandonamn [parameternamn1 [parametervärde1] ...]

Om parameternamn och parametervärde utelämnas innebär det att alla parametrar listas med tillhörande värde. Utelämnas parametervärde listas bara angiven parameter. Anges ett parameternamn i par med ett parametervärde ändras värdet. På så sätt kan man på samma rad ange listning av t.ex en parameter samtidig som man ändrar värdet på t.ex två. Nedan följer några exemel.

För att ändra t.ex samplingsintervallet i regulatorn till 0.01 sekunder skriv från konsolen
>>regpar h 0.01

Programmet bekräftar ändringen genom att skriva ut parameternamn och värde.

För att ändra samplingsintervallet till 0.01, titta hur stor amplituden på referensen är och samtidigt ändra sinusstörningens periodtid till 20 sekunder skriv
>>regpar h 0.01 refampl distperiod 20.0

Så länge kommandoraden inte blir tvetydig går det att använda sig av en kortform på kommandonamn och parameternamn. Senaste exemplet kan då skrivas som
>>reg h 0.01 refa distp 20.0

Observera att alla värde måste anges med decimalpunkt även om dom är heltal ! Glöm inte att kommandot *restart* måste ges efter en parameterändring för att ändringen ska träda i kraft.

7.4 Övergripande beskrivning

Programmet är uppbyggt av sju fristående moduler.

- Main
- Opcom
- OpcomLoop
- Command
- Estim
- Regul
- Plotter

Totalt snurrar fyra parallella processer. Två av dem startas från Opcom, en för den lokala konsolen på Sun och en för en extra konsol som kan öppnas från Matlab. Tanken är att t.ex en systemingenjör ska kunna styra processen från en konsol via Matlab, utföra beräkningar, plotta kurvor, utvärdera, ta fram nya parametrar och sedan applicera dessa på processen o.s.v. Vi har inte haft tid att prova Matlab-konsolen (vilken implementerades redan under övningsprojektet) varför vi lämnar den utan vidare kommentarer. En process utnyttjas av regulatorn och ytterligare en för plottningen.

7.5 Gränssnitt mot användare

Nedan följer en beskrivning av i programmet ingående moduler. Mer upplysning om varje modul finns att hämta i respektive definitionsmodul från programlistningarna.

Main.mod Huvudprogram.

Opcom.def Snitt mot övriga moduler med vilka här menas Regul, Plotter och Main.

Opcom.mod Fysisk systemkonfiguration d.v.s. vilka konsoler och hur man kommunicerar med dem bestäms här.

OpcomLoop.def Snitt mot kommandotolken. Endast en primitiv finns tillgänglig, DoLoop, vilken anropas av konsol-processerna.

OpcomLoop.mod Kommandotolk oberoende av hur kommandona ser ut men beroende av vilka kommando som finns.

Command.def Snitt mot övriga moduler. Innehåller primitiver för varje tillgängligt kommando. Kommandotolken (OpcomLoop) använder sig av dessa.

Command.mod Innehåller kommandonamn och argument.

Estim.def Snitt mot övriga moduler. Regulatorn kommunicerar och synkroniseras med Estim via primitiver Estimate, Restart och Init. Primitiverna GetParNames, GetPars och SetPars utnyttjas av Command för hantering av Estims parametrar.

Estim.mod Implementering av RLS parameterskattare.

Regul.def Snitt mot övriga moduler. Command har som i Estim tillgång till tre primitiver för att hantera parametrarna. Vidare finns primitiver för att utföra vissa kommando. Dessa anropas av Command.

Regul.mod Implementation av en indirekt självinställande regulator.

Plotter.def Snitt mot övriga moduler. Regul använder sig av tre primitiver för att möjliggöra plottnig av önskade signaler. Main använder de övriga två för uppstart och terminering.

Plotter.mod Implementation av plotterrutiner. Plotter använder sig av en biblioteksrutin, Logger.

7.6 Processbeskrivning

7.6.1 ConsoleProcess

Processen startas från Opcom.Init. Processen anropar DoLoop i OpcomLoop med skriv- och läsprocedurer, aktuella för Sun-konsolen, som argument. DoLoop är processens loop vars uppgift består i att kontinuerligt skriva ut en prompt på skärmen och därefter avkoda eventuell inmatad kommandorad för att slutligen anropa rutiner som utför kommandot.

7.6.2 MatlabProcess

Fungerar som ovan men anropar DoLoop i OpcomLoop med skriv- och läsprocedurer, aktuella för Matlab-konsolen, som argument.

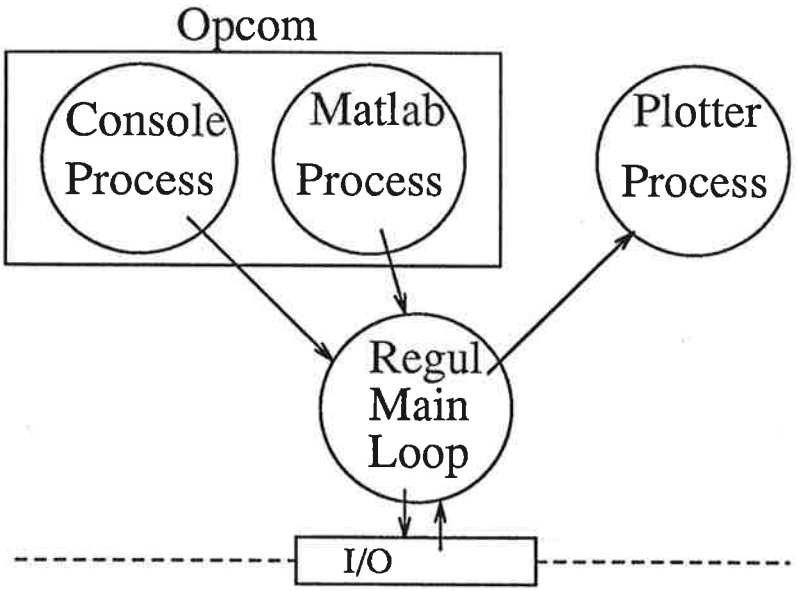
7.6.3 RegulMainLoop

Detta är själva hjärtat i regulatorn. Här sker tids-synkroniseringen under ett samplingsintervall. Referenssignal med eller utan störning genereras. Inläsning av processvärdet via AD-omvandlare görs och därefter beräknas en ny styrsignal vilken omedelbart ställs ut via DA-omvandlare. Därefter anropas Estimate i Estim för att estimera nya regulatorparametrar, vilka returneras vid anropet. Den skattade processparametern returneras också. Vidare görs en kontroll om regulatorn blivit terminerad av operatören och om så inte är fallet uppdateras slutligen signalerna som ska plottas.

7.6.4 PlotterProcess

Plotterprocessen har vi inte satt oss in i och kan då givetvis inte beskriva den. Tillräckligt med tid har lagts på det övriga programmet.

7.7 Processgraf



Figur 4: Realtidsprocesser i systemet

7.8 Datastrukturer

För att skydda de variabler som används både av operatören och programmet använder vi monitorer. I Regul återfinns följande datastruktur

```
TYPE NameType    = ARRAY [0..NameLength] OF CHAR;
     ParNameType = ARRAY [1..NrOfPars] OF NameType;
     ParType     = ARRAY [1..NrOfPars] OF LONGREAL;

     RegParType = RECORD
                    CASE :BOOLEAN OF
                      TRUE  : Par :ParType|
                      FALSE : h, RefAmpl, RefPeriod,
                              DistAmpl, DistPeriod : LONGREAL
                    END;
                  END;

     RegParNameType = RECORD
                        CASE :BOOLEAN OF
                          TRUE  : Par :ParNameType|
                          FALSE : h, RefAmpl, RefPeriod,
                                  DistAmpl, DistPeriod : ParNameType
                        END;
                      END;

     MonitorType = RECORD
                     Mon : Monitors.MonitorGate;
                     RegParameters : RegParType;
                     Dist_on : BOOLEAN;
                   END;
```

Strukturen möjliggör att man kan referera till variablerna med samma namn som operatören gör. Programmet blir på det viset mer lättläst. I Regul finns sedan sju monitorprocedurer som hanterar de skyddade variablerna på olika vis. Typerna NameType, ParNameType och ParType är synliga för användare av modulen.

I Estim används nedanstående datastruktur

```
TYPE PolynomType = RECORD
                     a0,a1,a2,a3:LONGREAL;
     (* a0,a1,a2 and a3 are the coefficients of the 3rd,2nd 1st *)
     (* and 0th term respectively.                              *)
                   END;

     NameType     = ARRAY [0..NameLength] OF CHAR;
     ParNameType = ARRAY [1..NrOfPars] OF NameType;
     ParType      = ARRAY [1..NrOfPars] OF LONGREAL;

     EstimParType = RECORD
                      CASE :BOOLEAN OF
                        TRUE : Par : ParType|
                        FALSE: P,lambda,zm,wm,ao,zo,wo : LONGREAL
                      END; (* CASE *)
                    END;

     EstimParMonType = RECORD
                         Mon : MonitorGate;
                         Parameters : EstimParType;
                       END;

     RegParMonType   = RECORD
                         Mon : MonitorGate;
                         r1,r2,s0,s1,s2,t0,t1,t2,t3 : LONGREAL;
                       END;
```

Strukturen är i sin uppbyggnad identisk med den i Regul. Estimatorn i sin tur har 5 monitorprocedurer. Typerna PolynomType, NameType, ParNameType och ParType är synliga för användare av modelen. PolyonomType används för de estimerade regulatorparametrarna som Regul får av Estim.

7.9 Problem

Ett problem har varit den något obeprövade VME miljön. Mycket tid har gått åt till fel som vi inte har kunnat hantera själva, mestadels hårdvarufel. Vidare har det bara funnits ett fungerande målsystem vilket har gjort arbetet ganska tungrott när flera grupper har jobbat samtidigt. Den för oss okända modulen Plotter var lite svår att få igång. Den utnyttjar Matlab och vissa mjukvaruproblem dem emellan fanns i begynnelsen.
Nu till problem med vårt program. En felaktig inbördes placering mellan proceduren för uppdatering och proceduren för beräkning av styrsignal orsakade oss mycket huvudbry. En och en halv dag tog det att hitta det felet.
Inför Simnon simuleringen beräknades filterkoefficienter i Matlab. Det visade sig att en felavrundning i fjärde decimalen (!) av en filterkoefficient ställde till stora problem i realtid.

A Programlistningar

Projekt i adaptiv reglering och
datorimplementering av reglersystem

Snabb självinställande regulator

Anders Carlsson
Peter Fransson
Torbjörn Olsson
Anders Robertsson
Jörgen Svensson

15 juli 1992

Handledare: Anders Blomdell
Karl Johan Åström

Inledning

Vårt projekt gick ut på att implementera en indirekt självinställande regulator (STR) med hjälp av signalprocessorkortet NB-DSP-2300, som är monterat i en Macintosh II. Vi använde oss även av en likströmsmotor som vi positionsreglerar, för att efterhand testa koden som skrevs. Vi utgick från en lösning av professor K.Jagannathan. Hans lösning var emellertid specialiserad för ett visst problem och ganska ostrukturerat skriven. Vi ville göra en generell lösning och strukturera programmet ordentligt, så att det blev mera lättläst och lättare att förstå. Ett mål var också att optimera kritiska bitar av koden så att samplingshastigheten kunde ökas. Om tiden räckte till tänkte vi även använda LabView för presentation av signaler och inställning av parametrar.

Teknisk beskrivning, mjukvara

Självinställande regulator

En indirekt självinställande regulator skattar processens parametrar med hjälp av en **estimator**. Parametrarna skickas till en **designalgoritm** som bestämmer **regulatorns** parametrar. På så vis påverkas inte reglerprestanda av processvariationer.

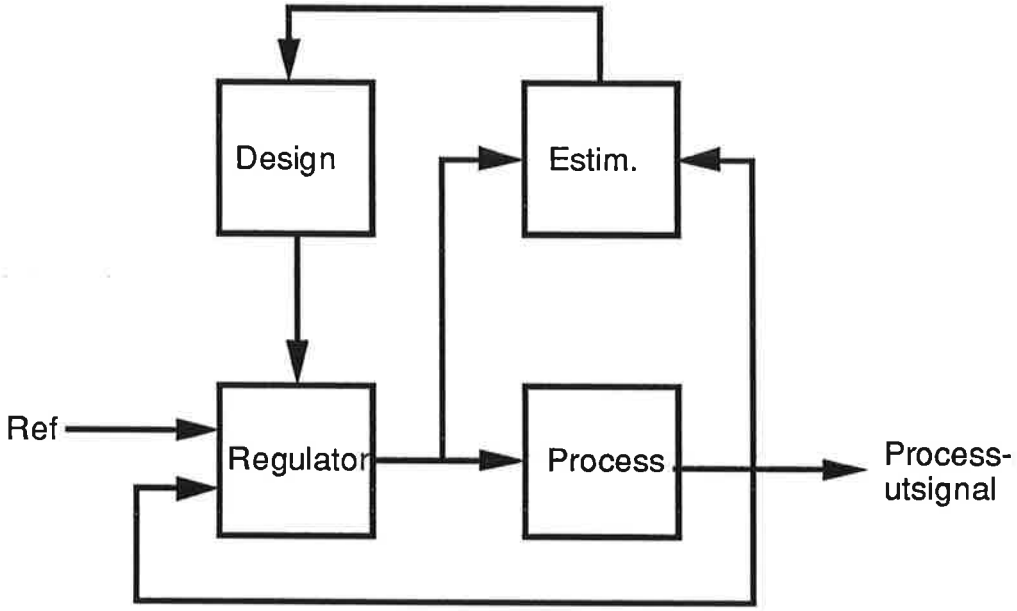


Fig 1 Principskiss, STR

Regulator

Vår regulator kan beskrivas på formen :
$R^*(q^{-1})\ u=T^*(q^{-1})\ y_{ref}-S^*(q^{-1})y$

där u är styrsignalen, y_{ref} är referenssignalen och y är mätsignalen.

Gamla styrsignaler,referenssignaler och mätsignaler lagras i vektorer för senare användning i styrsignalberäkningen. Regulatorn innehåller en antiwindup funktion, som begränsar styrsignalen innan den lagras .

Polynomet $R^*(q^{-1})$ är alltid moniskt dvs :
$R^*(q^{-1})=1 + r_1(q^{-1}) + r_2(q^{-2})+. . . . . + r_{nr}(q^{-nr})$.
Detta innebär att styrsignalen u kan beräknas som:
$u = -[\ r_1\ u(t-1)\ +r_2\ u(t-2) + . . . + r_{nr}\ u(t-nr)] + T^*(q^{-1})\ y_{ref} - S^*(q^{-1})y$

Beräkningen av styrsignalen görs i en avbrottsrutin, som körs varje gång A/D-omvandlaren har gjort en omvandling klar. Detta beskrivs i hårdvarubeskrivningen.

Estimator

Estimatorn hämtar de filtrerade styr- och mätsignalerna från processen. Processparametrarna skattas med hjälp av kvadratrots-RLS [1]. Algoritmen innehåller en glömskefaktor (lambda) som gör att estimatorn lägger större vikt vid de sist insamlade signalerna.

Design

Designalgoritmen får processens parametrar från estimatorn och beräknar regulatorparametrarna genom att lösa den diophantiska ekvationen :
$A\ R + B\ S = Ao\ Am$.
Algoritmen lägger automatiskt in en integrator samt ett nollställe vid halva samplingsfrekvensen
$A\ R'\ (q-1) + B\ S'\ (q+1) = Ao\ Am$
Ekvationen löses genom att sätta upp Sylvestermatrisen och lösa den med Gauss-elimination.

I dess nuvarande utformning tar beräkningen av regulatorparametrarna betydligt längre tid än någon annan del av

programmet. Detta innebär dock inte att det är dess cykeltid som sätter undre gräns för sampelfrekvensen eftersom regulatorparametrarna inte behöver uppdateras för varje mätcykel.

Filterdesign

Inget tidsdiskret system fungerar säkert utan filtrering av (in)signalerna. I den här sortens system inför man filter på ingången till hela regulatorn, filter för insignalerna till estimatorn, samt eventuellt något rekonstruktionsfilter på utgången. För att strukturera dessa filter har vi implementerat två klasser av filter, tillsammans med metoder för att mata in signaler till dem, hämta signaler från dem samt (givetvis) för att skapa och förstöra dem.
Den ena klassen av filter implementerar en "Direkt form 2" struktur för ett generellt linjärt tidsdiskret filter. Denna klass lämpar sig bäst för filter med få poler (helst färre än tre) och FIR-filter, dvs helt pollösa filter. Dessutom har funktioner införts i klassen så att man kan använda filtren av denna klass för omvandling av sampelfrekvens (Decimering, interpolering och konvertering med rationellt förhållande mellan sampelfrekvenserna). Dessa funktioner utnyttjas i ingångs- och utgångsfiltren.
Den andra klassen implementerar filter bestående av kaskadkopplade första eller andra ordningens filter, s.k Biquads. Denna realisering har betydligt bättre numeriska egenskaper för filter med många poler, vilket är fördelaktigt när man skall åstadkomma IIR-filter med skarp avskärning vid gränsfrekvensen. Även i denna klass har införts funktioner för att möjliggöra skilda samplingsfrekvenser på in- och utgångarna.
Denna strukturering, tillsammans med integrerade filterberäkningsrutiner, har gjort det möjligt att enkelt kunna pröva egenskaperna för regulatorsystemet med olika typer av filtrering på olika platser.

Teknisk beskrivning, hårdvara

Regulatorn använder sig av två kort från National Instruments: NB-DSP-2300 och NB-MIO16.

Signalprocessorkortet innehåller flyttalsprocessorn TMS320C30. Den har en klockfrekvens på 33 MHz och kan t.ex. utföra en multiplikation **och** en addition under samma klockcykel. Detta är speciellt användbart vid användning av långa filter. Vi använder denna finess i vårt antialiasingfilter, som är 200 tappar långt.

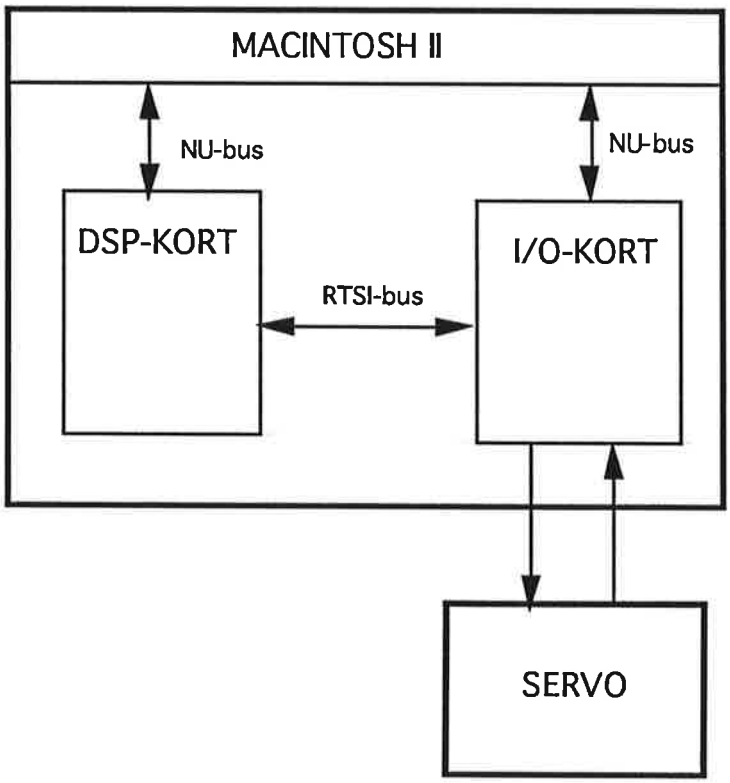


Fig 2 Hårdvara

I/O-kortet innehåller förstärkare, A/D-D/A-omvandlare och timingkretsar. Signalprocessorns inbyggda timer skickar ett timeravbrott med ett mellanrum som bestäms av samplingshastigheten. I timer-avbrottsrutinen startas en A/D-omvandling genom att skriva i ett "Start-convert"-register via NuBusen. När en A/D-omvandling är klar skickas ett avbrott via RTSI-bussen till DSP-kortet. Detta görs för att kunna utföra beräkningar i huvudprogrammet under tiden omvandling sker.

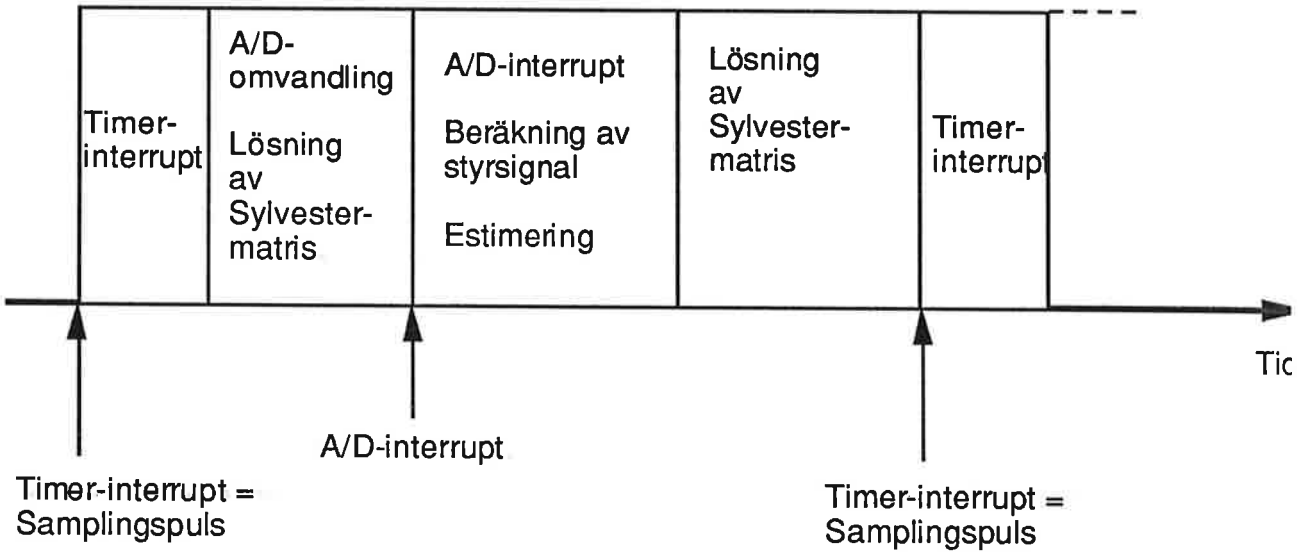


Fig 3 Timing

Vårt projekt har inte varit problemfritt.

* Vi har försökt lösa kommunikationen mellan I/O-kortet och signalprocessorn genom att ansluta en RTSI-busskabel mellan dem. Det första problemet vi stötte på var att A/D-kortets avbrott hade högre prioritet än debuggern, vilket innebar att macen ideligen hängde sig. Detta löste vi genom att först i timeravbrottet sätta på A/D-avbrottet och därefter stänga av det efter A/D-rutinen.

* I/O-kortet bytte mätkanal utan att vi ändrade något i programmet. Efter några dagar rättade det själv till sig ?!?.

* Av en händelse tog vi bort RTSI-bussen. Döm om vår förvåning när regulatorn fungerade lika bra som innan! A/D-avbrottet har av någon anledning varit aktivt hela tiden. Detta förklarar även de tidigare problemen med debuggern. Detta berodde på att processorn själv lade ut ett avbrott på samma kanal som den läste av. Genom att ändra en bit i ett register undveks detta problem.

* C-kompilatorn har ett fatalt fel. Om man i sitt program skriver "a=-b-1" översätter kompilatorn detta så att a=b+1.

* Det programavsnitt som skattar nya regulatorparametrar , dvs lösningen av den 'Diofantiska ekvationen', tar väldigt lång tid (ca 2ms !!!). Detta medför att denna rutin måste anropas i huvudprogrammet. Det utförs alltså inte någon uppdatering av regulatorn lika ofta som modellen skattas utan den utnyttjar återstående tid av en mätcykel för beräkning. Hur ofta regulatorn kommer att uppdateras kommer följdaktligen att bero på sampelfrekvensen.

Sammanfattning

Syftet med vårt projekt var att göra en generell indirekt adaptiv regulator. Detta medförde att vår samplingsfrekvens inte riktigt kunde uppnå vad professor K.Jagannathans implementering gjorde. Hans regulator klarade en samplingsfrekvens på närmare 3 kHz, vilket inte verkar orimligt, ty hans problem var specialiserat och det gjorde att han kunde skräddarsy sina algoritmer.
Enligt våra beräkningar kommer vår regulatorn att kunna köras på ca 2 kHz, beroende på vilken ordning av system som används. Flaskhalsen i systemet, vad gäller tiden, är lösningen av den Diophantriska ekvationen. Vi använde oss här först av en metod som med hjälp av Gauss-elimination löste Sylvestermatrisen, men fann senare en metod, polynomreduceringsmetoden, som var ca tre gånger snabbare och dessutom klarade av gemensamma faktorer på vänster respektive högersida av den Diophantriska ekvationen.

Projekt i adaptiv reglering

Implementering av adaptiv regulator i LabVIEW

Per Lantorp E-88

920519

Handledare:

Per-Olof Källén

Karl-Johan Åström

Inledning

Mitt projekt gick ut på att implementera en adaptiv regulator i LabVIEW. En indirekt självinställande regulator av andra ordningen skulle byggas och sedan provköras på ett servo.Jag har haft en hel del problem och för närvarande fungerar inte hela programmet .

Kort om LabVIEW

LabVIEW är ett programpaket till Macintosh där man bl.a. kan bygga upp regulatorer. Genom att använda symboler kan man bygga upp stora komplicerade system. Man använder dessa symboler som byggklossar och kopplar sedan ihop dem med hjälp av en tråddragningsmekanism. Det finns många färdiga byggstenar såsom aritmetiska och logiska funktioner, skiftregister och while-loopar, men man kan också skapa egna t.ex. genom att skriva C-program som sedan integreras. LabVIEW finns på Macintoshdatorer och man kan utnyttja många av dess vanliga funktioner. Här kan nämnas redigering och fönsterhantering.

Ett LabVIEW-objekt består av två delar, en frontpanel och ett diagram. Frontpanelen fungerar precis som på ett vanligt instrument, t.ex. finns där knappar och reglage som man kan trycka och vrida på. I diagrammet bygger man upp systemet genom att koppla ihop olika funktioner från LabVIEW eller egna skapade objekt. Man kan således bygga upp system i flera olika nivåer.

Indirekt självinställande regulator

Processmodellen som jag använde var följande:

$$A(q)*y(t)=B(q)*u(t)$$

$$A(q)=q^2 + a_1*q + a_2\ ,\ \ B(q)=b_1*q + b_2.$$

För att skatta a_1, a_2, b_1 och b_2 använder jag standardmodellen av den rekursiva minsta kvadratmetoden.
Regulatorn är av typen $R(q)*u(t)=T(q)*u_c(t)-S(q)*y(t)$, där

$$R(q)=(q -1)*(q - r_1)\ ,\ \ S(q)=s_0*q^2 + s_1*q + s_2\ ,\ \ T(q)=t_0*q^2+ t_1*q + t_2$$

Observerarpolynomet $A_o(q)$ är av andra ordningen med båda polerna placerade på samma ställe, dvs

$$A_o(q)=(q - a_o)*(q - a_o)$$

$A_m(q)$ är också av andra ordningen, dvs

$$A_m(q)=q^2 + p_1*q + p_2$$

där p_1 och p_2 väljs genom parametrarna omega och zeta. Den Diophantiska ekvationen som sedan löses i regularorn får då följande utseende

$$AR + BS = A_m A_o$$

$$(q^2+a_1*q+a_2)*(q-1)*(q+r_1) + (b_1*q+b_2)*(s_0*q^2+s_1*q+s_2)= (q^2+p_1*q+p_2)*(q-a_o)*(q-a_o)$$

Lösningen till denna ekvation blir

$$r_1=b_2/b_1 - (b_2^2-p_1*b_2*b_1+p_2*b_2^2)*(b_2^2+2*a_o*b_2*b_1+a_o^2*b_1^2)/ (b_1*(b_1+b_2)*(b_2^2-a_1*b_2*b_1+a_2*b_1^2))$$

$$s_0=(-2*a_o+p_1+1-a_1-r_1)/b_1$$

$$s_1=(a_o^2*p_1-2*a_o*p_2+a_2-b_1*s_2-r_1*(a_2-a_1))/b_2$$

$$s_2=(p_2*a_o^2+r_1*a_2)/b_2$$

$$t_0=(1+p_1+p_2)/(b_1+b_2)$$

$$t_1=-2*a_o*(1+p_1+p_2)/(b_1+b_2)$$

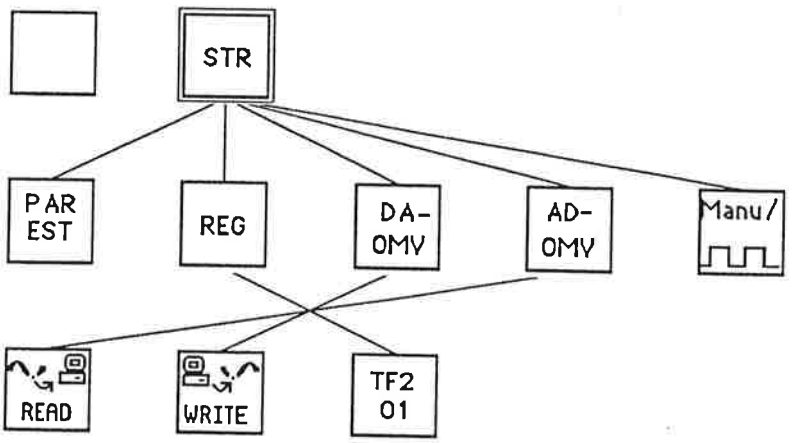
$$t_2=a_o^2*(1+p_1+p_2)/(b_1+b_2)$$

Jag använder också följande Antiwindup-regel

$$A_o(q)*v=T(q)*u_c - S(q)*y + (A_c(q) - R(q))*u$$

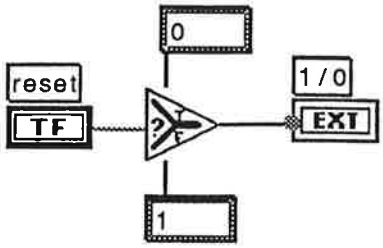
Implementering i LabVIEW

I LabVIEW har jag byggt upp systemet i tre nivåer med sammanlagt nio olika objekt. Hirarkin är följande:



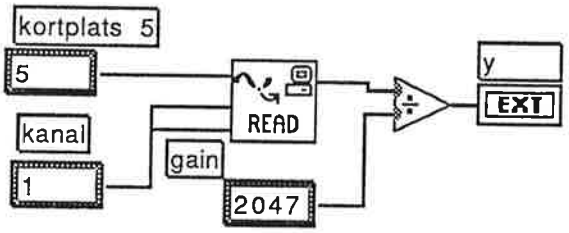
READ och WRITE är färdiga LabVIEW-funktioner som sköter inmatning och utmatning av signaler mellan datorn och omvärlden.

TF2 01 används för att nollställa r,s och t parametrarna och har följande diagram.

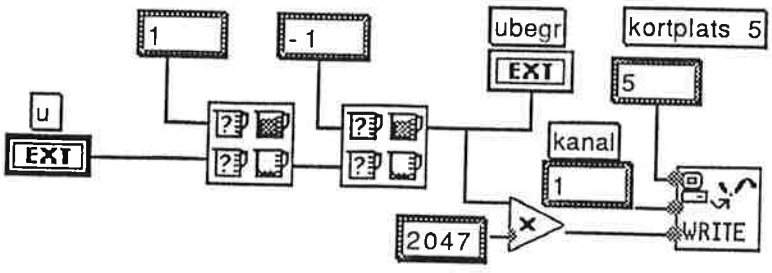


A/D-omvadlaren och D/A-omvandlaren omvandlar (naturligtvis) analoga signaler till digitala signaler och tvärtom.

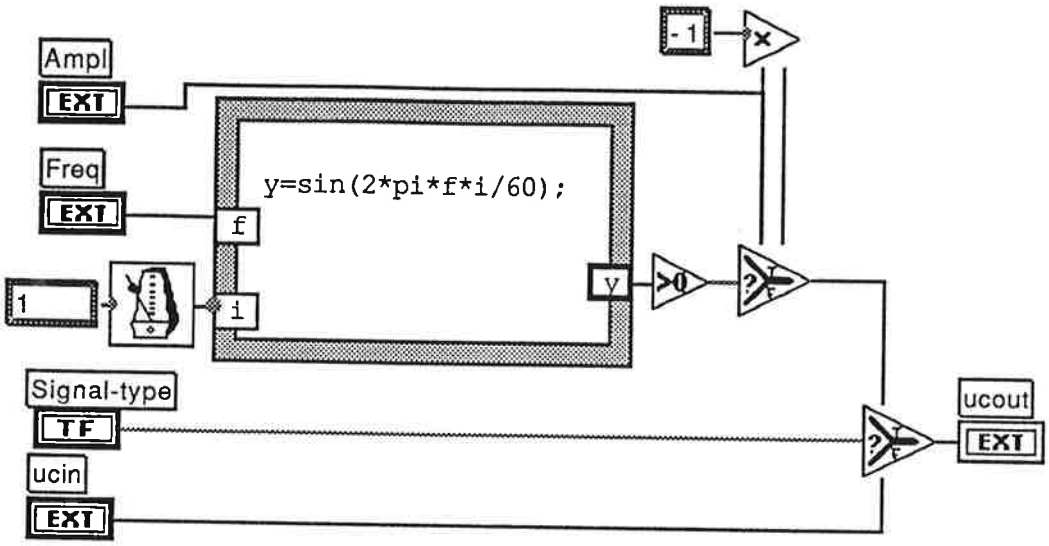
AD-omvandlare



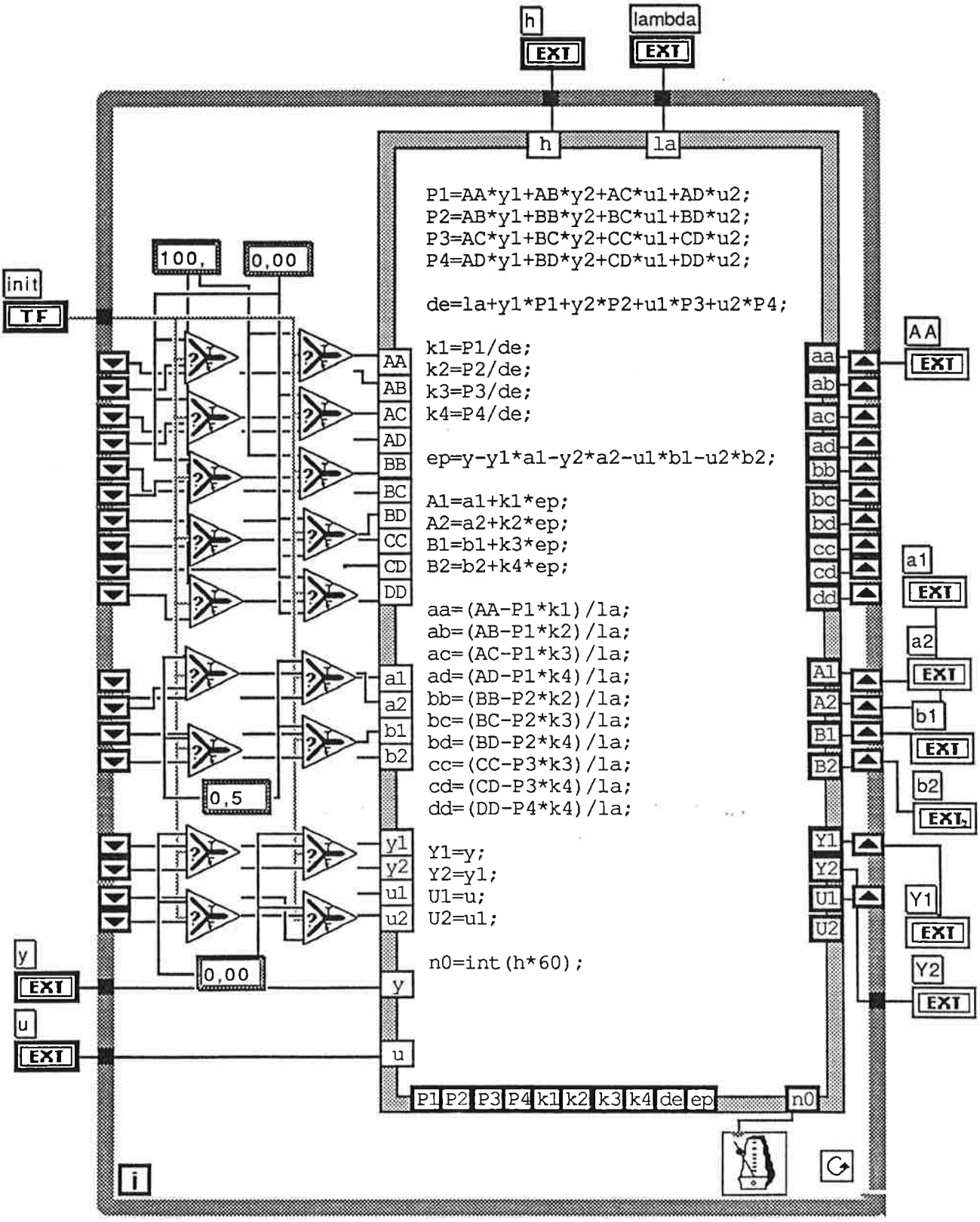
DA-omvandlare



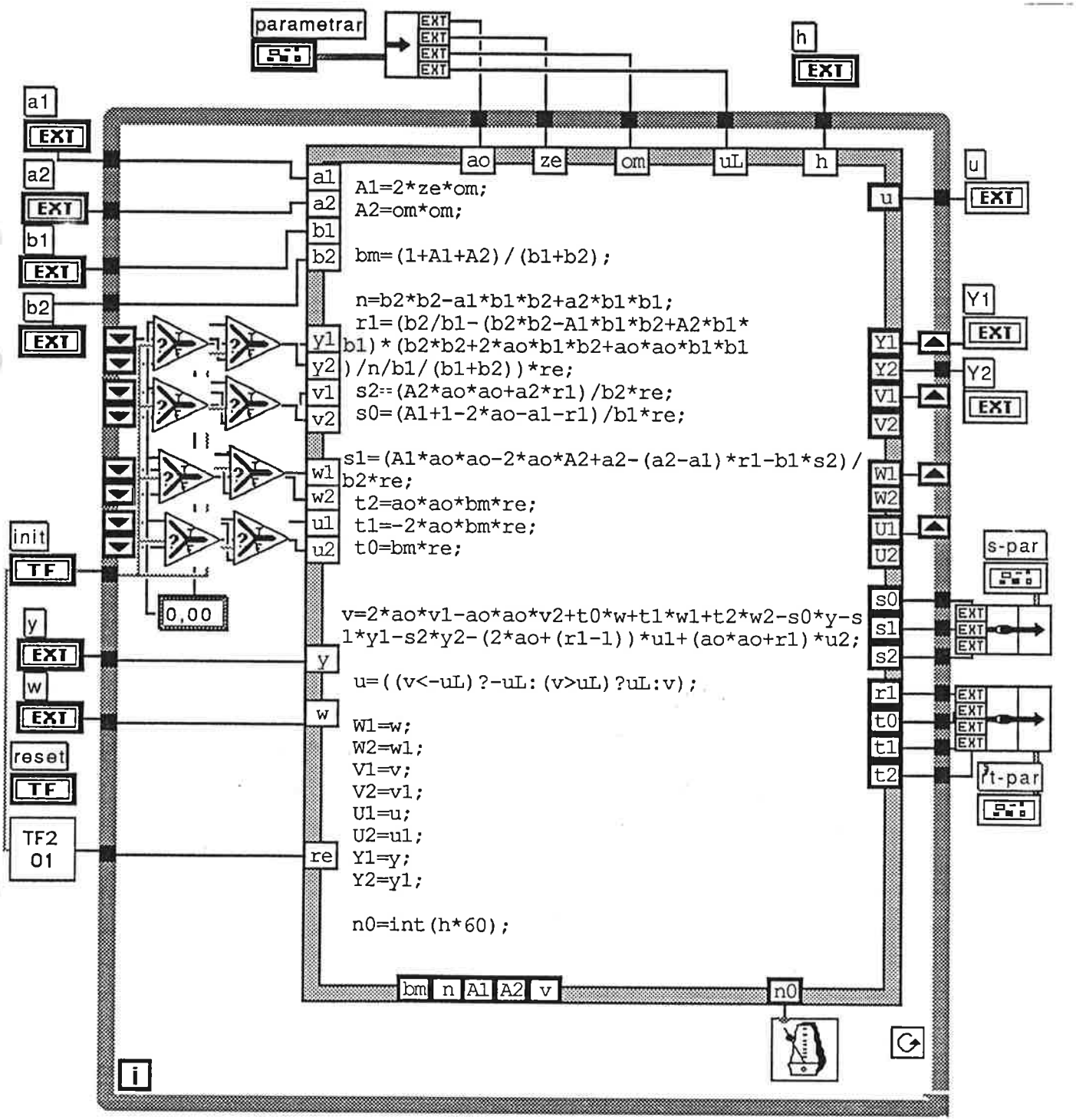
Manu används för att bestämma om man skall ha fyrkantsvåg eller manuellt inställbar signal som referenssignal.



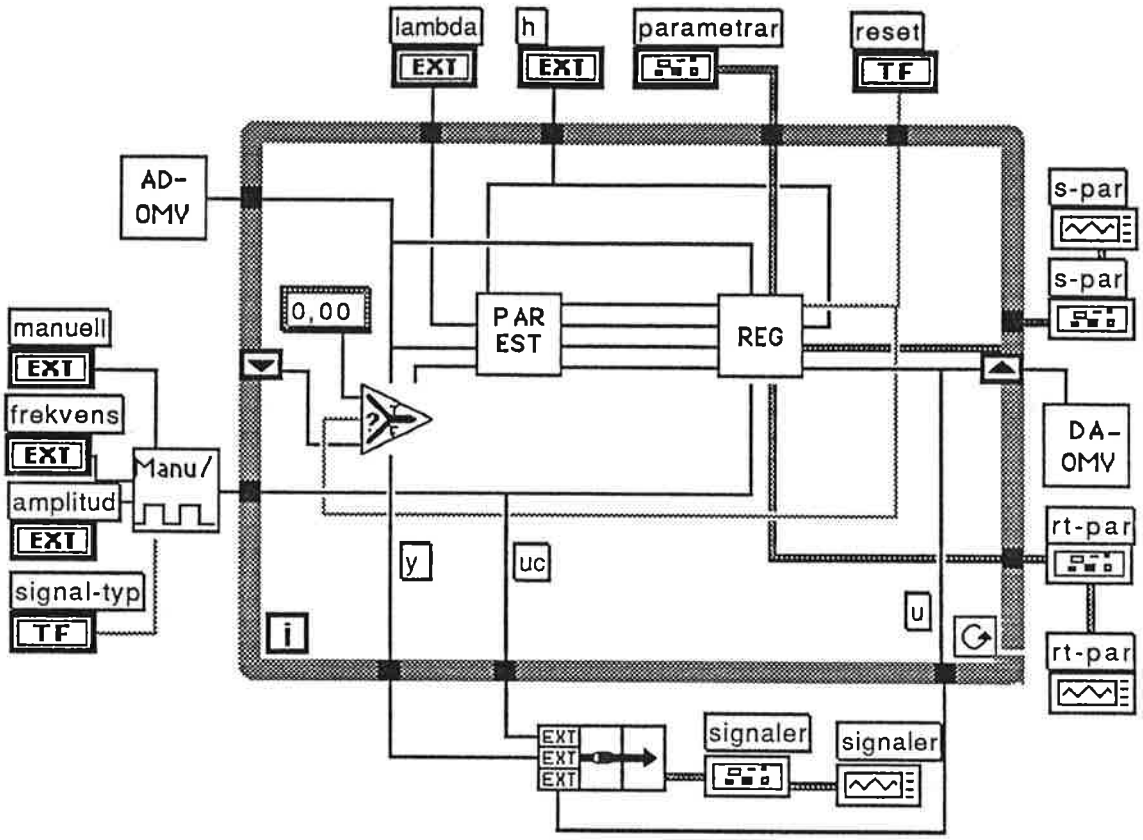
PAR EST estimerar processparametrarna a_1, a_2, b_1 och b_2. Som insignaler kommer y och u. Konstanterna h och lambda som kommer från användaren är också inparametrar.



REG löser den Diophantiska ekvationen och ger som utsignal u och parametrarna s_0, s_1, s_2, t_0, t_1, t_2 och r_1. Insignaler är y och u_c och inparametrar är a_1, a_2, b_1 och b_2. Här stoppas även in konstanterna a_o, lambda, omega, ulimit och h som användaren bestämmer.



I STR kopplas alla bitar samman till en enhet. Val av olika parametrar kan göras från frontpanelen på STR-objektet.



Experiment

Jag har ännu inte kunnat köra min regulator mot servot p.g.a. att jag inte fått delarna att fungera tillsammans. Problemet ligger i ihopkopplingen av parameterestimatorn och regulatorn. Regulatorn uppför sig instabilt och jag tror att det antagligen beror på att parameterestimeraren inte fungerar som den ska. Jag har heller inte några församplingsfilter, vilket också kan påverka.

Slutsatser

LabVIEW är i många stycken ett bra och roligt program att arbeta med. Man kan bygga upp mycket snygga och avancerade system och få dem att se ut precis som riktiga instrument. Det finns en uppsjö av olika funktioner färdiga att användas och man kan ändra och flytta omkring i diagrammet eller på panelen på ett enkelt sätt. Det är en fördel om man är van att jobba med Macintosh eftersom en hel del funktioner används som i andra Macintosh-program.

Det finns dock också en del negativa sidor. Man kan bara sampla med 1/60 sekund som snabbast. När man binder samman objekt med tråddragningsmekanismen får man ibland problem med lösa trådändar och konstiga kontakter mellan tråd och anslutning. Ett bra hjälpmedel för att hitta sådana fel är att använda Remove-Bad-Wires-funktionen i redigera-menyn. Manualen är inte heller alltid så tydlig och lite fler exempel skulle inte skada tycker jag. Jag hade t.ex. stora problem med initiering av skiftregister som står mycket kortfattat i manualen.

Stabilisering av inverterad pendel

Fredrik Hammar E87
Karl Henrik Johansson Spec
Kristofer Ljunggren E87
Johan Nilsson E87

HANDLEDARE:

Anders Nilsson
Karl Johan Åström

19 maj 1992

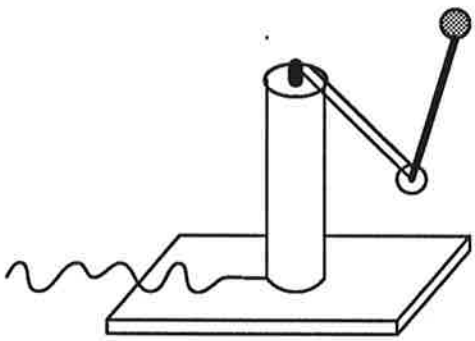
Innehåll

1 Inledning

Detta är vår redovisning av ett projektarbete i kurserna Datorimplementering av reglersystem och Adaptiv reglering som ges av Institutionen för Reglerteknik vid Lunds Tekniska Högskola.

Projektuppgiften har bestått i att balansera en inverterad pendel. Pendeln har varit monterad på en svängarm enligt figur 1. Svängarmen vrids runt av en motor så att pendelns pivotpunkt kommer att göra en cirkulär bana. Genom att alternera riktningen hos motorns vridmoment, går det att stabilisera pendeln i upprätt läge.

Stabiliseringen av den inverterade pendeln skulle ske med olineär tillståndsåterkoppling, där tillstånden var skattade med ett Kalmanfilter med varierande förstärkning, d v s en typ av parameterstyrning. Vi skulle jämföra denna regulator med en som bygger på lineär återkoppling. Undersökningar av våra regulatorer skulle utföras både vid simuleringar av processen i Simnon och på en verklig pendel.



Figur 1: Inverterad pendel med pendeln stående i upprätt läge.

2 Matematisk härledning

Om vi inför vinkeln φ, punktmassan m och pendelns längd l enligt figur 2, så erhålles följande differentialekvation för den inverterade pendeln

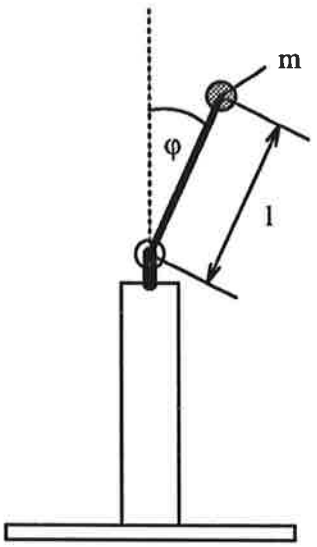
$$ml^2\frac{d\varphi}{dt} = mgl\sin\varphi + uml\cos\varphi$$

där pivotpunktens rätvinkliga acceleration u ingår i den sista termen. Vi sätter nu tillståndet $\omega = \frac{d\varphi}{dt}$ och definierar konstanterna a och b genom

$$a^2 = \frac{g}{l} \qquad\qquad b^2 = \frac{1}{l}$$

Vårt olineära system kan då skrivas på formen

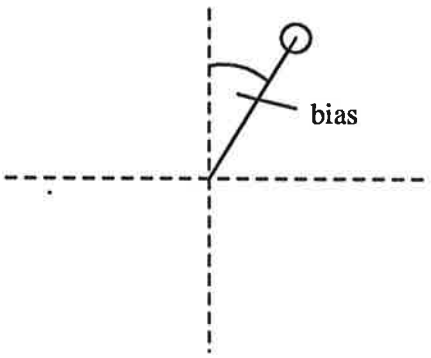
$$\begin{aligned} \frac{d\varphi}{dt} &= \omega \\ \frac{d\omega}{dt} &= a^2\sin\varphi + ub^2\cos\varphi \end{aligned} \tag{1}$$



Figur 2: Utslagsvinkeln φ, massan m och pendelns längd l definierad för den inverterade pendeln.

3 Kalmanfilter

Då bara pendelarmens vinkelläge är mätbart måste pendelns andra tillstånd estimeras, t ex med ett Kalmanfilter. Linjäriseras pendelns tillståndsekvationer får den matematiska modellen två tillstånd: vinkelläge φ och vinkelhastighet ω. För att pendeln skall kunna hållas stilla i stabiliserat läge krävs att pendelns vinkel kan mätas exakt. Då detta inte kan göras, t ex är inte vinkelgivaren monterad med vinkelläget noll uppåt, så estimerar vi även den bias ϵ mätningarna är behäftade med, se figur 3.



Figur 3: Pendelns läge då vinkelgivaren ger utsignalen 0. Mätsignalen från vinkelgivaren innehåller alltså en biasterm ϵ.

Vi inför tillståndsvektorn $x = \begin{pmatrix} \varphi & \omega & \epsilon \end{pmatrix}^T$ och linjäriserar processen kring φ_0 och u_0. Då

fås systemet

$$\begin{aligned} \frac{dx}{dt} &= Ax + Bu = \begin{pmatrix} 0 & 1 & 0 \\ c & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ d \\ 0 \end{pmatrix} u \\ y &= Cx = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix} x \end{aligned} \tag{2}$$

där $c = a^2 \cos\varphi_0 - u_0 b^2 \sin\varphi_0$ och $d = b^2 \sin\varphi_0$.

För att tillstånden skall kunna estimeras krävs att systemet är observerbart. Observarbarhetsmatrisen för vårt system blir

$$W_0 = \begin{pmatrix} C \\ CA \\ CA^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ c & 0 & 0 \end{pmatrix}$$

Eftersom denna matris har full rang är tillstånden observerbara. Ett rimligt Kalmanfilter har nu strukturen

$$\frac{d}{dt} \begin{pmatrix} \hat{\varphi} \\ \hat{\omega} \\ \hat{\epsilon} \end{pmatrix} = \begin{pmatrix} \hat{\omega} \\ a^2 \sin(\hat{\varphi}) + b^2 \cos(\hat{\varphi}) u \\ 0 \end{pmatrix} + K(y - \hat{\varphi} - \hat{\epsilon}) \tag{3}$$

där Kalmanförstärkningen

$$K = \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix}$$

Om vi betecknar skattningen av x med $\hat{x}$, så får vi vid små vinkelutslag Kalmanfiltret

$$\frac{d\hat{x}}{dt} = A\hat{x} + Bu + K(y - C\hat{x})$$

där A, B och C införts enligt ekvation (3). Filtrets dynamiska egenskaper bestäms alltså av matrisen $(A - KC)$ och filtrets karakteristiska polynom följdaktligen av

$$\det(sI - (A - KC)) = s^3 + (k_1 + k_3)s^2 + (k_2 - c)s - k_3 c$$

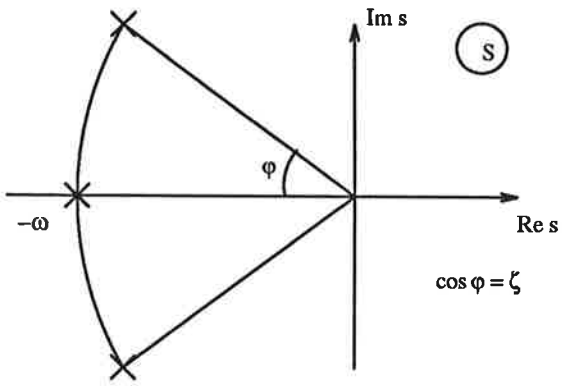
Polynomets nollställen har vi valt att placera på en cirkelbåge med en radie lika med egenvinkelfrekvensen ω. Om vi vidare inför dämpningen ζ, erhålles en polkonfiguration för Kalmanfiltret enligt figur 4. Det önskade karakteristiska polynomet får då formen

$$s^3 + (2\zeta + 1)\omega s^2 + (2\zeta + 1)\omega^2 s + \omega^3$$

Identifiering av koefficienterna ger

$$\begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} = \begin{pmatrix} (2\zeta + 1)\omega + \frac{\omega^3}{c} \\ (2\zeta + 1)\omega^2 + c \\ -\frac{\omega^3}{c} \end{pmatrix}$$

där c ges av den tidigare definitionen med aktuella värden på φ_0 och u_0. På detta sätt fås ett parameterstyrt kalmanfilter med förstärkning $K = K(\varphi, u)$.



Figur 4: Nollställen till Kalmanfiltrets önskade karakteristiska ekvation markerade med kryss.

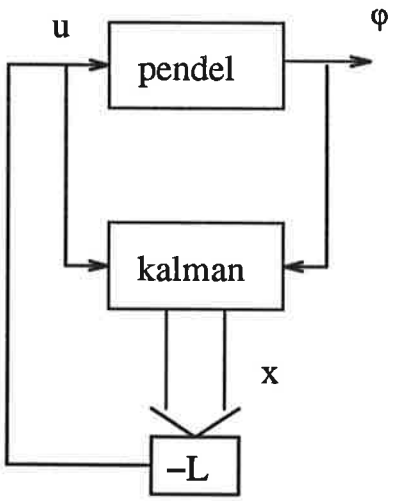
4 Regulatorer

Vid regleringen av vår pendelprocess utnyttjar vi tillstånden som skattas i Kalmanfiltret. Nedan härleder vi två regulatorer som bygger på lineär respektive olineär tillståndsåterkoppling. Båda härledningarna bygger på vårt olineära system

$$\begin{aligned} \frac{d\varphi}{dt} &= \omega \\ \frac{d\omega}{dt} &= a^2 \sin\varphi + ub^2 \cos\varphi \end{aligned} \tag{4}$$

För att få enklare beteckningar i fortsättningen inför vi tillståndsvektorn $x = (\varphi \quad \omega)^T$.

Lineär tillståndsåterkoppling



Figur 5: Systemet då de av Kalmanfiltret skattade tillstånden återkopplas lineärt.

När lineär tillståndsåterkoppling används, får vårt system en struktur enligt figur 5. Den lineariserade processen kring $\varphi_0 = 0$ är enligt tidigare

$$\frac{dx}{dt} = \begin{pmatrix} 0 & 1 \\ a^2 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ b^2 \end{pmatrix} u$$

Om tillståndsåterkopplingen $u = -Lx$ används med $L = (\,l_1 \quad l_2\,)$, erhåller vi för det slutna systemet det karakteristiska polynomet

$$p(s) = s^2 + l_2 b^2 s + l_1 b^2 - a^2$$

Alltså uppfyller systemet den önskade karakteristiska ekvationen

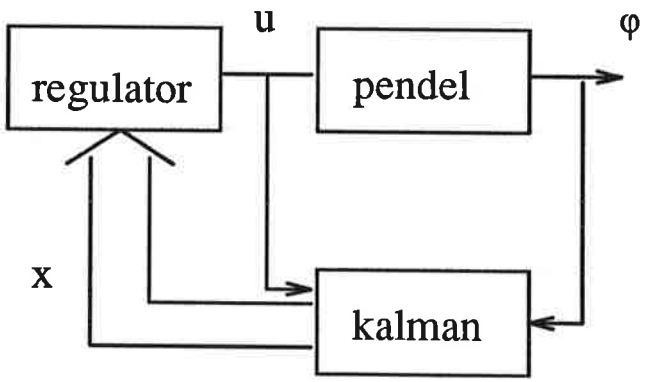
$$s^2 + 2\zeta\omega_0 s + \omega_0^2 = 0$$

om

$$L = \frac{1}{b^2}\,(\,2\zeta\omega_0 \quad \omega_0^2 + a^2\,)$$

Detta L ger alltså den lineära regulatorn.

Olineär tillståndsåterkoppling



Figur 6: Struktur hos vårt system då de av Kalmanfiltret skattade tillstånden återkopplas olineärt.

Vi kan illustrera vårt olineärt återkopplade system med figur 6, där regulatorn består av en olineär funktion som ger styrsignalen u ur tillståndet x.

Införs tillfälligt styrsignalen

$$u' = a^2 \sin\varphi + ub^2 \cos\varphi \tag{5}$$

så får vi ur ekvationerna (5) det lineära systemet

$$\frac{dx}{dt} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u'$$

$$y = (1 \quad 0)\,x$$

Om detta återkopplas med lineär tillståndsåterkoppling $u' = -L'x$, så att det slutna systemet får det karakteristiska polynomet

$$s^2 + 2\zeta\omega_0 + \omega_0^2$$

erhålles

$$L' = \begin{pmatrix} \omega_0^2 & 2\zeta\omega_0 \end{pmatrix}$$

När vi nu slutligen löser ut u ur ekvation (5), får vi vår verkliga styrsignal

$$u = \frac{u' - a^2 \sin\varphi}{b^2 \cos\varphi} = -\frac{\omega_0^2\varphi + 2\zeta\omega_0\omega + a^2 \sin\varphi}{b^2 \cos\varphi} \tag{6}$$

5 Tidsdiskretisering

När regleralgoritmerna skall implementeras i en dator för att styra den verkliga processen, måste observerare och regulator diskretiseras. För att göra detta på ett enkelt sätt använde vi Eulers framåtapproximation.

Diskretiseras det kontinuerliga Kalmanfiltret fås

$$\begin{aligned} \hat{\varphi}(kh+h) &= \hat{\varphi}(kh) + h(\hat{\omega} + k_1\Delta(kh)) \\ \hat{\omega}(kh+h) &= \hat{\omega}(kh) + h(a^2 sin\hat{\varphi}(kh) + ub^2 \cos\hat{\varphi}(kh) + k_2\Delta(kh)) \\ \hat{\epsilon}(kh+h) &= \hat{\epsilon}(kh) + hk_3\Delta(kh) \end{aligned} \tag{7}$$

där

$$\Delta(kh) = y(kh) - \hat{\varphi}(kh) - \hat{\epsilon}(kh)$$

På samma sätt diskretiserar vi den lineära tillståndsåterkopplingen, så att en regulator given av

$$u(kh) = -l_1\hat{\varphi}(kh) - l_2\hat{\omega}(kh) \tag{8}$$

erhålles. Den olineära återkopplingen blir slutligen

$$u(kh) = -\frac{\omega_0^2\hat{\varphi}(kh) + 2\zeta\omega_0\hat{\omega}(kh) + a^2 \sin\hat{\varphi}(kh)}{b^2 \cos\hat{\varphi}(kh)} \tag{9}$$

6 Simuleringar

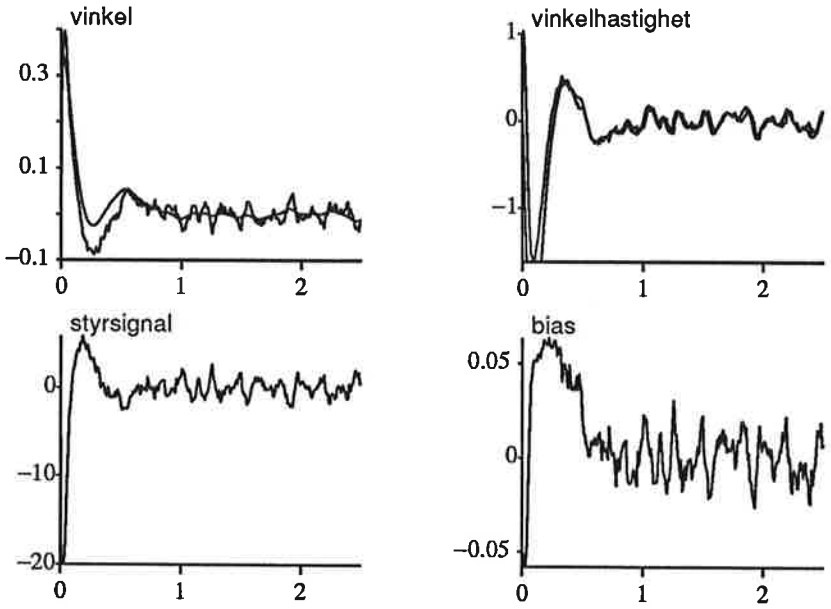
För att kontrollera våra beräkningar och för att försöka få en känsla för systemet gjordes simuleringar. För simuleringarna användes programpaketet Simnon, vari en modell av pendeln och regulatorerna implementerades. Simuleringarna koncentrerades till att jämföra egenskaper hos systemet när det reglerades med lineär respektive olineär tillståndsåterkoppling.

Längden på pendeln ingår som enda obestämda konstant i vår matematiska modell. Den sattes till 0.3 m. Mätbruset som vi tillför vårt system är genomgående normalfördelat med väntevärde 0 och standardavvikelse 0.005. Axlarna i de grafer i detta avsnitt som visar vinkel och bias är givna i radianer, de som visar vinkelhastighet i rad/s och de som visar styrsignalen i m/s^2.

Simulering av tidskontinuerligt system

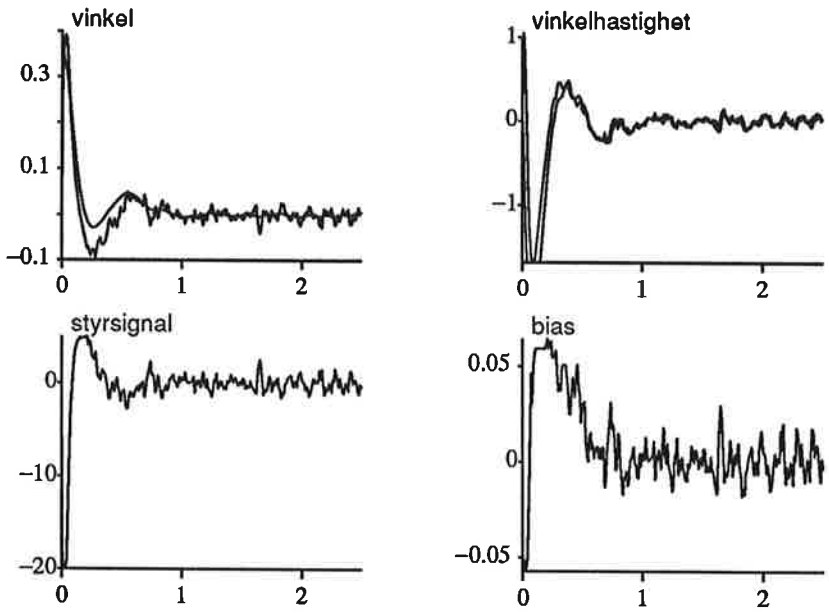
Inledningsvis använde vi ett tidskontinuerligt Kalmanfilter enligt ekvation (3) och en regulator med antingen lineär eller olineär tillståndsåterkoppling. Brus adderades till den variabel som var pendelns vinkel, för att den mer skulle efterlikna den verkliga mätsignalen. Polerna i observeraren placerades så att egenvinkelfrekvensen $\omega = 3a \approx 17$ och dämpningen $\zeta = 0.7$. Initialt var tillstånden i Kalmanfiltret nollställda och pendeln hade en lutning på 20 grader.

En simulering av systemet med **lineär tillståndsåterkoppling** $u = -Lx$ presenteras i figur 7. Polerna för det slutna systemet ges av $\omega = 2a \approx 11$ och $\zeta = 0.7$. Den kurvan i både vinkel- och vinkelhastighetgrafen som är slät representerar de verkliga signalerna, d v s vinkel- och vinkelhastighet utan mätbrus. Vi noterar att alla signalerna konvergerar mot noll (om vi bortser från brus) på ungefär en sekund. I styrsignalens transient ses hur den når sin begränsning; den är initialt -20 m/s^2.



Figur 7: Resultat vid simulering av systemet med lineär återkoppling. Alla signaler konvergerar efter någon sekunds transienter.

När vi gjorde en simulering motsvarande ovanstående med en regulator baserad på **olineär tillståndsåterkoppling** enligt ekvation (6), fick vi resultatet i figur 8. Polerna för det transformerade systemet placerades som i föregående simulering. Vi ser att beteendet är näst intill identiskt för systemet med olineär återkoppling gentemot det med lineär.



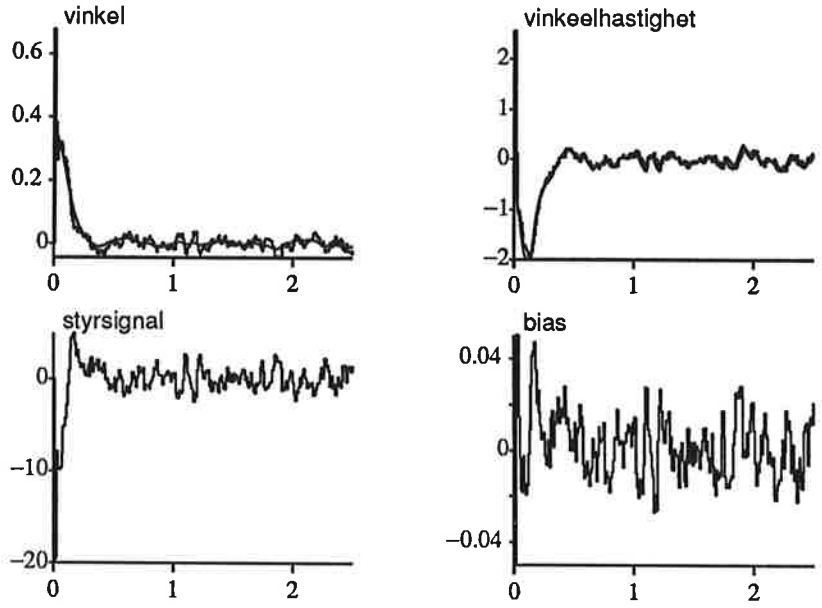
Figur 8: Resultat vid simulering av systemet med olineär återkoppling. Graferna överensstämmer näst intill identiskt med de i föregående figur.

Simulering av tidsdiskret system

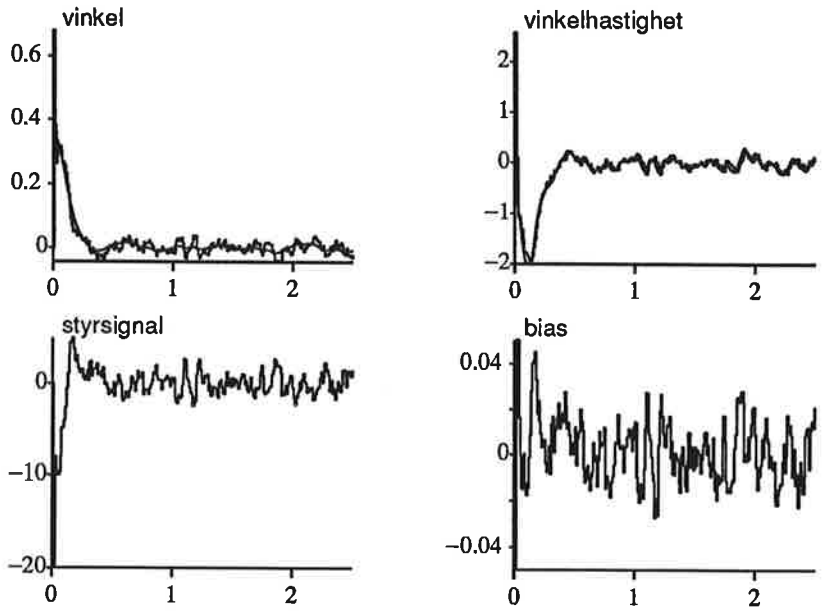
För att validera den approximation som tidsdiskretiseringen innebär, gjorde vi även simuleringar där regulatorer och Kalmanfilter var diskreta. Här överfördes de ekvationer som redovisas i avsnitt 5 till Simnon-program, i vilka vi lät sampelfrekvensen vara 100 Hz.

I figur 9 visar vi resultatet av en simulering av det diskreta systemet med **lineär tillståndsåterkoppling**, där polplacering och begränsningar är helt enligt tidigare (notera skalorna hos graferna). Överrensstämmelsen hos graferna är god med de för det tidskontinuerliga fallet. Av detta drar vi slutsatsen att en sampelfrekvens på 100 Hz är tillräcklig för att Eulerapproximationen ska vara bra.

Även vid simulering med **olineär tillståndsåterkoppling** blev resultaten vid kontinuerlig och diskret regulator snarlika. I figur 10 visas en simulering som tillsammans med figur 8 illustrerar detta.



Figur 9: Simulering av system med tidsdiskret lineär återkoppling. Kurvorna påminner mycket om det tidigare kontinuerliga fallet.

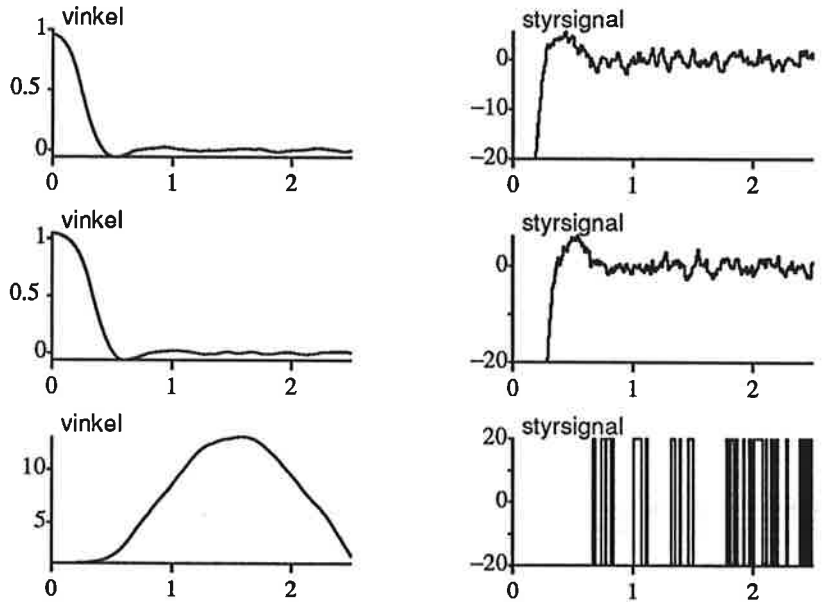


Figur 10: Simulering av system med tidsdiskret olineär återkoppling. Även här påminner kurvorna mycket om det tidigare kontinuerliga fallet.

Simulering av maximal inhämtningsvinkel

För att studera regulatorernas förmåga att klara av laststörningar gjordes simuleringar där pendeln startas i vinkelläge nära den teoretiska gränsen för inhämtning. Vid simuleringarna

antogs att den maximala accelerationen i upphängningspunkten var som tidigare 20 m/s^2, d v s ungefär $2g$. I figur 11 och 12 visas simuleringsresultat vid startvinkel 55, 60 och 65 grader för de båda regulatorerna.



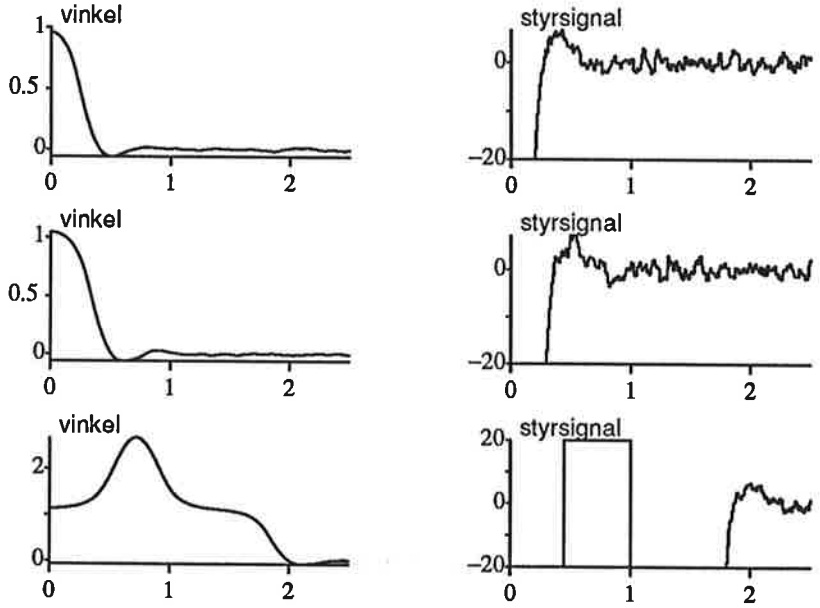
Figur 11: Simuleringsresultat med linear tillståndsåterkoppling, startvinkel 55, 60 resp 65 grader.

Vi ser att båda regulatorerna klarar av en startvinkel av 55 och 60 grader, men de klarar inte 65-gradersfallet. Som väntat kan noteras att styrsignalen initialt är bottnad en längre tid vid 60 grader än vid 55. För de startvinklar som klaras av uppför sig de båda regulatorerna mycket lika.

Sammanfattning av simuleringarna

Våra simuleringar bekräftar att regulatorerna borde klara av att stabilisera den verkliga pendeln i upprätt läge, om inte vår processmodell är alltför dålig. Vi sammanfattar resultaten av våra simuleringar i några punkter:

- Både den linearа och den olinearа regulatorn klarar av att stabilisera pendeln vid en sampelfrekvens av 100 Hz.
- Det är väldigt lika uppförande hos systemet, oberoende av vilken av de båda regulatorerna som det regleras av.
- Vid en maximal styrsignal på 20 m/s^2 är den största vinkel från upprätt läge de båda regulatorerna klarar att hämta in 60 grader.



Figur 12: Simuleringsresultat med olineär tillståndsåterkoppling, startvinkel 55, 60 resp 65 grader.

7 Implementering

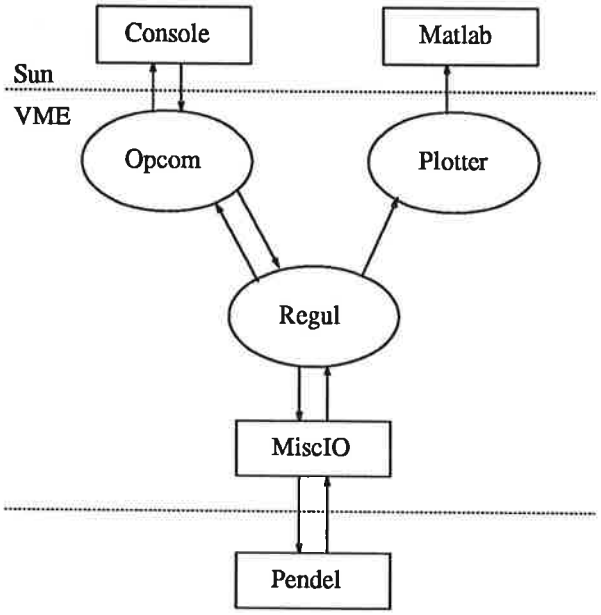
Vi har implementerat vår regulator i en VME-baserad styrdator som är uppbyggd kring Motorolas processor 68020. Vårt datorprogram är skrivet i Modula-2 och kompilerat på och nedladdat från en SUN arbetsstation. Den använda realtidskärnan har skrivits vid institutionen och länkats ihop med vår programkod. Modula-2 kompilatorn kommer från Oregon Software.

För att operatörskommunikation, reglering och kurvplottning skall kunna ske samtidigt delas programmet upp i tre parallellt exekverade realtidsprocesser enligt figur 13. Processen `Opcom` sköter kommunikationen med användaren, `Plotter` presenterar via ett Matlab-fönster information om processen och `Regul` utför uppdatering av Kalmanfilter, regulatorer och styrsignal.

Operatörskommunikation

Styrning av regulatorn och övervakning av processen sker via fönster på en eller flera arbetsstationer. Kommunikationen arbetsstation–styrdator kan ske antingen via Ethernet eller Appletalk.

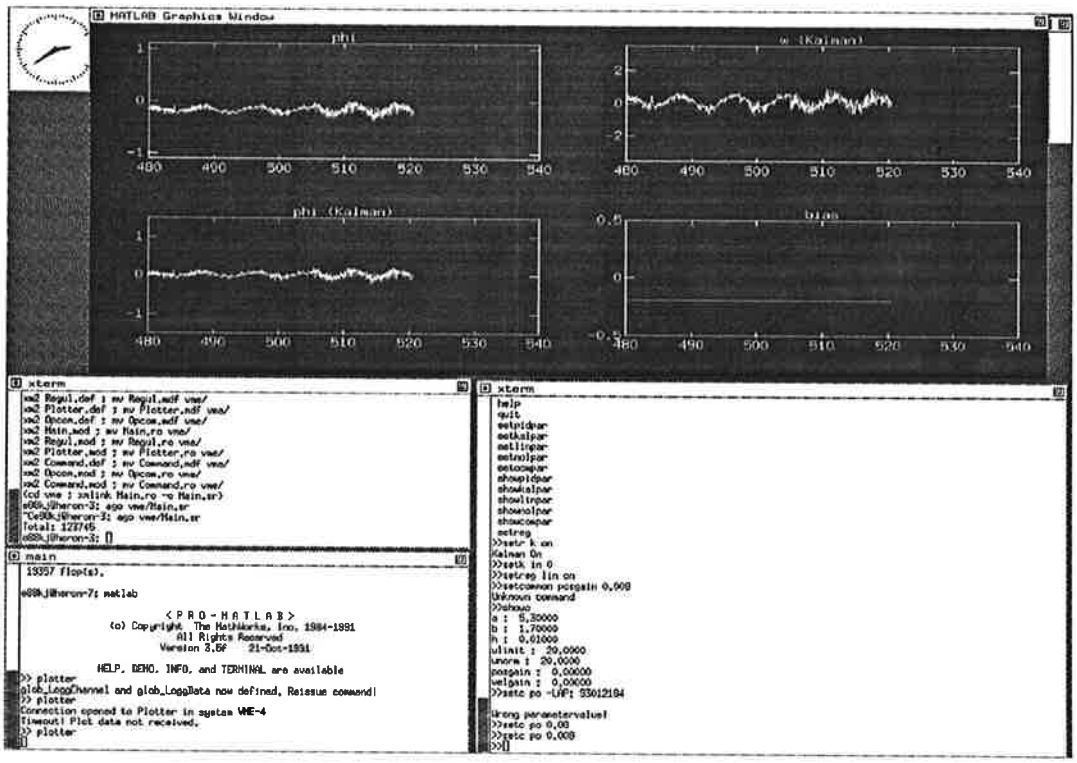
I ett fönster som symboliserar terminal har operatören kontakt med systemet via processen `Opcom`. Här kan en av tre olika regulatorer väljas och olika parametrar till dessa ställas in. Vidare har operatören från terminalen möjlighet att sätta begränsningar hos t ex styrsignalen eller ändra konstanter i systemet. Nedan visas en tabell över implementerade kommando tillsammans med en kort förklaring.



Figur 13: Processgraf

Kommando	*Förklaring*
help	Listar alla kommandon
quit	Avsluta körning
setcompar	Ställer allmänna parametrar
setkalpar	Ställer parametrar i Kalmanfilter
setlinpar	Ställer parametrar i lineär regulator
setnolpar	Ställer parametrar i icke-lin. regulator
setpidpar	Ställer parametrar i PID-regulator
setreg	Ställer typ av regulator
showcompar	Visar aktuella allmänna parametervärden
showkalpar	Visar aktuella parametervärden i Kalmanfilter
showlinpar	Visar aktuella parametervärden i lin. regulator
shownolpar	Visar aktuella parametervärden i icke-lin. reg.
showpidpar	Visar aktuella parametervärden i PID-reg.

Övervakning av processen sker via ett eller flera Matlab-fönster. Varje fönster har kontakt med processen `Plotter` genom att det i fönstret körs olika Matlab-program. Tack vare de här programmen kan val ske om vilka signaler som ska plottas. Intressanta signaler att se kan vara pendelns utslagsvinkel, styrsignalen till motorn och tillstånden i Kalmanfiltret. Det finns även möjlighet att från Matlab välja hur ofta en graf ska uppdateras. På det sättet är det möjligt att bibehålla en kontinuerlig plottning, trots att styrdatorns processor jobbar hårt med regleringen. I figur 14 visar vi operatörens datorskärm under en körning.



Figur 14: Operatörens kommunikation med processen. Överst visas grafer över några signaler från processen och därunder till höger terminalfönstret.

Regulator

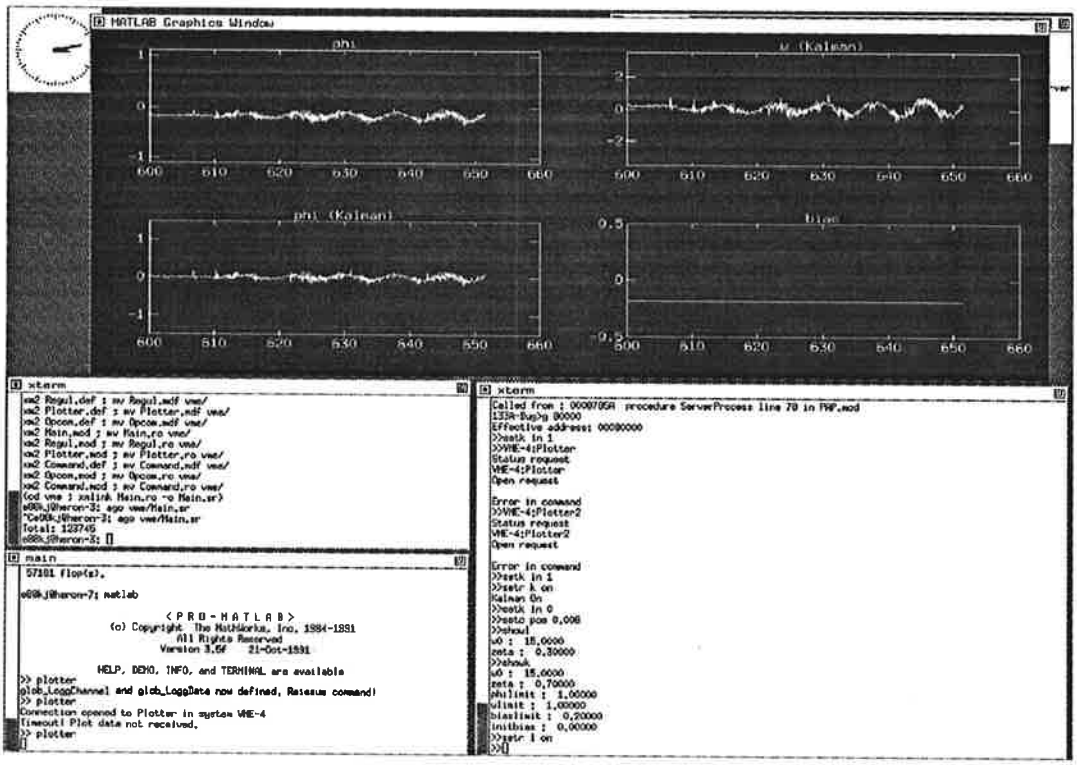
Med regulatorprocessen `Regul` finns det möjlighet att välja mellan en PID-regulator, en regulator med lineär och en med olineär tillståndsåterkoppling. Processen sköter om uppdatering av tillstånden i PID-regulatorn och i det Kalmanfilter som krävs för tillståndsåterkopplingen. För att inte reglering skall äventyras har `Regul` fått högre prioritet än både processerna `Opcom` och `Plotter`.

I `Regul` har de diskreta algoritmer implementerats som har diskuterats i ett tidigare avsnitt. För att regleringen av processen ska fungera väl har det visat sig att algoritmerna måste exekveras med en sampelfrekvens på 100 Hz. Detta visade sig också vara en gräns för hur fort styrdatorn klarade av att exekvera huvudloopen i regulatorn.

8 Reglering av verklig pendel

Omodellerad dynamik hos den verkliga pendel har gjort att vi har varit tvungna att ändra lite på vår styrstratergi. I stort sett har det dock visat sig att den verkliga pendeln uppför sig på ett sätt som motsvarar det som vi hade förväntat oss.

Eftersom vi hade som mål att stabilisera pendeln i upprätt läge, har vi valt att starta med den där. Då vi inledningsvis startade Kalmanfiltret och regulatorn samtidigt, visade det sig att



Figur 15: En reglering av pendeln med linear återkoppling av tillstånden i Kalmanfiltret och positionen hos motoraxeln. I figuren på nästa sida finns fler grafer.

den skattade vinkelbiasen $\hat{e}$ inte konvergerade mot ett stationärt värde. Därför ändrade vi i vår styrstratergi och lät biasen nå ett stationärt värde som låstes innan regleringen startades.

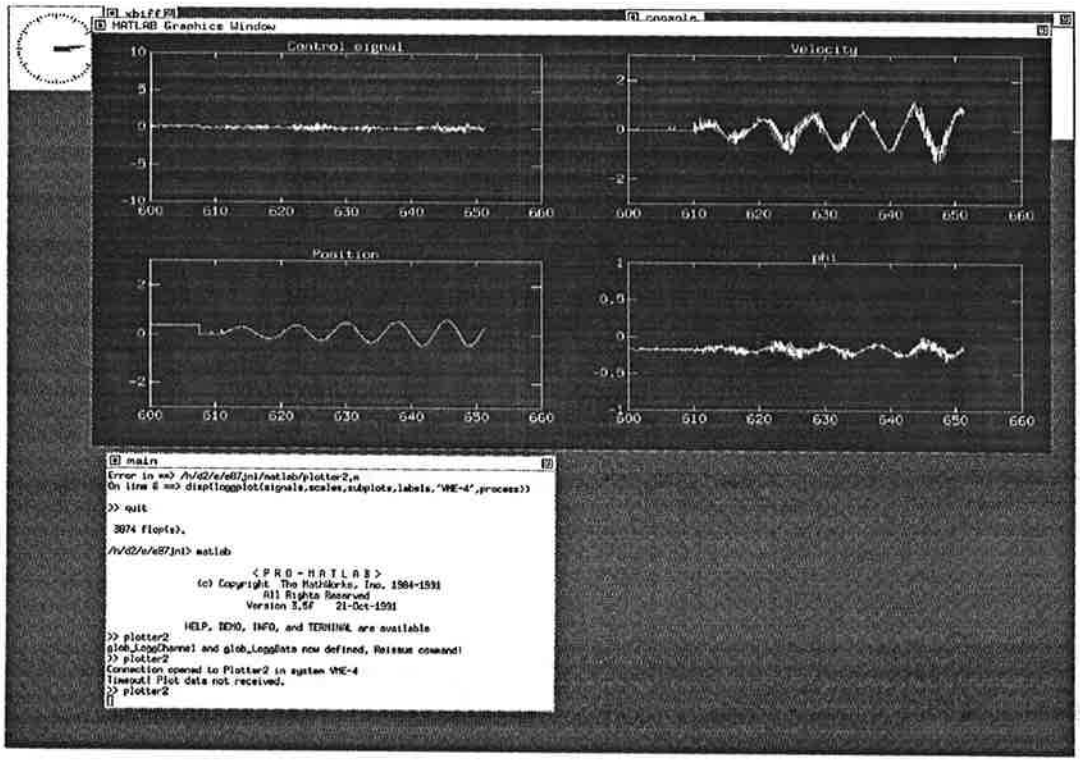
Polerna hos Kalmanfiltret visade sig kunna placeras som vid simuleringarna, emedan det återkopplade systemet fick göras något snabbare. Bra värden har varit $\omega \approx 13 - 15$ och $\zeta \approx 0.5 - 0.7$. Observerare och system valdes ungefär lika snabba.

Vi har lyckats stabilisera den verkliga pendeln med alla tre regulatorerna. Uppförandet hos systemet reglerat med respektive regulator är dock starkt beroende på hur regulatorernas parametrar väljs. Därför har ett tidsödande arbete varit att ställa in dessa så bra som möjligt. Vi tror oss inte ha nått några optimala parameterval, men tycker oss känna systemet så mycket att vi kan säga att regleringar med den lineära och olineära regulatorn är likvärdiga. Däremot överträffar de PID-regleringar.

Ett problem som uppstod vid alla regleringarna var att, trots att pendeln stod stabilt i upprätt läge så accelererade motoraxeln sakta. Det här beteendet går teoretiskt inte att bli av med utan att mäta positionen hos axeln, vilket beror på att det enda vi vet om axeln är indirekt dess acceleration (utsignalen u till motorn). Eftersom sedan utsignalen inte exakt motsvarar accelerationen, kommer en beräknad axelposition att innehålla ett allt större fel. När axelns position mättes och återkopplades upphörde den långsamma accelerationen hos motoraxeln.

I figur 15 och 16 visas en reglering av pendeln med lineär återkoppling av tillstånden i Kalmanfiltret och positionen hos motoraxeln. Bland graferna i det övre fönstret i figur 15 ser vi

bl a hur väl skattade vinkeln överensstämmer med den mätta och att biasen ligger konstant. I det grafiska fönstret i figur 16 kan noteras hur liten styrsignalen är och den sinusformade svängningen runt noll hos motoraxeln position. Just vid denna reglering verkar amplituden hos svängningen att divergera.



Figur 16: Ytterligare grafer till den reglering som illustreras i figuren på föregående sida.

9 Slutsatser

För vårt projekt har vi haft ett klart och väl definierat mål: Att stabilisera den inverterade pendeln. Detta har vi lyckats med! På vår väg fram mot målet har vi dock stött på del problem, av vilka vi har löst vissa och gått runt andra. Nedan sammanfattas de delresultat som vi har kommit fram till.

- För den olineära matematiska modell som vi har använt för den inverterade pendeln går det enkelt att göra ett parameterstyrt Kalmanfilter och en regulator med olineär tillståndsåterkoppling.
- Simuleringarna visar att med rimlig begränsning på styrsignalen så klarar både den lineära och den olineära regulatorn att stabilisera pendeln.
- Enligt simuleringarna är det allmäna uppförandet hos systemet näst intill identiskt med de båda regulatorerna.

- Det verkliga systemet är känsligt för parametervariationer, t ex är det viktigt hur polerna placeras för Kalmanfiltret och det slutna systemet.
- Även för den verkliga pendeln märks ingen skillnad om den regleras med regulatorn som bygger på lineär eller olineär återkoppling.

Styrning av inverterad pendel

Anders Kristenson D-88
Magnus Wiklund E-88

Handledare:
Karl Johan Åström
Anders Nilsson

7 maj 1992

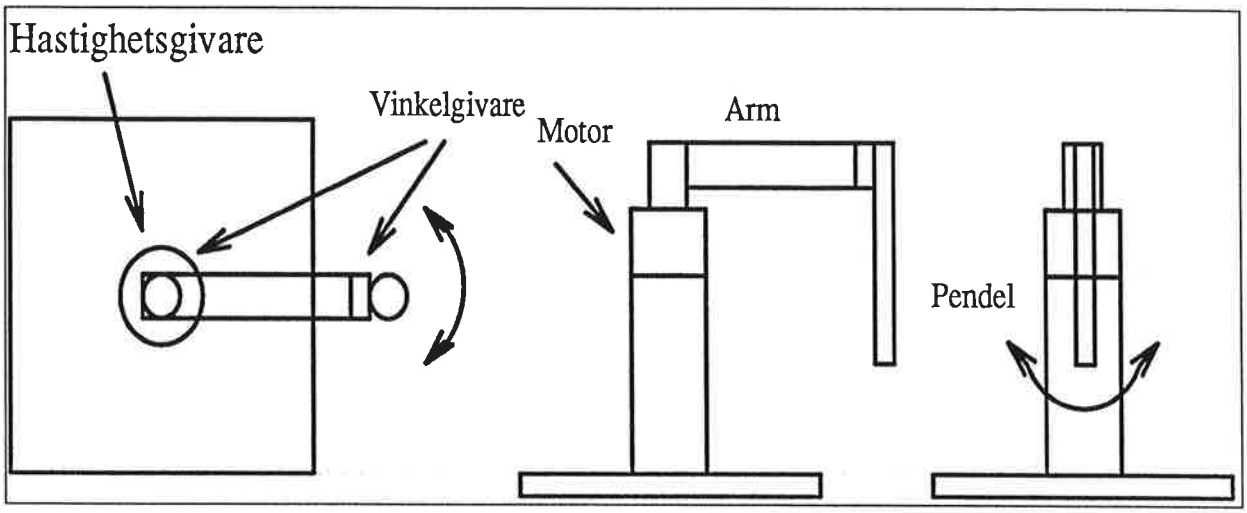
Sammanfattning

Denna rapport är ett resultat av projekt som utförts under kurserna Datorimplementering av regelersystem och Adaptiv reglering, som ges vid institutionen för regelteknik vid Lunds tekniska högskola. Målet med uppgiften var att med en dator styra den inverterade pendel som byggts vid institutionen. För detta ändamål har ett olinjärt kalmanfilter använts för att skatta pendelns tillstånd. För reglering har använts tillståndsåterkoppling. Resultatet har blivit en regulator som både kan kasta upp pendeln och reglera den i övre läget. Denna regulator har efter simulering implementerats i Modula-2. Det visade sig ej vara möjligt att balansera pendeln i övre läget baserat endast på pendelns tillstånd, utan även armens tillstånd var nödvändiga att använda för balansering.

Innehåll

1 Inledning

Vår uppgift är att med kalmanfilter följa pendelns tillstånd, och kunna balansera den i upprätt (inverterat) läge.



Figur 1: Principskiss av den inverterade pendeln,på bilden i stabilt nedre läge.

Härvid uppstår två problem nämligen

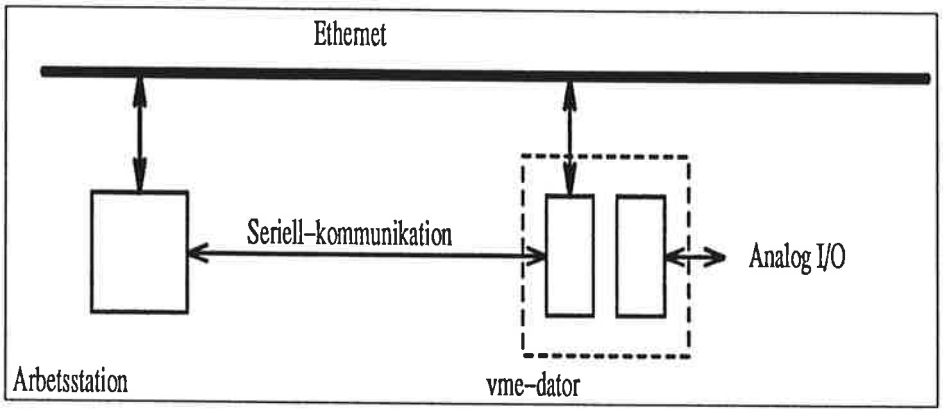
1. Uppkastning av armen från det stabila nedre läget till det instabila övre läget.
2. Balansera pendeln i det övre läget.

Ytterligare problem får man på grund av att lägesgivaren mättar i sitt omslagsområde, och att lägesgivarens nollpunkten inte motsvarar någon lämplig nollpunkt för den matematiska nollpunkten.

Det vi vill åstakomma är en regulator som "själv" inser när arman skall kastas upp, regleras i upprätt läge eller när störningen är för stor för att vara möjlig att reglera. I första hand har vi försökt att endast använda oss av pendelns vinkelgivare. Vi har dock även behandlat det SIMO system som uppstår när även armens vinkel och hastighet mäts.

1.1 Materiel

För att styra pendeln i figur 1 användes en vme-kort baserad dator kopplad till arbetsstationsnätet. För programmeringen användes modula-2 som korskompilerades på arbetsstationen och laddades ned för exekvering på vme-datorn, enligt figur 2.



Figur 2: Försöksuppkopplingen.

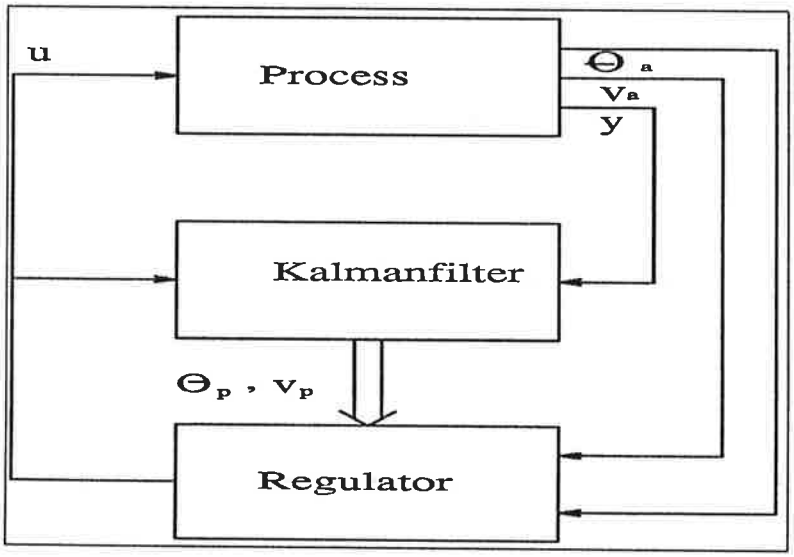
Mot den seriella porten fungerar arbetsstationen endast som en terminal. Detta för att erhålla operatörskommunikationen med vme-datorn. Via Ethernet laddas sedan programmet ned. Det är även via Ethernet som variabler för plottning skickas tillbaka till arbetsstationen från vme-datorn.

1.2 Rapportens indelning

Rapportens indeling följer i grova drag den arbetsgång som vi haft. I avsnittet *Teoretisk analys 2* behandlas den matematiska pendel samt den teoretiska delen av vår lösning på problemet. I avsnittet *Simulering av systemet 3* redogörs för simuleringsresultaten. I avsnittet *Realisering 4* behandlas realiseringen av den i de föregående kapitlen framtagna lösningen. I avnittet *Resultat 5* behandlas de resultat som erhölls vid praktiska prov med programmet samt de slutsatser som framkommit under arbetets gång.

2 Teoretisk analys

I detta kapitel behandlas den teoretiska delen av projektarbetet. Detta innefattar val av modellstruktur, val av regulatortyp och val av kalmanfilter, läsaren bör notera att valet av modellstruktur för pendeln styr valet av kalmanfilter. I avsnitten 2.1 – 2.5 behandlas SISO fallet, det vill säga att endast y i figur 3 används. I avsnitten 2.6 – 2.7 behandlas SIMO fallet, det vill säga när även θ_a och v_a används, dock används fortfarande samma kalmanfilter för att skatta $\hat{\theta}_p$ och $\hat{v}_p$ från y. I avsnittet 2.8 utförs en stabilitetsanalys med hjälp av en Lyapunov funktion. Hur mycket energi som behövs vid uppkast, och vilken strategi som är lämpligast, behandlas i avsnittet 2.9. I figur 3 framgår blockstrukturen hos vårat system.



Figur 3: Systemets uppbyggnad i vår realisering.

2.1 Matematisk modell av pendeln

I vår modell av pendeln antar vi att vinkeln θ_p utgår från pendeln i upprätt läge, och att insignalen u är proportionell mot pivotpunktens vridmoment för pendeln med en faktor α. Om man antar att den cirkulära rörelsen som pivot punkten rör sig kan approximeras med en rät linje så beskrivs pendeln av

$$J\ddot{\theta}_p = m_p g l_p \sin(\theta_p) + \alpha u \cos(\theta_p) \tag{1}$$

där tröghetsmomentet $J = ml_p^2$ vilket förenklat ger

$$\ddot{\theta}_p = \frac{g}{l_p}\sin(\theta_p) + \frac{\alpha u}{ml_p^2}\cos(\theta_p)$$

På grund av att nollstället för vinkelgivaren inte är rakt upp som tidigare antagits så blir den mätta signalen en kombination av en bias b_p och vinkeln θ_p. Vi väljer därför att införa två nya tillstånd, b_p och v_p, för att få över modellen på tillståndsform och med hänsyn tagen till biasen.

$$\frac{d\theta_p}{dt} = \sqrt{\frac{g}{l_p}} v_p$$

vilket ger systemet

$$\begin{cases} \frac{d\theta_p}{dt} = \sqrt{\frac{g}{l_p}} v \\ \frac{dv_p}{dt} = \sqrt{\frac{g}{l_p}} \sin(\theta_p) + \frac{\sqrt{l_p}\alpha u}{m_p \sqrt{g} l_p^2} \cos(\theta_p) \\ \frac{db_p}{dt} = 0 \\ y = \theta_p + b_p \end{cases} \tag{2}$$

linearisering och omskrivning på matrisform ger

$$x = \begin{pmatrix} \theta_p \\ v_p \\ b_p \end{pmatrix}$$

$$A = \nabla f(\theta_p, v_p, b_p, u)|_{x_0} = \left. \begin{pmatrix} 0 & \frac{\sqrt{g}}{\sqrt{l_p}} & 0 \\ \frac{\sqrt{g}\cos(\theta_p)}{\sqrt{l_p}} - \frac{\alpha u \sin(\theta_p)}{l_p^{3/2} m_p \sqrt{g}} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \right|_{(\theta_p=\theta_0, v_p=v_0, b_p=b_0, u=u_0)}$$

$$B = \left.\frac{df}{du}\right|_{x_0} = \left. \begin{pmatrix} 0 \\ \frac{\alpha \cos(\theta_p)}{l_p^{3/2} m_p \sqrt{g}} \\ 0 \end{pmatrix} \right|_{(\theta_p=\theta_0, v_p=v_0, b_p=b_0 u=u_0)}$$

$$C = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}$$

Vi ser att observerbarhetsmatrisen

$$W_o = \begin{pmatrix} C \\ CA \\ CA^2 \end{pmatrix} = \left. \begin{pmatrix} 1 & 0 & 1 \\ 0 & \frac{\sqrt{g}}{\sqrt{l_p}} & 0 \\ -\frac{\alpha u \sin(\theta_p) - g\cos(\theta_p) l_p m_p}{l_p^2 m_p} & 0 & 0 \end{pmatrix} \right|_{(\theta_p=\theta_0, v_p=v_0, b_p=b_0 u=u_0)}$$

har full rang , viket betyder att det är möjligt att skatta tillstånden θ_p,v_p och b_p.

2.2 Diskretisering av systemet

Att sampla detta system på traditionellt sätt ger

$$\Phi = e^{Ah} =$$

$$= \begin{pmatrix} e^{-\frac{h\beta}{l_p\sqrt{m_p}}}1/2 + e^{\frac{h\beta}{l_p\sqrt{m_p}}}1/2 & \sqrt{l_p}\sqrt{m_p}\left(e^{\frac{h\beta}{l_p\sqrt{m_p}}} - e^{-\frac{h\beta}{l_p\sqrt{m_p}}}\right)\sqrt{g}1/2\frac{1}{\beta} & 0 \\ -\left(e^{\frac{h\beta}{l_p\sqrt{m_p}}} - e^{-\frac{h\beta}{l_p\sqrt{m_p}}}\right)(\gamma)\frac{1}{2\beta\sqrt{l_p m_p g}} & e^{-\frac{h\beta}{l_p\sqrt{m_p}}}1/2 + e^{\frac{h\beta}{l_p\sqrt{m_p}}}1/2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

där

$$\beta = \sqrt{g\cos(\theta_p)l_p m_p - \alpha\, u\sin(\theta_p)}$$

$$\gamma = \alpha\, u\sin(\theta_p) - g\cos(\theta_p)l_p m_p$$

Detta är ett inte allt för trevligt sätt att arbeta med problemet varför det lämnas därhän.

2.3 Kalmanfilter i kontinuerlig tid

Ett kalmanfilter i kontinuerlig tid till systemet i ekvation 2 ges av

$$\begin{cases} \frac{d\hat{\theta}_p}{dt} = \sqrt{\frac{g}{l_p}}\hat{v}_p + k_1(y-\hat{y}) \\ \frac{d\hat{v}_p}{dt} = \sqrt{\frac{g}{l_p}}\sin(\hat{\theta}_p) + \frac{\sqrt{l_p}\alpha u}{m_p\sqrt{g}l_p^2}\cos(\hat{\theta}_p) + k_2(y-\hat{y}) \\ \frac{d\hat{b}_p}{dt} = k_3(y-\hat{y}) \end{cases} \tag{3}$$

Där

$$\hat{y} = C\begin{pmatrix}\hat{\theta}_p \\ \hat{v}_p \\ \hat{b}_p\end{pmatrix} = \begin{pmatrix}1 & 0 & 1\end{pmatrix}\begin{pmatrix}\hat{\theta}_p \\ \hat{v}_p \\ \hat{b}_p\end{pmatrix} = \hat{\theta}_p + \hat{b}_p$$

för att bestämma observerardynamiken används polynomet

$$det(sI - A + KC) = \begin{vmatrix} s+k_1 & -\frac{\sqrt{g}}{\sqrt{l_p}} & k_1 \\ \frac{\alpha\, u\sin(\theta_p)}{l_p^{3/2}m_p\sqrt{g}} - \frac{\sqrt{g}\cos(\theta_p)}{\sqrt{l_p}} + k_2 & s & k_2 \\ k_3 & 0 & s+k_3 \end{vmatrix} =$$

$$= s^3 + \frac{\left(l_p^2 m_p k_3 + k_1 m_p l_p^2\right)s^2}{l^2 m_p} + \frac{\left(\alpha\, u\sin(\hat{\theta}_p) - g\cos(\hat{\theta}_p)l_p m_p + k_2 l_p^{3/2}m_p\sqrt{g}\right)s}{l_p^2 m_p} + \frac{\alpha\, u\sin(\hat{\theta}_p)k_3 - g\cos(\hat{\theta}_p)l_p m_p k_3}{l^2 m_p}$$

Identifiering mot önskat polynom

$$(s+\alpha_k)(s^2 + 2\zeta_k\omega_k s + \omega_k^2)$$

ger att vektorn K skall väljas som

$$\begin{cases} k_1 = 2\zeta_k\omega_k + \alpha_k - \frac{\alpha_k\omega_k^2 l^2 m}{\alpha u\sin(\hat{\theta}_p) - l_p m_p g\cos(\hat{\theta}_p)} \\ k_2 = \sqrt{\frac{l_p}{g}}(\omega_k^2 + 2\alpha_k\zeta_k\omega_k + \frac{g}{l_p}\cos(\hat{\theta}_p) - \frac{\alpha u\sin(\hat{\theta}_p)}{l_p^2 m_p}) \\ k_3 = \frac{\alpha_k\omega_k^2 l^2 m}{\alpha u\sin(\hat{\theta}_p) - l_p m_p g\cos(\hat{\theta}_p)} \end{cases} \tag{4}$$

En omskrivning av detta gör det enklare att implementera.

$$\begin{cases} k_1 = k_{11} - k_3 \\ k_2 = k_{21} + k_{22}\cos(\hat{\theta}_p) - k_{23}u\sin(\hat{\theta}_p) \\ k_3 = \frac{k_{31}}{\alpha u \sin(\hat{\theta}_p) - k_{32}\cos(\hat{\theta}_p)} \end{cases}$$

Detta möjliggör förberäkning av konstanter enligt

$$\begin{cases} k_{11} = 2\zeta_k\omega_k + \alpha_k \\ k_{21} = \sqrt{\frac{l_p}{g}}(\omega_k^2 + 2\alpha_k\zeta_k\omega_k) \\ k_{22} = \sqrt{\frac{g}{l_p}} \\ k_{23} = \sqrt{\frac{l_p}{g}}\frac{\alpha}{l_p^2 m_p} \\ k_{31} = \alpha_k\omega_k^2 l_p^2 m_p \\ k_{32} = l_p m_p g \end{cases}$$

Med polplacering enligt stycket 2.10 blir K matrisen i storleksordningen

$$\begin{cases} k_1 \approx 44 \\ k_2 \approx 96 \\ k_3 \approx -22 \end{cases}$$

2.4 Diskretisering av kalmanfiltret

Diskretisering av kalmanfiltret enligt ekvation 3 och 4 med en framåt approximation ger

$$\delta = \frac{q-1}{h}$$

$$\begin{cases} \hat{\theta}_p(k+1) = \hat{\theta}_p(k) + h\left(\sqrt{\frac{g}{l_p}}\hat{v}_p(k) + k_1(y(k) - \hat{y}(k))\right) \\ \hat{v}_p(k+1) = \hat{v}_p(k) + h\left(\sqrt{\frac{g}{l_p}}\sin(\hat{\theta}_p(k)) + \frac{\sqrt{l_p}\alpha u}{m_p\sqrt{g}l_p^2}\cos(\hat{\theta}_p(k)) + k_2(y(k) - \hat{y}(k))\right) \\ \hat{b}_p(k+1) = \hat{b}_p(k) + hk_3\,(y(k) - \hat{y}(k)) \end{cases} \quad (5)$$

2.5 Reglering i SISO fallet

Tre olika alternativ för att reglera pendeln i inverterat läge har framtagits i SISO fallet.

2.5.1 Reglering med tillståndsåterkoppling

För beräkning av återkopplingsvektorn $L = \begin{pmatrix} l_1 & l_2 \end{pmatrix}$ använder vi oss av en enklare modell av pendeln med endast θ_p och v_p som tillstånd. Den kring $\theta_p = 0$ linjäriserade modellen blir då

$$\begin{cases} \frac{d\theta_p}{dt} = \sqrt{\frac{g}{l_p}}v_p \\ \frac{dv_p}{dt} = \sqrt{\frac{g}{l_p}}\theta_p + \frac{\alpha}{m_p l_p^2}u \end{cases}$$

Använd styrlagen

$$u = -l_1\theta_p - l_2 v_p$$

ger det slutna systemets karakteristiska polynom

$$det(sI - A + BL) = s^2 + \frac{\alpha}{m_p l_p^2} l_2 s + \sqrt{\frac{g}{l_p}}\frac{\alpha}{m_p l_p^2} l_1 - \frac{g}{l_p}$$

identifiering mot önskat polynom som är

$$s^2 + 2\zeta_s\omega_s s + \omega_s^2$$

ger att vektorn L bör väljas som

$$\begin{cases} l_1 = \frac{\omega_s^2 + \frac{g}{l_p}}{\frac{\alpha}{m_p l_p^2}\sqrt{\frac{g}{l_p}}} \\ l_2 = \frac{2\zeta_s\omega_s}{\frac{\alpha}{m_p l_p^2}} \end{cases}$$

2.5.2 Reglering med tillståndsåterkoppling med integrator

Att införa en regulator med integrerande verkan i regulatorn kan ske genom att införa ett nytt tillstånd i kalmanfiltret, och därefter använda en linjär tillståndsåterkoppling.

$$\begin{cases} \frac{d\theta_p}{dt} = \sqrt{\frac{g}{l_p}} v_p \\ \frac{dv_p}{dt} = \sqrt{\frac{g}{l_p}}\theta_p + \frac{\alpha}{m_p l_p^2} u \\ \frac{dz}{dt} = -\theta_p \end{cases}$$

Användning av styrlagen $u = -l_1\theta - l_2 v - l_3 z$ ger det slutna systemets karakteristiska polynom:

$$det(sI - A + BL) = s^3 + \frac{\alpha}{m_p l_p^2} l_2 s^2 + \sqrt{\frac{g}{l_p}}\left(\frac{\alpha}{m_p l_p^2} l_1 - \sqrt{\frac{g}{l_p}}\right)s - \sqrt{\frac{g}{l_p}}\frac{\alpha}{m_p l_p^2} l_3$$

Identifiering av koefficienter mot det önskade polynomet $(s + \alpha_s)(s^2 + 2\zeta_s\omega_s s + \omega_s^2)$ ger:

$$\begin{cases} l_1 = \frac{\frac{2\zeta_s\omega_s\alpha_s + \omega_s^2}{\sqrt{\frac{g}{l_p}}} + \sqrt{\frac{g}{l_p}}}{\frac{\alpha}{m_p l_p^2}} \\ l_2 = \frac{2\zeta_s\omega_s + \alpha_s}{\frac{\alpha}{m_p l_p^2}} \\ l_3 = -\frac{\omega_s^2\alpha_s}{\sqrt{\frac{g}{l_p}}\frac{\alpha}{m_p l_p^2}} \end{cases}$$

Implementeringen av det nya tillståndet z har skett med en enkel framåt approximation enligt $\delta = \frac{q-1}{h}$ vilket ger uppdateringen av z som $z(k+1) = z(k) - h\theta$.

2.5.3 Reglering med olinjär transformation

En ide är att tranformera den olinjära differentialekvationen till en linjär dito. Det vi vill åstadkomma är att det slutna systemet

$$\ddot{\theta}_p + k_v\dot{\theta}_p + k_p\theta_p = 0 \tag{6}$$

där $k_v = 2\zeta_s\omega_s$ och $k_p = \omega_s^2$. Insättning av ekvation 6 i pendelns ekvation 1 ger efter omskrivning att styrsignalen skall väljas som

$$u = \frac{J(-k_p\theta_p - k_v\dot{\theta}_p) - m_p g l_p \sin(\theta_p)}{\alpha\cos(\theta_p)}$$

differentialekvationen kan nu skrivas som en dubbel integrator enligt

$$\dot{x} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ \frac{\alpha}{m_p l_p^2} \end{pmatrix} u$$

där

$$x = \begin{pmatrix} \theta_p \\ \dot{\theta}_p \end{pmatrix}$$

2.6 Matematisk modell av pendeln med arm

En från avsnitt 2.1 utökad modell av pendeln omfattar även armen och dess mätbara tillstånd (θ_a, v_a). Fortfarande erhålles samma ekvation för pendeln.

$$\ddot{\theta}_p = \frac{g}{l_p}\sin(\theta_p) + \frac{\alpha u}{m_p l_p^2}\cos(\theta_p)$$

Man bör notera att u styr accelerationen hos pendelns pivotpunkt. På samma sätt erhålles ekvationen för armen.

$$\ddot{\theta}_a = \frac{\alpha}{m_a l_a^2} u$$

Inför tillstånden $\frac{d\theta_p}{dt} = \sqrt{\frac{g}{l_p}}v_p$ och $\frac{d\theta_a}{dt} = \sqrt{\frac{g}{l_a}}v_a$ ger det kring $\theta_p = 0$ linjäriserade systemet

$$x = \begin{pmatrix} \theta_p \\ v_p \\ v_a \\ \theta_a \end{pmatrix}$$

$$A = \begin{pmatrix} 0 & \sqrt{\frac{g}{l_p}} & 0 & 0 \\ \frac{\sqrt{g}\cos(\theta_p)}{\sqrt{l_p}} - \frac{\alpha u \sin(\theta_p)}{l_p^{3/2} m_p \sqrt{g}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \sqrt{\frac{g}{l_a}} & 0 \end{pmatrix}\Bigg|_{(\theta_p=\theta_0, v_p=v_0, b_p=b_0 u=u_0)}$$

$$B = \begin{pmatrix} 0 \\ \frac{\alpha\cos(\theta_p)}{l_p^{3/2} m_p \sqrt{g}} \\ \frac{\alpha}{l_a^{3/2} m_a \sqrt{g}} \\ 0 \end{pmatrix}\Bigg|_{(\theta_p=\theta_0, v_p=v_0, b_p=b_0 u=u_0)}$$

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

2.7 Reglering i SIMO fallet

Två olika alternativ för att reglera pendeln i inverterat läge har framtagits i SIMO fallet. Härvid har certain equivalence principen använts, det vill säga att hela tillståndsvektorn har betraktats som känd.

2.7.1 Reglering med tillståndsåterkoppling

Antag tillståndsåterkoppling med $L = \begin{pmatrix} l_1 & l_2 & l_3 & l_4 \end{pmatrix}$ vilket ger det slutna systemets karakteristiska polynom

$$det(sI - A + BL) =$$

$$s^4 + \left(\frac{\alpha}{l_a^{3/2} m_a \sqrt{g}} l_3 + \frac{\alpha}{l_p^{3/2} m_p \sqrt{g}} l_2\right) s^3 + \left(\sqrt{\frac{g}{l_p}} \frac{\alpha}{l_p^{3/2} m_p \sqrt{g}} l_1 - \frac{g}{l_p} + \sqrt{\frac{g}{l_a}} \frac{\alpha}{l_a^{3/2} m_a \sqrt{g}} l_4\right) s^2$$
$$- \frac{g}{l_p} \frac{\alpha}{l_a^{3/2} m_a \sqrt{g}} l_3 s - \frac{g}{l_p} \sqrt{\frac{g}{l_a}} \frac{\alpha}{l_a^{3/2} m_a \sqrt{g}} l_4$$

Identifiering mot önskat polynom

$$(s^2 + 2\zeta_{s1}\omega_{s1}s + \omega_{s1}^2)(s^2 + 2\zeta_{s2}\omega_{s2}s + \omega_{s2}^2)$$

ger att återkopplingen bör väljas som

$$\begin{cases} l_1 = \dfrac{\left(\omega_{s2}^2 + 4\zeta_{s1}\zeta_{s2}\omega_{s1}\omega_{s2} + \omega_{s1}^2 + \frac{g}{l_p} - \frac{\alpha}{l_a^2 m_a} l_4\right) l_p^2 m_p}{\alpha} \\ l_2 = \dfrac{2\zeta_{s2}\omega_{s2} + 2\zeta_{s1}\omega_{s1} - \frac{\alpha}{l_a^{3/2} m_a \sqrt{g}} l_3}{\frac{\alpha}{l_p^{3/2} m_p \sqrt{g}}} \\ l_3 = \dfrac{2\left(\zeta_{s1}\omega_{s1}\omega_{s2}^2 + \zeta_{s2}\omega_{s2}\omega_{s1}^2\right) l_a^2 l_p m_a}{\alpha\sqrt{g}} \\ l_4 = \dfrac{\omega_{s1}^2 \omega_{s2}^2 l_a^2 l_p m_a}{g\alpha} \end{cases}$$

2.7.2 Reglering med LQ regulator

Att implementera lösning av RICCATI ekvationen i vårat program vore att ta i. Vi har därför använt oss av Matlab för att ta fram den stationära lösningen, och implementerat en ren tillståndsåterkoppling.

En i Matlab beräknad diskret modell för $h = 0.01$ s blir:

$$\begin{pmatrix} \theta_p(k+1) \\ v_p(k+1) \\ v_a(k+1) \\ \theta_a(k+1) \end{pmatrix} = \begin{pmatrix} 1.0018 & 0.0592 & 0 & 0 \\ 0.0592 & 1.0018 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0.07 & 1 \end{pmatrix} \begin{pmatrix} \theta_p(k) \\ v_p(k) \\ v_a(k) \\ \theta_a(k) \end{pmatrix} + \begin{pmatrix} 0.0064 \\ 0.2156 \\ 0.238 \\ 0.0083 \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \theta_p \\ v_p \\ v_a \\ \theta_a \end{pmatrix}$$

Med viktmatriserna Q_1 och Q_2 enligt:

$$Q_1 = \begin{pmatrix} 400 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 25 \end{pmatrix}$$

$$Q_2 = \begin{pmatrix} 1 \end{pmatrix}$$

erhålles den enligt Matlab optimala lösninen av:

$$L = \begin{pmatrix} 21.2361 & 16.5738 & -12.2182 & -3.3234 \end{pmatrix}$$

2.8 Stabilitetsanalys av systemet

Om vi väljer u så att vridmomentet för pendeln blir

$$\tau = \alpha u \cos(\theta_p) = -k_d\dot{\theta}_p - k_p\theta_p - m_p g l_p \sin(\theta_p) \tag{7}$$

får vi det slutna systemet

$$J\ddot{\theta}_p + k_d\dot{\theta}_p + k_p\theta_p = 0 \tag{8}$$

Som Lyaponovfunktion väljer vi

$$V = \frac{1}{2}J\dot{\theta}_p^2 + \frac{1}{2}k_p e^2 \tag{9}$$

där J och k_p är positiva

$$\dot{V} = \frac{1}{2}\dot{J}\dot{\theta}_p^2 + J\dot{\theta}_p\ddot{\theta}_p - k_p e\dot{\theta}_p =$$

$$= \frac{1}{2}\dot{J}\dot{\theta}_p^2 + \dot{\theta}_p(J\ddot{\theta}_p - k_p e)$$

ersätt $J\ddot{\theta}_p$ enligt ekvation 1

$$\dot{V} = \frac{1}{2}\dot{J}\dot{\theta}_p^2 + \dot{\theta}_p(\tau + m_p g l_p \sin(\theta_p) - k_p e)$$

med kraften τ enligt ekvation 7 och $e = -\theta$ erhålles

$$\dot{V} = \frac{1}{2}\dot{J}\dot{\theta}_p^2 + \dot{\theta}_p(-k_d\dot{\theta}_p - k_p\theta_p - m_p g l_p \sin(\theta_p) + m_p g l_p \sin(\theta_p) + k_p\theta_p) =$$

$$= \frac{1}{2}\dot{J}\dot{\theta}_p^2 - k_d\dot{\theta}_p^2$$

eftersom $\dot{J} = 0$ får vi slutligen att

$$\dot{V} = -k_d\dot{\theta}_p^2 \leq 0$$

Lösningarna till ekvation 8 kommer alltså att vara asymptotiskt stabila.

Vi ser även i ekvation 8 att $\lim_{n\to\infty}\theta_p = 0$ eftersom $\lim_{n\to\infty}\dot{\theta}_p = 0$ och $\lim_{n\to\infty}\ddot{\theta}_p = 0$.

Vi har alltså asymptotisk konvergens i lösningen för θ_p.

2.9 Styrstrategi för uppkast av pendel

Om vi antar att lägesenergin för pendeln är 0 i hängande läge, så behöver pendeln den totala energin

$$W_{uppe} = 2m_p g l_p$$

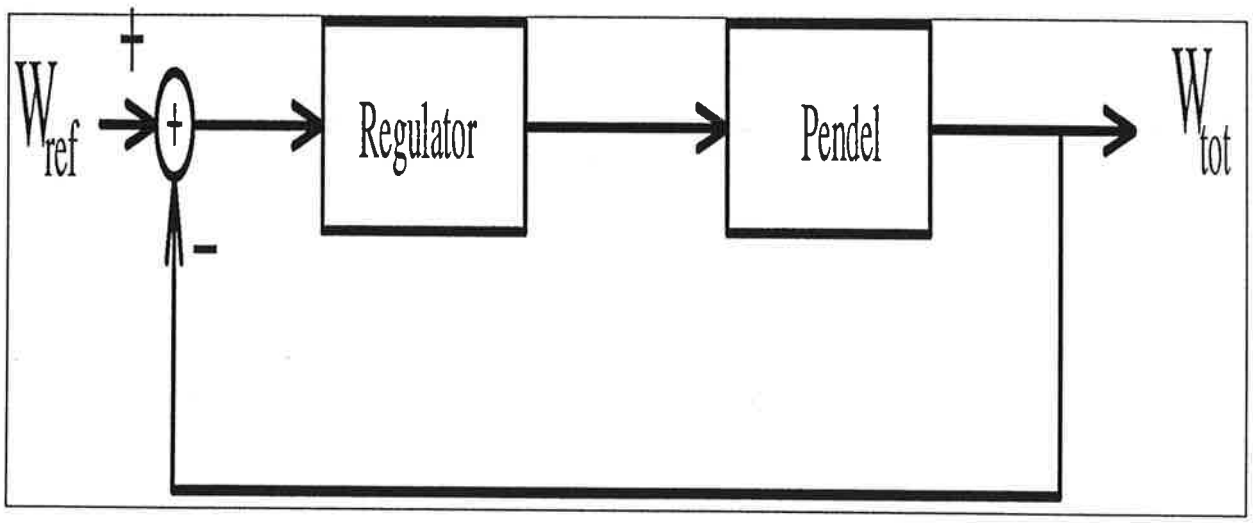
för att kunna balansera i upprätt läge. Den energikick pendeln behöver för ett uppkast, tillför vi genom att accelerera pivotpunkten så att rörelseenergin blir

$$\frac{J\dot{\theta}_p^2}{2} = 2m_p g l_p$$

Vi kan formulera problemet som ett reglerproblem, där vi ska konstruera en regulator som ställer in den totala energin

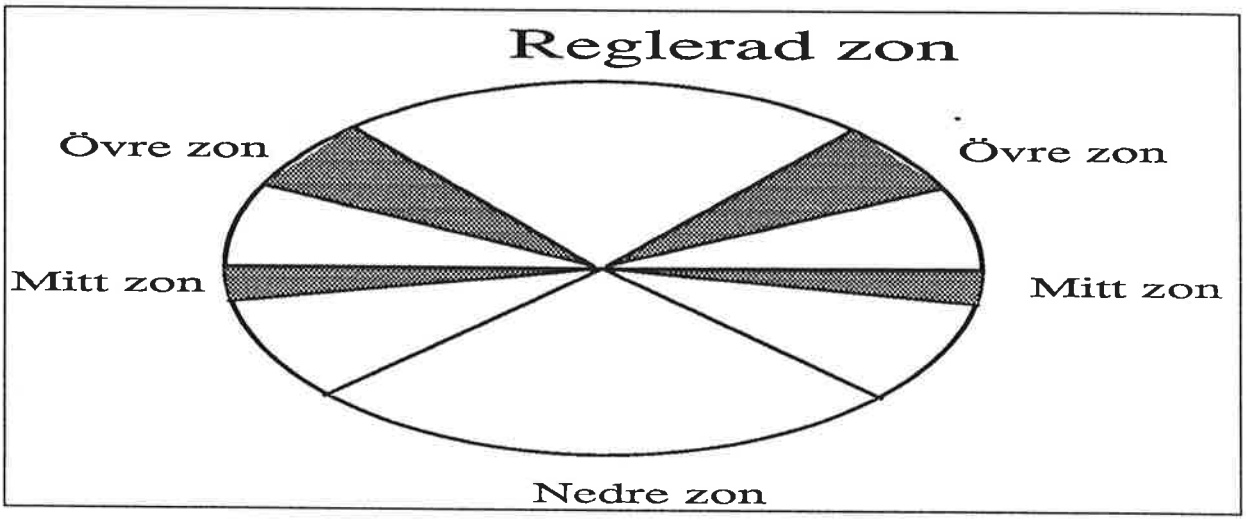
$$W_{tot} = \frac{J\dot{\theta}_p^2}{2} + m_p g l_p(\cos(\theta_p) + 1) \tag{10}$$

till önskad energi W_{uppe}.



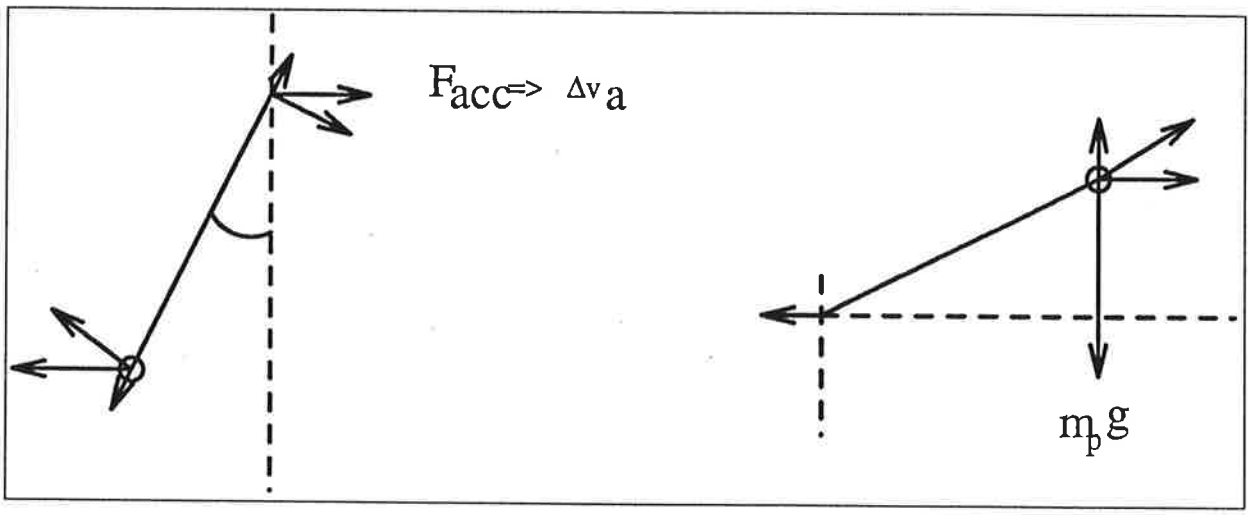
Figur 4: Regulator för uppkast av pendeln.

Vi har designat regulatorn så att den klarar av att få upp pendeln på ett kick. Den styrstrategi vi valt för uppkastet uttnyttjar pendelns egenheter genom att dela in uppkastet i zoner.



Figur 5: Bilden visar hur vi har delat in regulatorns arbetsområde i olika zoner. Observera att zongränserna inte bara beror av skattade vinkeln θ_p utan även v_p.

Indelningen av zonerna kan motiveras genom att undersöka hur en hastighetsändring, orsakad av en impulslik acceleration av pivotpunkten, ändrar på $\dot{\theta}_p$ och därmed den totala energin W_{tot}.



Figur 6: Schematisk bid av pivotpunktens kraftpåverkan på pendeln i olika lägen.

Pendeln kommer att ändra $\dot{\theta}_p$ enligt

$$\Delta\dot{\theta}_p = \frac{\Delta v_a}{l}\cos(\theta_{p0})$$

Denna ekvation säger oss att det är lättast att påverka pendelns energi då $|\cos(\theta_p)|$ är nära ett.

I nedre zonen har vi en till/från- regulator som tillför så mycket energi att vi kan komma upp till den övre zonen, där resterande energi tillförs med en p-regulator. När pendeln är i mittzonen ger en hastighetsändring av pivotpunkten ingen märkbar förändring av pendelns totala energin, eftersom $|\cos(\theta_p)|$ är nära noll. Detta innebär att vi i mittzonen kan passa på att minska armens hastighet utan att förlora för mycket av pendelns totala energi.

Den reglerade zonen bestäms av den maximala vinkelavvikelse som regulatorn kan hämta upp. Uppfångningsområdet bestäms i första hand av hur stor styrsignal u_{max} som vi kan skicka in i systemet. I figur 6 ser vi att vinkeln ges av

$$m_p g = \alpha u_{max} \cos(\theta_{p,max}) \Rightarrow \theta_{p,max} = \arccos(\frac{m_p g}{\alpha u_{max}})$$

Nu är det inte bara vinkeln som bestämmer uppfångningsområdet, utan vi måste även ta hänsyn till vinkelhastigheten. Vi fann det lämpligt att använda lyaponovfunktionen 9 som ett mått på den "energi" regulatorn kan hämta upp.

2.10 Val av lämplig polplacering

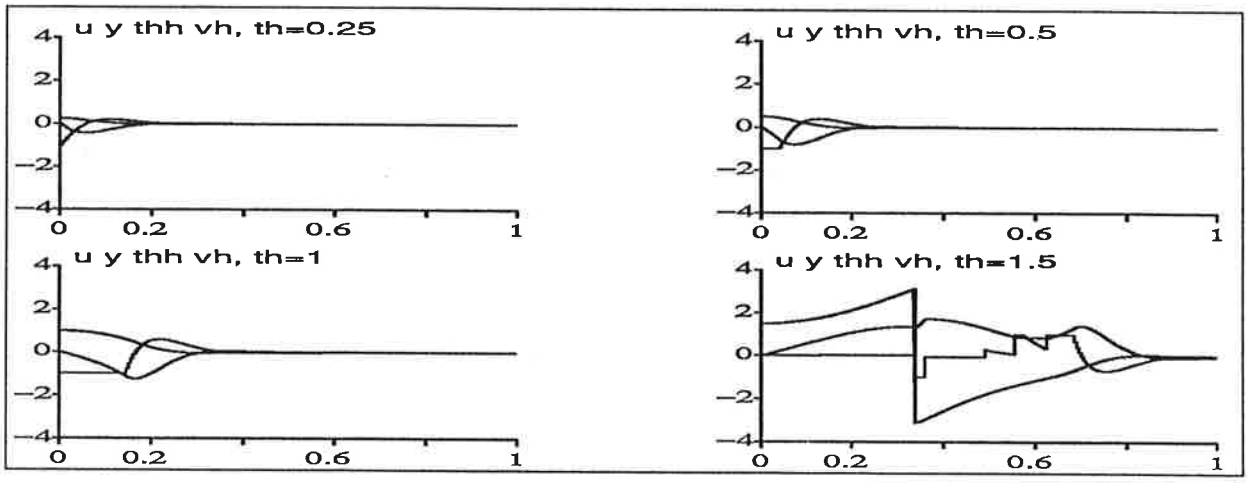
Val av polplacering till detta system har skett genom att styra variablerna ζ_s, ω_s i det slutna systemet och observerarpoler via variablerna $\alpha_k, \omega_k, \zeta_k$. Problemet att välja dessa på lämpligt sett återstår. I vårat fall skedde detta genom att prova systemet med rimliga värden och därefter heuristiskt justera tills en god reglering uppstod. Detta resulterade i följande parametervärden:

$$\begin{cases} \zeta_s = 0.6 \\ \omega_s = 18 \\ \alpha_k = 3 \\ \omega_k = 30 \\ \zeta_k = 0.7 \end{cases}$$

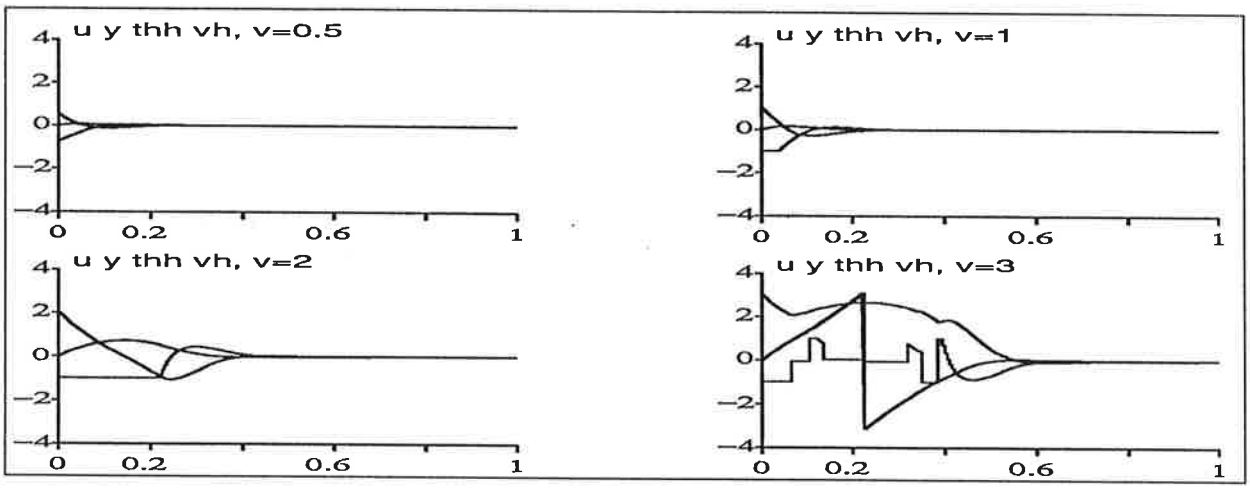
3 Simulering av Systemet

Simuleringarna har delats upp i två avdelningar. I det första avsnittet behandlas hur regulatorn fungerar och begränsar sig till pendeln i den reglerade zonen enligt figur 5. Det andra avsnittet behandlar simulering av uppkastning, och följer med andra ord pendeln varvet runt.

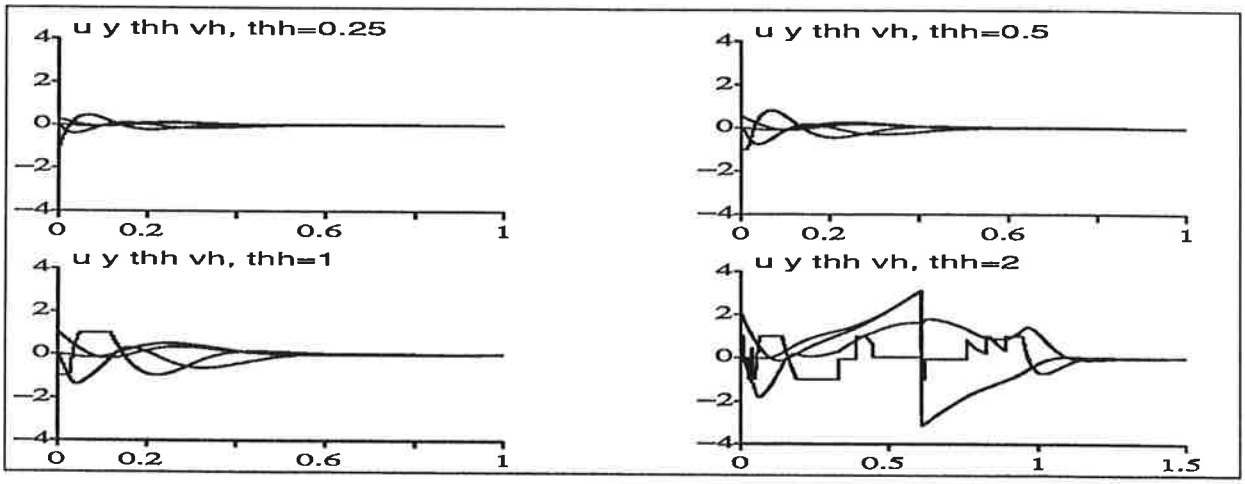
3.1 Pendeln i övre läget med olika avvikelser i begynnelsevärden



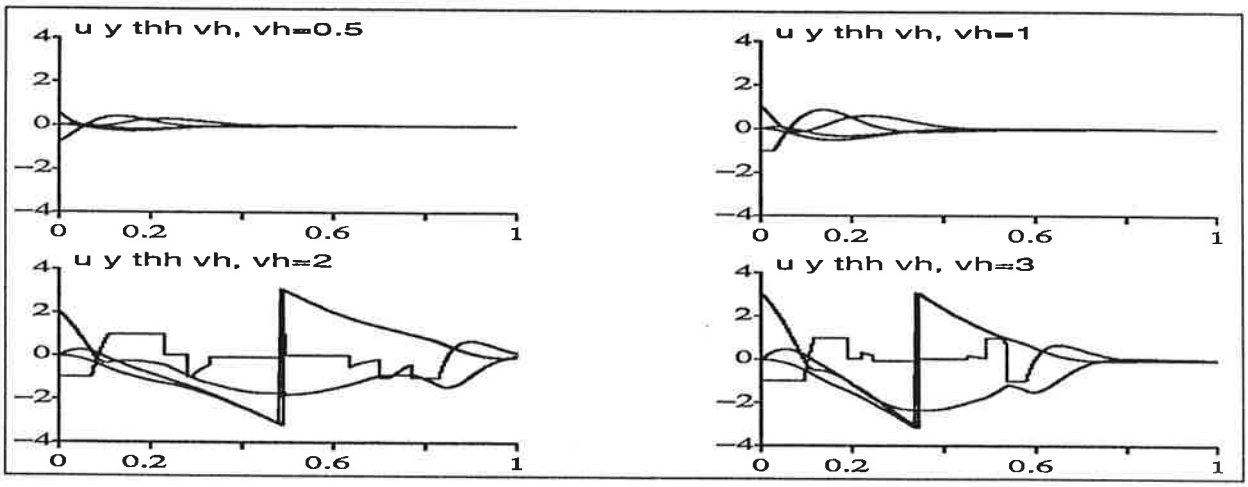
Figur 7: Simulering av pendeln i inverterat läge med olika startvinklar θ_p initialt.



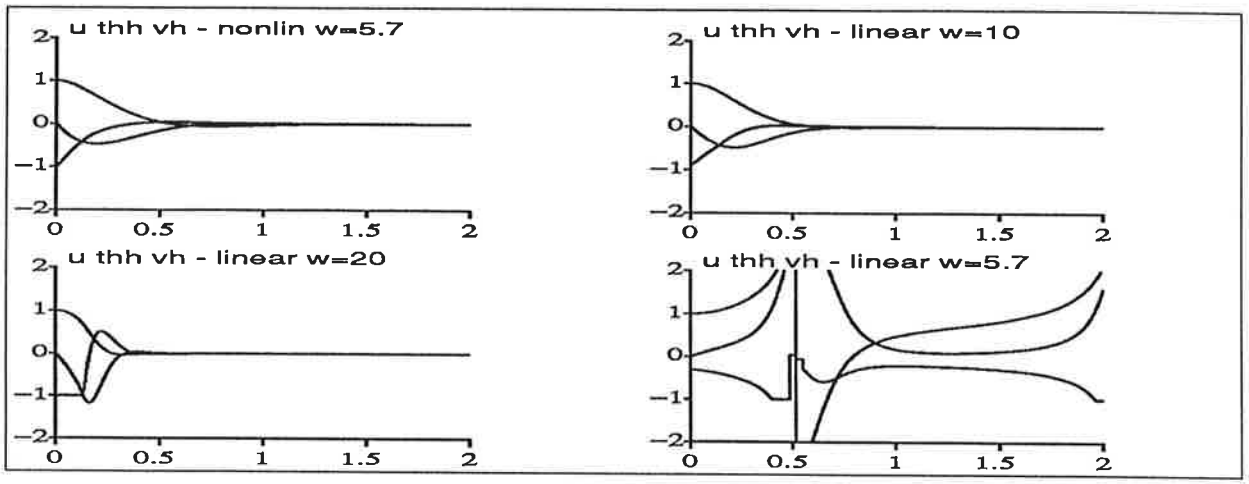
Figur 8: Simulering av pendeln i inverterat läge med olika hastigheter v_p initialt.



Figur 9: Simulering av pendeln i inverterat med olika initialvärden för θ_p i processen och Kalmanfiltret. Processen har initialvärdet noll.



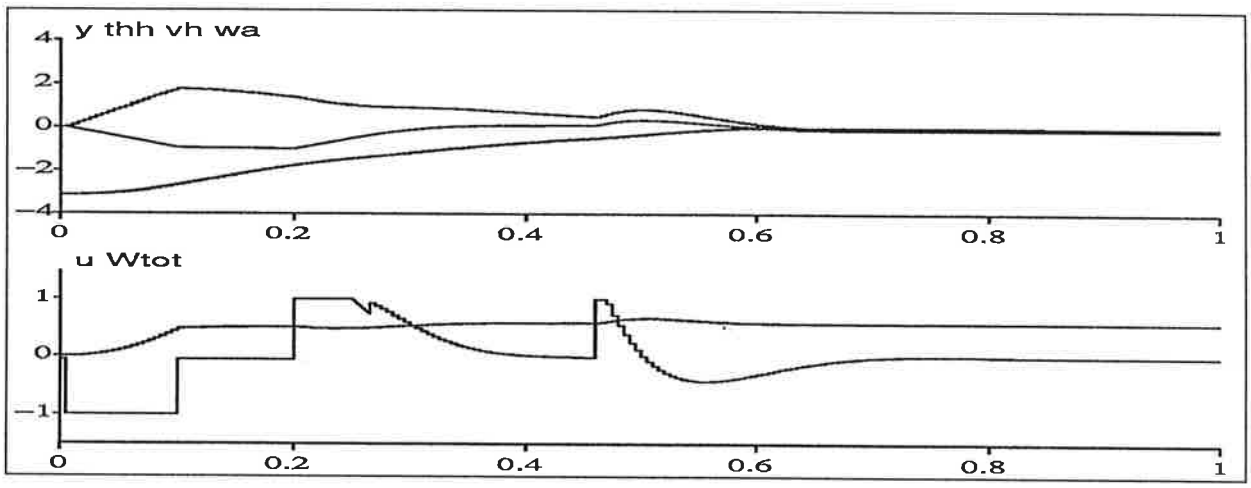
Figur 10: Simulering av pendeln i inverterat med olika initialvärden för v_p i processen och Kalmanfiltret. Processen har initialvärdet noll.



Figur 11: Jämförelse mellan den linjära regulatorn och den olinjära.

Ur ovanstående simuleringar drar vi slutsatsen att systemet bör gå att reglera med de strategier vi tänkt oss.

3.2 Prov av uppkastningsstrategi

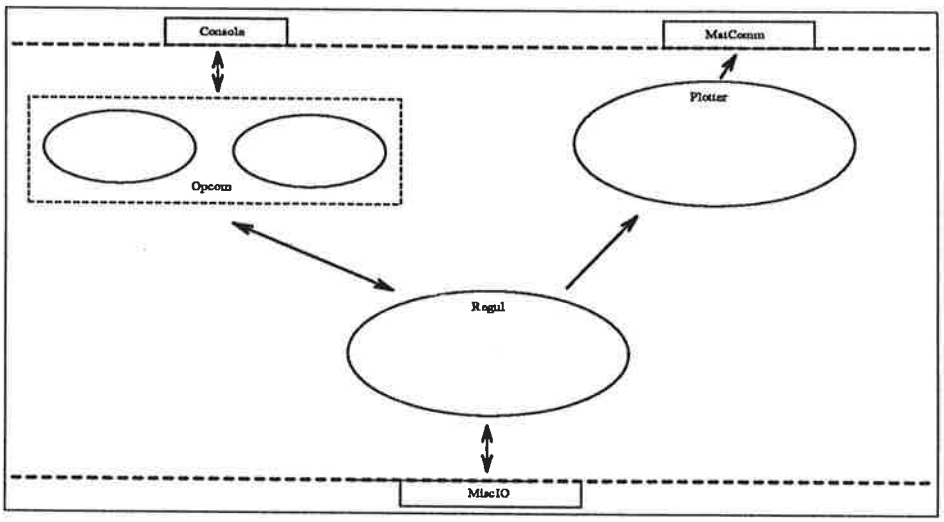


Figur 12: Simulering av uppkastet med den strategi som finns beskrivet i Teori delen.

4 Realisering

Realisering av regersystemet sker på det datorsystem som finns beskrivet i avsnittet *Inledning 1.*

4.1 Processer



Figur 13: Processgraf.

Rektanglarna Console, MatComm och MiscIO avser färdiga bibliotek som vi använt oss av.

4.2 Processbeskrivningar

De kommandon som implementerats är:

Kommando	*Verkan*
`help`	Listar alla tillgängliga kommandon.
`start`	Startar regleringen (oavsett vilken regulator som valts).
`stop`	Stoppar regleringen $u \equiv 0$ (oavsett vilken regulator som valts).
`reset`	Initierar en skattning av biasen (regulatorn skall vara stoppad).
`par`	Visar eller sätter parametrar.
`linear2`	Väljer regluatorn enligt stycket 2.5.1.
`integrating`	Väljer regluatorn enligt stycket 2.5.2.
`lq`	Väljer regluatorn enligt stycket 2.7.2.
`linear4`	Väljer regluatorn enligt stycket 2.7.1.

De parametrar som kan sättas av användaren är:

Parameter	*Betydelse*
h	Samplingstid i sek.
alphak	Kalmanfiltrets placering av "tredje polen".
zetak	Kalmanfiltrets dämpning.
omegak	Kalmanfiltrets "snabbhet".
zetas	Slutna systemets dämpning vid reglering enligt stycket 2.5.1.
omegas	Slutna systemets snabbhet vid reglering enligt stycket 2.5.1.
alphas	Slutna systemets "tredje pol" vid reglering enligt stycket 2.5.1.
mp	Pendelns massa.
lp	Pendelns längd.
ma	Armens massa.
la	Armens längd.
alpha	Skalfaktor från utsignal till acc. i pivotpkt.
eps	Odokumenterad feature.
zreg	Zongräns för regulatorn.
zupp	Zongräns för sista energikicken.
zres	Zongräns för retardation av armen.
zkick	Zongräns för kick i nedre läge.
ukick	Maximal utsignal vid kick.
umax	Maximal utsignal.
ulow	Minsta kicksignal, istället för konstant hastighet.
fkick	Kickenergi i nedre zon.
fupp	Kickenergi i övre zon.
lupp	Förstärkning i regulatorn för sista kicken.
lres	Förstärkning i regulatorn för retardationen av armen.
l1	Tillståndsåterkoppling enligt stycket 2.7.2.
l2	Tillståndsåterkoppling enligt stycket 2.7.2.
l3	Tillståndsåterkoppling enligt stycket 2.7.2.
l4	Tillståndsåterkoppling enligt stycket 2.7.2.
omegas1	Slutna systemets snabbhet vid reglering med tillståndsåterkoppling från fyra tillstånd
omegas2	Slutna systemets snabbhet vid reglering med tillståndsåterkoppling från fyra tillstånd
zetas1	Slutna systemets dämpning vid reglering med tillståndsåterkoppling från fyra tillstånd
zetas2	Slutna systemets dämpning vid reglering med tillståndsåterkoppling från fyra tillstånd

4.2.1 OpCom

Inom modulen OpCom finns två processer för hanteringen av operatörskommunikationen, dessa är OpComLoop och Command.

4.2.2 OpComLoop

Hjälp modul till Opcom.

4.2.3 Command

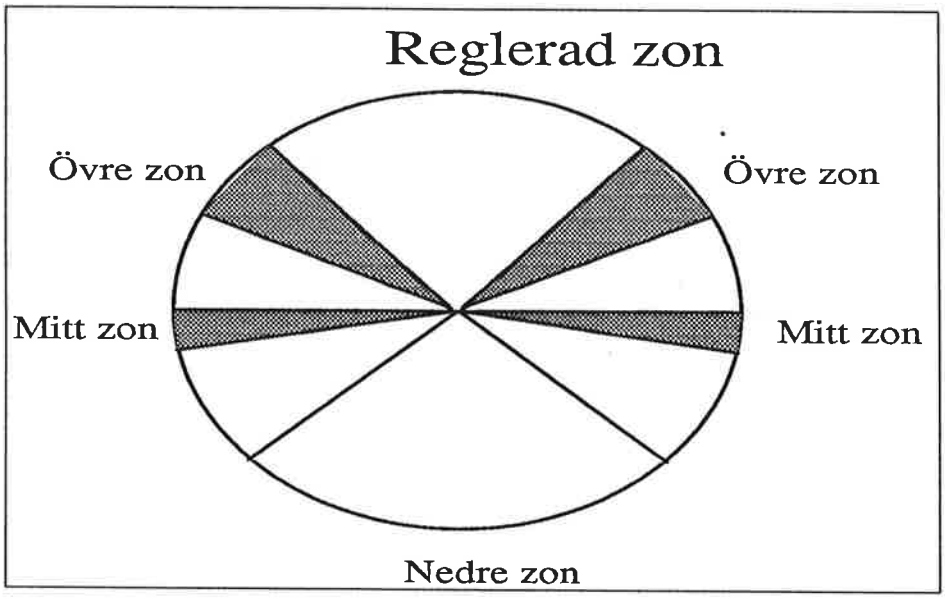
Hjälp modul till Opcom.

4.2.4 Regul

Det principiella arbetsättet för regulatorn framgår av figur 14. Vad som däremot kan vara lite svårare att få grepp om är de "fixar" som gjorts får att regulatorn skall fungera inte bara på

pappret utan även i verkligheten. Dessa omfattar:

- Skydd mot integrator uppvridning av tillståndet z i Kalman filtret.
- Begränsning av θ_p vid 90^o.
- Modulusfunktion för θ_p vid 180^o.
- Skattning av b_p endast vid `reset`.



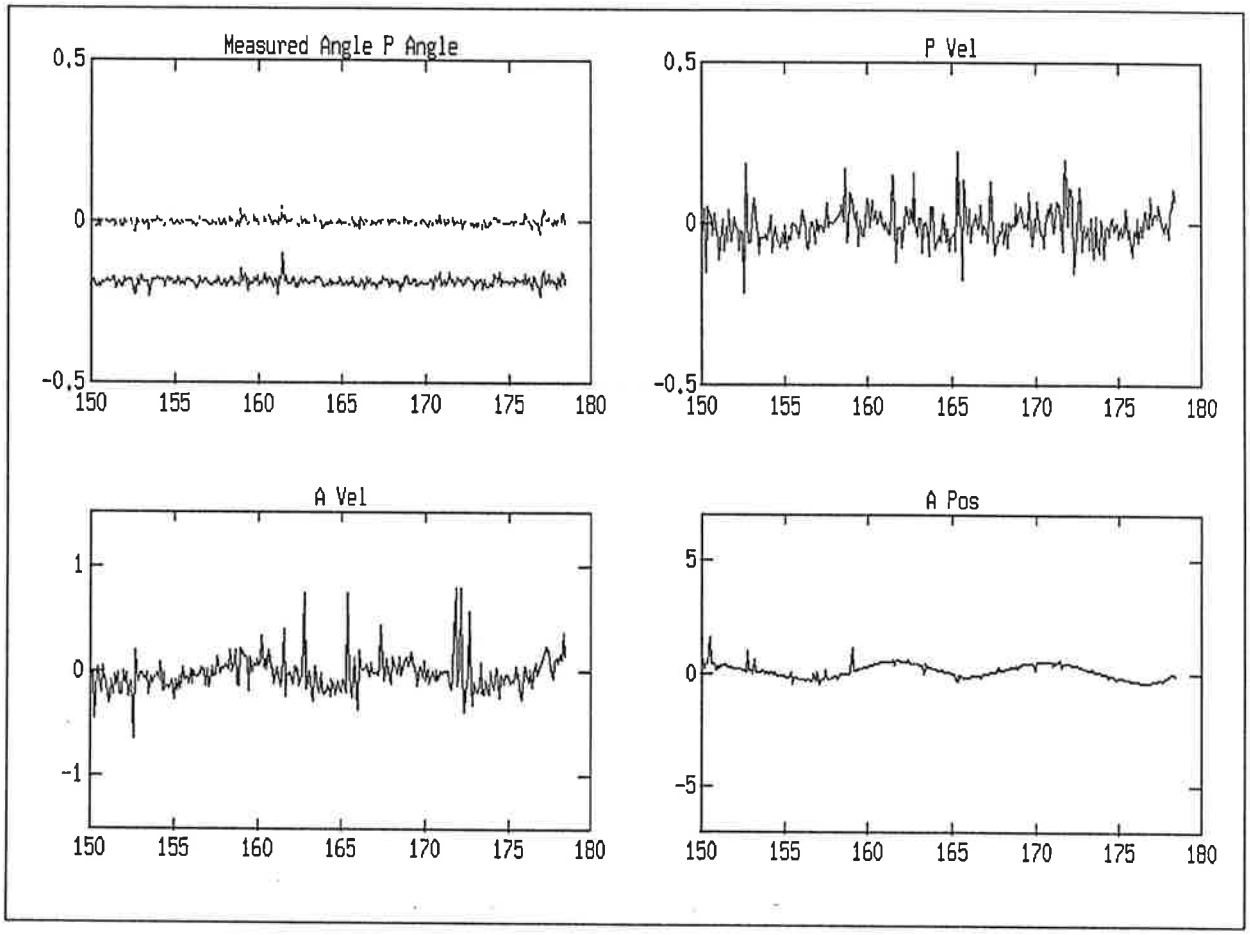
Figur 14: Schematisk bild över hur regulatorn skall uppträda i de olika fallen.

4.2.5 Plotter

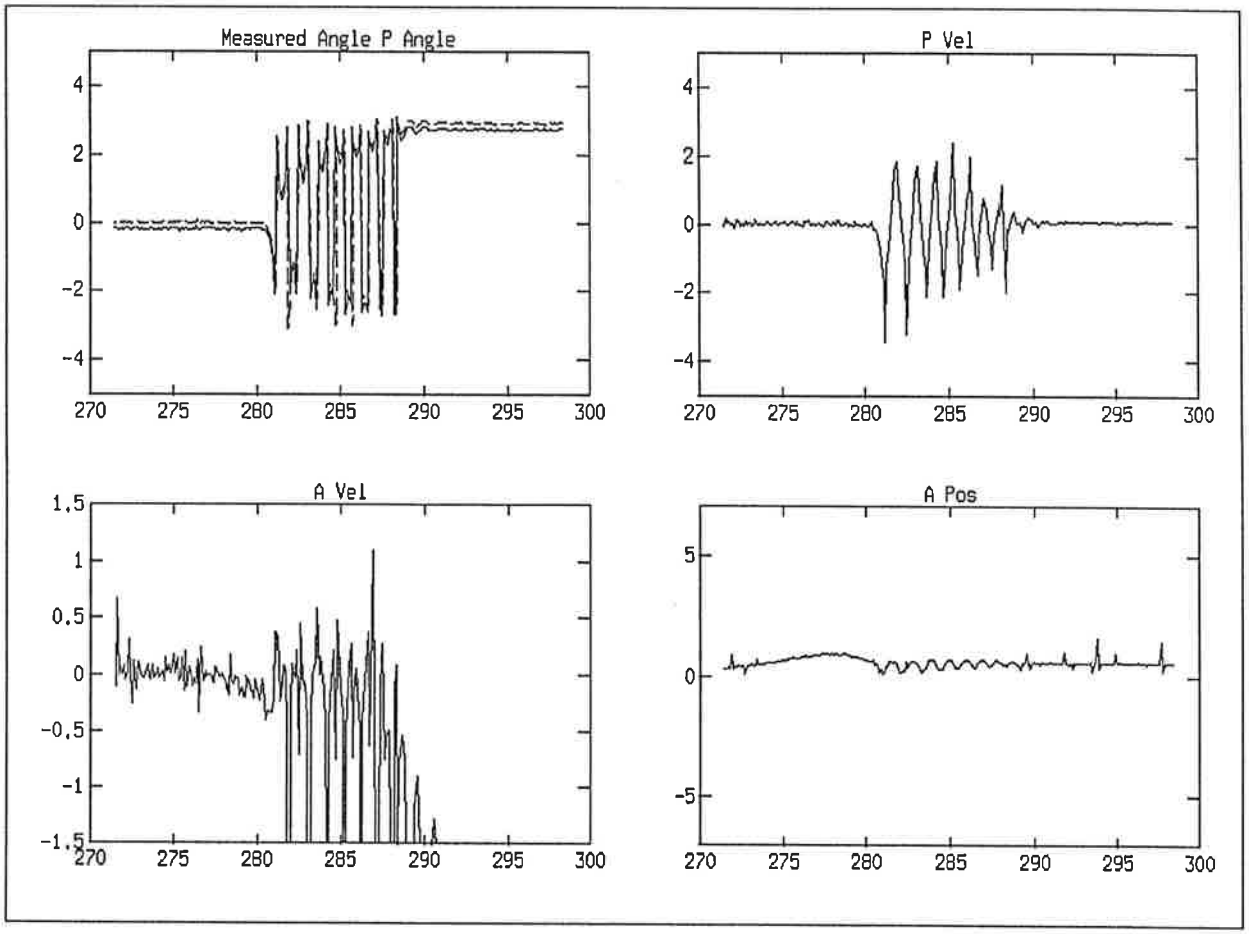
Modul för sändande av data till arbetsstation för plottning med Matlab.

5 Resultat

Det program som så småningom kördes visade att det var möjligt att reglera pendeln i övre läget, vilket framgår av nedanstående bildsvit:

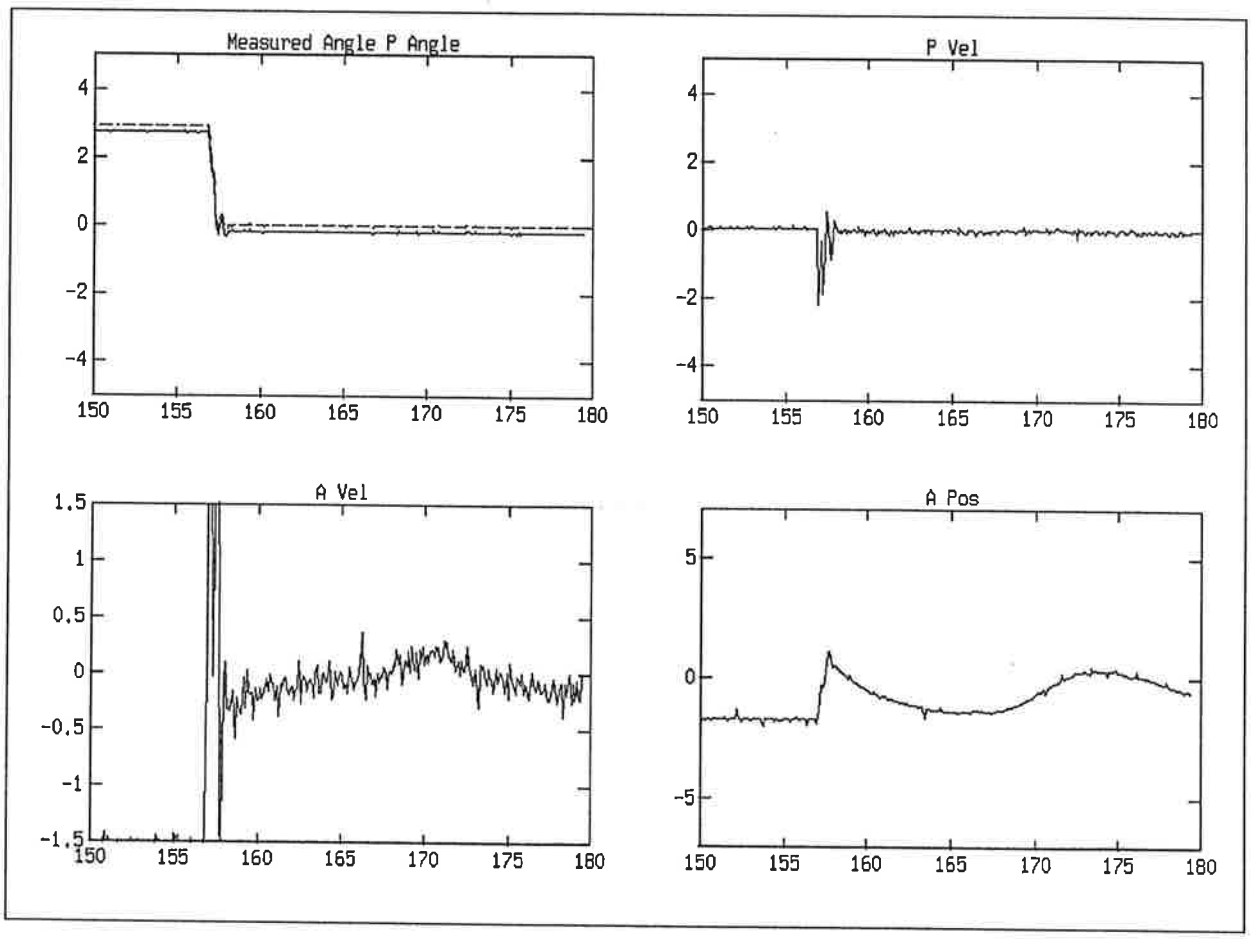


Figur 15: Resultat av körning med pendeln initialt uppställd i övre läget.

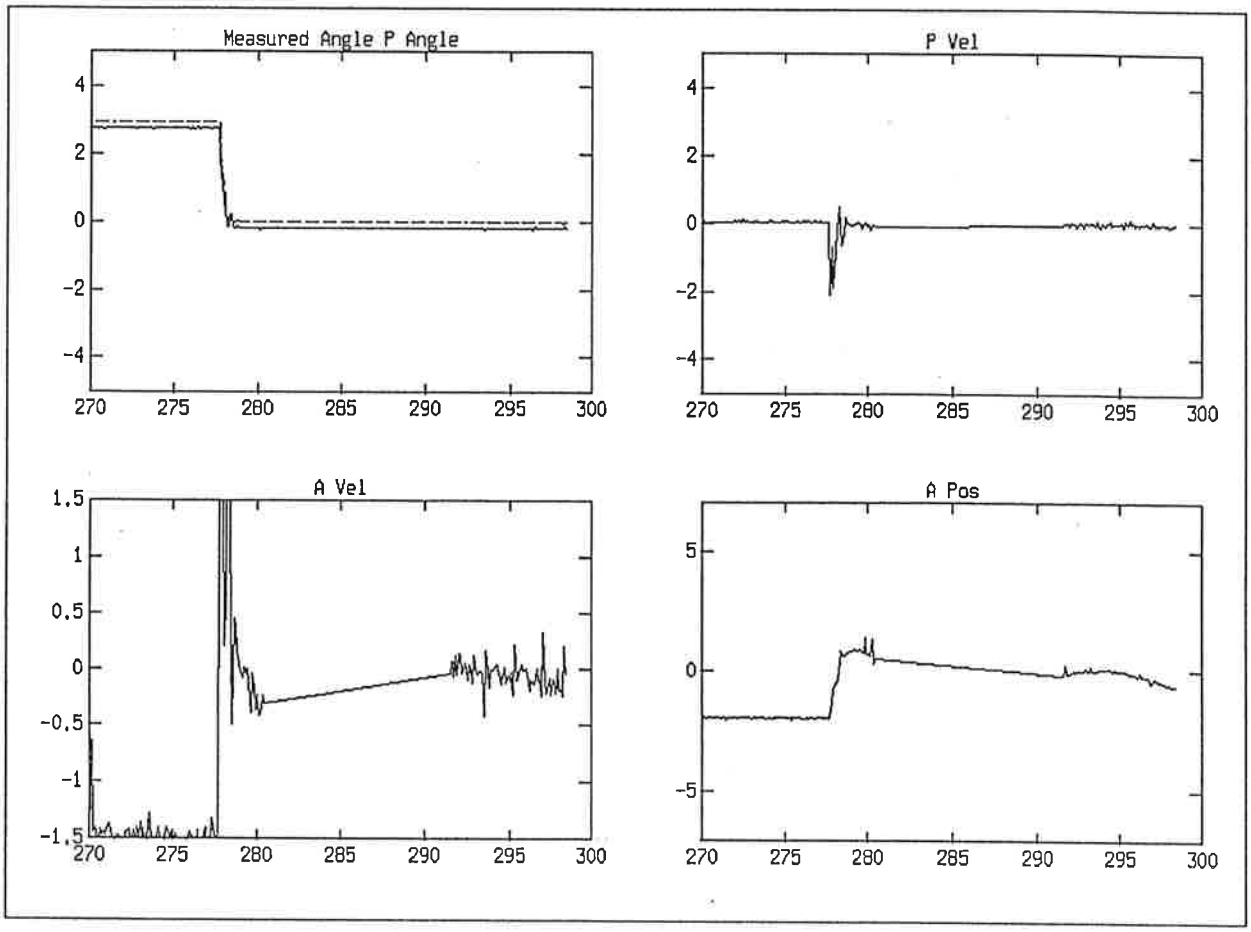


Figur 16: Resultat av körning med pendeln initialt uppställd i övre läget, vid 280 mätvärden slutade vi (medvetet) reglera.

Efter en del trixande lyckades vi även att få uppkastet att fungera tillfredställande.



Figur 17: Uppkast av pendeln. Notera att det går på en kick.



Figur 18: Uppkast av pendeln. Notera att försöket är reproducerbart.

5.1 Slutsats

Det visade sig att angreppssättet med Kalmanfilter för att skatta vinkeln, hastigheten och biasen för pendeln var mycket lämplig, ty filtret gav en god skattning av de tre storheterna runt hela varvet. Däremot var det ej möjligt att reglera pendeln i övre läget med styrlag baserat endast på de tre ovan nämda storheterna. Pendel började då driva med konstant acceleration. Vi har även visat att det går bra att kicka upp pendeln på ett uppkast. Den implemeneterade regulatorn visade mycket goda egenskaper mot laststörningar, utan att armen drev eller oscillerade.

Referenser

[1] Karl Johan Åström. *Reglerteori.* Almqvist & Wiksell 1985

[2] Karl Johan Åström, Björn Wittenmark. *Adaptive control.* Addison-Wesley 1989.

[3] Karl Johan Åström, Björn Wittenmark. *Computer controlled systems, theory and design.* Prentice Hall 1990, andra upplagan.

[4] Rolf Johansson. *Processidentifiering.* KF sigma 1991

[5] Torsten Söderström, Petre Stoica. *System Identification.* Prentice Hall 1989.

[6] Lennart Ljung. *System Identification, theory for the user.* Prentice Hall 1987.

[7] John J. Craig. *Introduction to robotics.* Addison-Wessley 1989.

A Använda förkortningar och beteckningar

SISO	System med en ingång och en utgång.
SIMO	System med en ingång och flera utgångar.
θ_p	Vinkeln mellan pendeln och lodlinjen (utgående från inverterat läge).
v_p	Hastigheten hos pendeln.
m_p	Pendelns massa.
l_p	Pendelns längd.
τ	Vridmoment på pendeln.
J	Tröghetsmoment för pendeln.
θ_a	Vinkeln hos armen.
v_a	Hastigheten hos armen.
m_a	Armens massa.
l_a	Armens längd.
α	Skalfaktor mellan modellens styrsignal och verkligheten.
ω_k	Kalmanfiltrets "snabbhet".
ζ_k	Kalmanfiltrets dämpning.
α_k	Kalmanfiltrets placering av "tredje polen".
ω_s	Slutna systemets snabbhet vid reglering med tillståndsåterkoppling från θ_p och v_p, samt eventuellt z.
ζ_s	Slutna systemets snabbhet vid reglering tillståndsåterkoppling från θ_p och v_p, samt eventuellt z.
α_s	Slutna systemets "tredje pol" vid reglering med integrerande regulator.
ω_{s1}	Slutna systemets snabbhet vid reglering med tillståndsåterkoppling från fyra tillstånd
ω_{s2}	Slutna systemets snabbhet vid reglering med tillståndsåterkoppling från fyra tillstånd
ζ_{s1}	Slutna systemets dämpning vid reglering med tillståndsåterkoppling från fyra tillstånd
ζ_{s2}	Slutna systemets dämpning vid reglering med tillståndsåterkoppling från fyra tillstånd

B Utskrifter av definitionsmoduler

C Utskrifter av modula-2 program

D Utskrifter av simnon program

Studie av artikeln:
Robust Adaptive Control: A Unified Approach

Av Martin Strand

Kort referat

I artikeln "Robust Adaptive Control: A Unified Approach" [1] görs ett försök att sammanföra de olika adaptiva algoritmerna i en enhetligt teoretisk struktur. Målet är att konstruera robusta adaptiva regulatorer. Strukturen som införs består i en uppdelning av algorimtmerna i en parameterestimeringsdel och en regulatorstrukturdel dvs, strukturen för en "Self tuning regulator" definierad i [2]. För estimeringsalgoritmerna visas m.h.a införandet av en "normaliserade signal" att algoritmernas robusthet kan garanteras. För regulatorstrukturerna ges krav på dynamiken för den omodellerade delen av processen, för att garantera stabilitet.
"Olika" adaptiva regulatorer kan sedan konstrueras genom att kombinera olika varianter av estimering respektive reglering. Genom att utnyttja "the Certanty Equivalence Approach" garanteras den adaptiva regulatorns robusthet av estimeringsalgoritmens respektive regulatorstrukturens robusthet var för sig.

Inledning

I den här rapporten görs ett försök att verifiera de mer praktiska aspekterna på några av de föreslagna metoderna i ovan refererade artikell. I artikeln finns fyra exempel på icke rubust beteende. Två av dessa exempel har använts för att testa de i artikeln presenterade metoderna. Processerna och regulatorerna har simulerats i Simnon.
Både införandet av en "normaliserade signal" och garantierna för stabila styrlagar kräver *a priori* kunskap om modellfelet. Kraven på modellfelet är tämligen konservativa, vilket betyder att metodens praktiska nytta är begränsad, då det krävs ingående kunskap om processens *okända* delar för att uppnå en robust regulator.

Simulering i Simnon

Instabilitet pågrund av hög förstärkning

Exempel B i artikeln [1] beskriver hur en adaptiv algoritm konstruerad för ett förenklat system kan uppträda. I exemplet används processen

$$\dot{x}=ax+z-u$$

$$\mu\dot{z}=-z+2u$$

$$y=x\ .$$

En adaptiv algoritm designas mha MIT-regeln under förutsättningen att $\mu=0$ vilket leder till parameteruppdateringsalgoritmen

$$u=-kx$$

$$\dot{k}=\gamma\varepsilon x\ ;\ \varepsilon=x.$$

Analys av den verkliga processen reglerad med en proportionell regulator ger att förstärkningen måste uppfylla följande krav:

$$a< k < \frac{1}{\mu}-a. \qquad (i.1)$$

Eftersom $\dot{k} > 0$ följer det att om k(0) ej uppfyller (i.1) kommer det slutna systemet att vara instabilt. I *fig.1* kan man se hur k(t) divergerar då k(0) valts utanför konvergensintervallet (i.1).

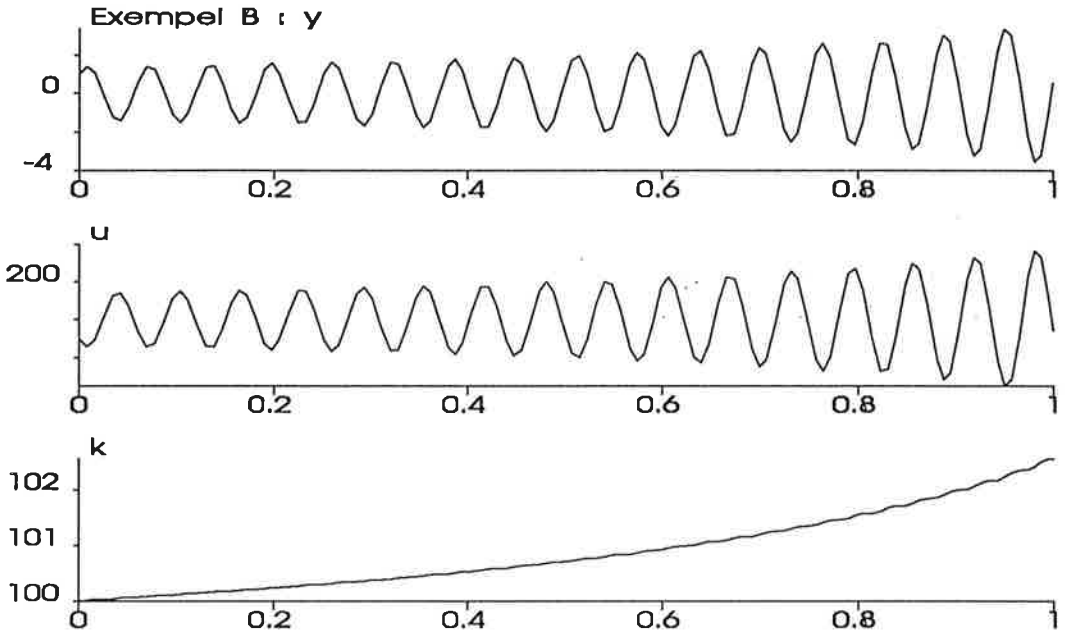


fig1. k(t) med a=1 , μ=0.01, γ=1 och k(0)=100 .

Om däremot k(0) väljs så att (i.1) uppfylls kommer systemet att vara stabilt vilket illustreras i *fig.2*.

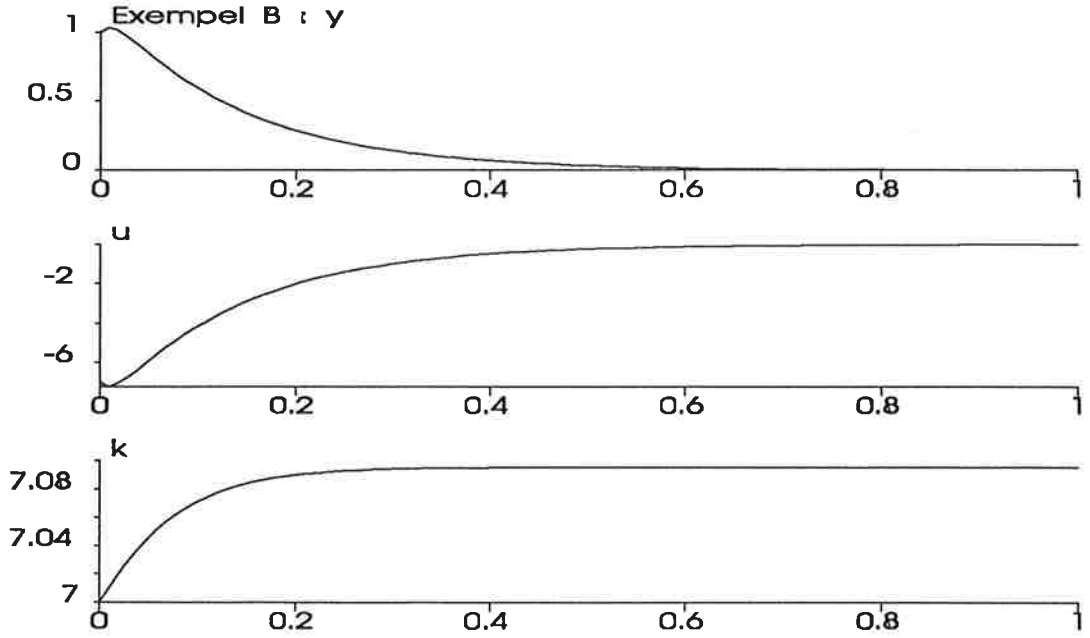


fig2. y(t) då k(0) valts innanför konvergensintervallet. y(0)=1, k(0)=5, a=1, μ=0.01 och γ=1.

På detta exempel appliceras en direkt adaptiv algoritm designad enligt exempel 6.1 i artikeln. Artikeln föreslagna trestegsalgoritm har använts.

steg 1: Som styrlag väljes en "modellreferensregulator". För att anpassa styrlagen till exemplet ovan väljs referensmodellen till identiskt noll vilket leder till följande styrlag:

$$u=\Theta_0^* y;\ \Theta_0^*=a$$

Om den nominella processmodellen ej överenstämmer med den verkliga, dvs modellfelet $\Delta m(s)\neq 0$, granteras det slutna systemets stabilitet av att $\Delta m(s)$ uppfyller (i.2) (visas i appendix 1 del I).

$$\left|\left|\frac{a\Delta m(s)}{s}\right|\right|_\infty^{\delta_0} \leq 1 \qquad \text{(i.2)}$$

steg 2: Gradientmetoden väljs för parameterskattningen. Processen parametriseras om för att anpassas till en regressions modell ur vilken Θ_0^* kan skattas.

steg 3: Analys av den adaptiva regulatorn, som uppstått i kombinationen av steg 1 och steg 2, med avseende på robusthet ger kraven (i.3),(i.4) och (i.5) på modellfelet $\Delta_m(s)$.

i. $\Delta_m(s)$ analytisk i Re[s] $\geq -\frac{\delta_0}{2}$ för något $\delta_0>0$ (i.3)

ii. $\Delta_1=\left|\left|\frac{\Delta m(s)}{s}\right|\right|_\infty^{\delta_0} \leq c$ för något $c>0$ (i.4)

iii. $\Delta_2=\left|\left|\frac{(s+\delta_0)\Delta m(s)}{s}\right|\right|_\infty^{\delta_0} \leq c\delta_0$ för något $c>0$ (i.5)

$\Delta_m(s)$ uppfyller (i.3), (i.4) och (i.5) (visas i appendix 1 del II) vilket betyder att den konstruerade regulatorn bör vara robust m.a.p val av initial förstärkning i styrlagen. Detta verifieras även i simuleringar gjorda i Simnon där man i *fig.3* kan se att systemet förblir stabilt trots att $\Theta_0^*(0)$ valts utanför konvergens intervallet (i.1).

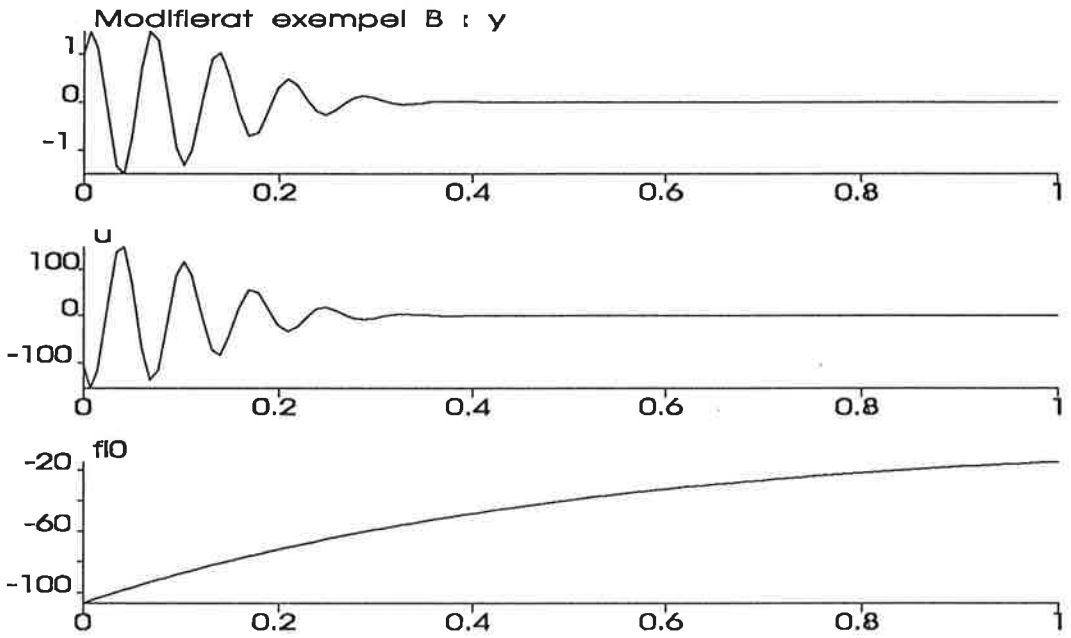


fig3. Exempel B simulerat under samma förutsättningar som i *fig1* men med $\Theta_0^*(0)=-107$, dvs utanför konvergens intervallet (i.1).

Vid en simmulering av systemet då initalvärdet valts innanför konvergens-intervallet (i.1) kan man se att den robusta algoritmen fungerar lika snabbt och bra som den icke robusta vilket kan ses i *fig.4* . Jämför *fig.2* .

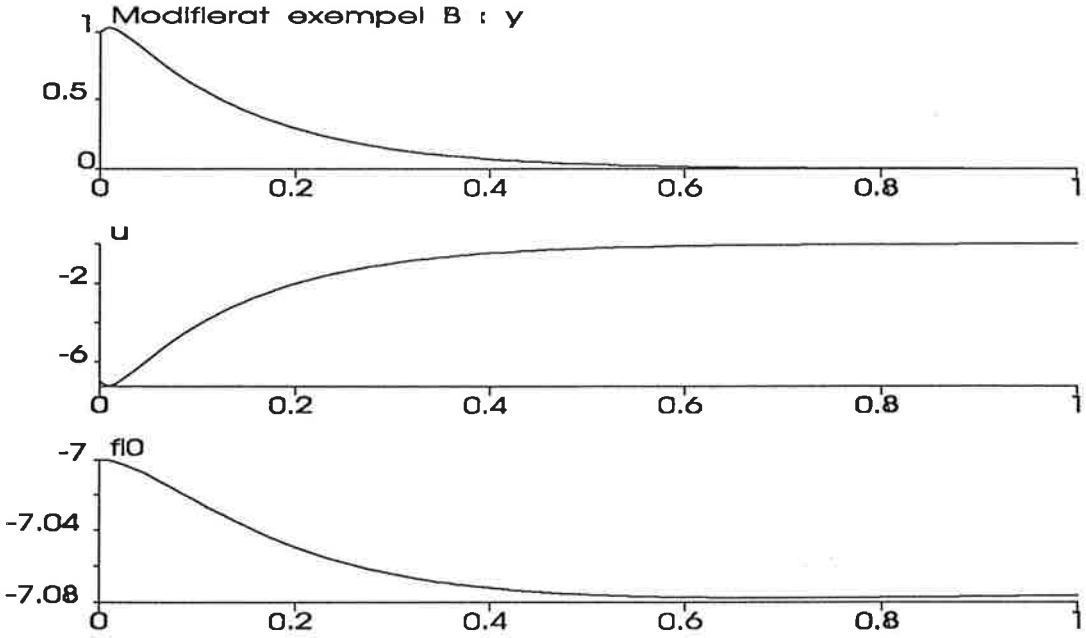


fig.4 Exempel B simulerat under samma förutsättningar som *fig.3* och med $\Theta_0^*(0)=-7$. Jämför *fig.2*

Instabilitet pågrund av snabb adaptering

Exempel C i artikeln beskriver hur snabb adaptering kan innebära instabilitet, då regulatorns design är baserad på en icke helt känd process.
I detta exempel är det ett servo-problem som behandlas. För att kunna använda den modifierade regulatorn som designades i förra exemplet används en annan process än den som anges i artikeln. Den nya processen uppvisar dock samma stabilitetsproblem. Följande process har använts:

$$\dot{x}=-ax+z-u$$
$$\mu\dot{z}=-z+2u$$
$$y=x \ . \qquad \text{(ii.1)}$$

Med följande referensmodell:

$$\dot{x}_m=-x_m+r. \qquad \text{(ii.2)}$$

En adaptiv algoritm konstruerad m.h.a stabilitetsteori (sid 126-128 i [2]) under förutsättning att $\mu=0$ ger följande parameter uppdaterings algoritm

$$u=r-S_0x$$
$$\dot{S}_0=\gamma\varepsilon x \ ; \ \varepsilon=x-x_m \ .$$

Analys av stabilitetsvillkoren för det slutna systemet då (ii.1) regleras med (i.2) (redovisade i appendix 2) ger följande krav på γ :

$$\gamma x_m^2 < \frac{\mu+1}{\mu(2+\mu)} \qquad \text{(ii.3)}$$

I *fig.5* visas en simulering där γ valts så att det slutna systemet skall vara stabilt.

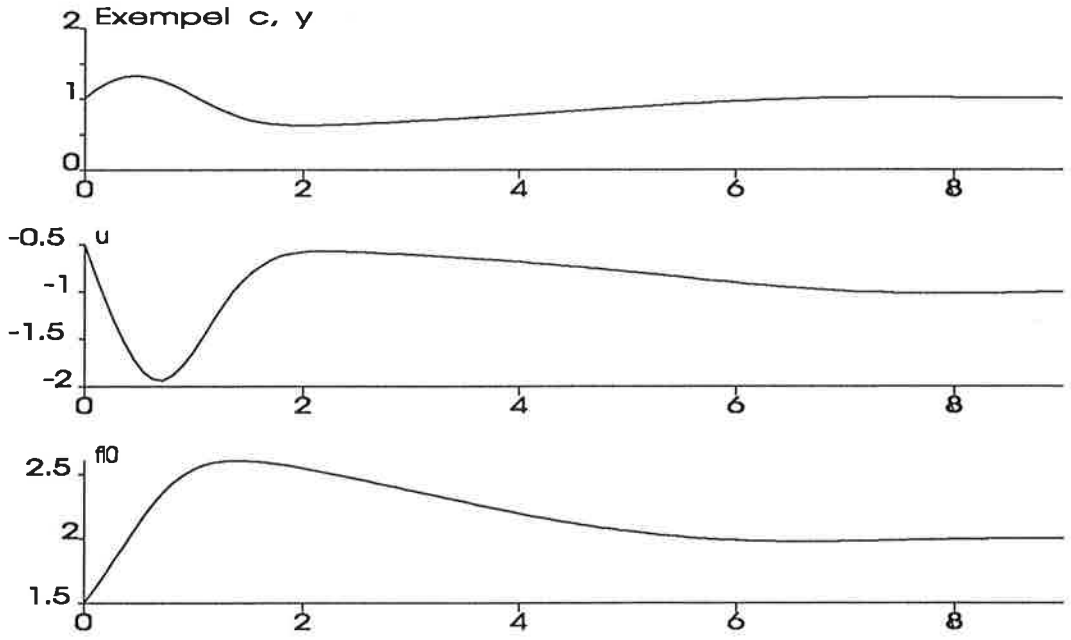


fig.5 Exempel C simulerat med $\gamma=1$, $\mu=0.1$, a=1 och r konstant 1.

Systemet simuleras även med ett γ valt så att kravet (ii.3) inte uppfylls. I *fig.6* visas en simulering av det instabila slutna systemet .

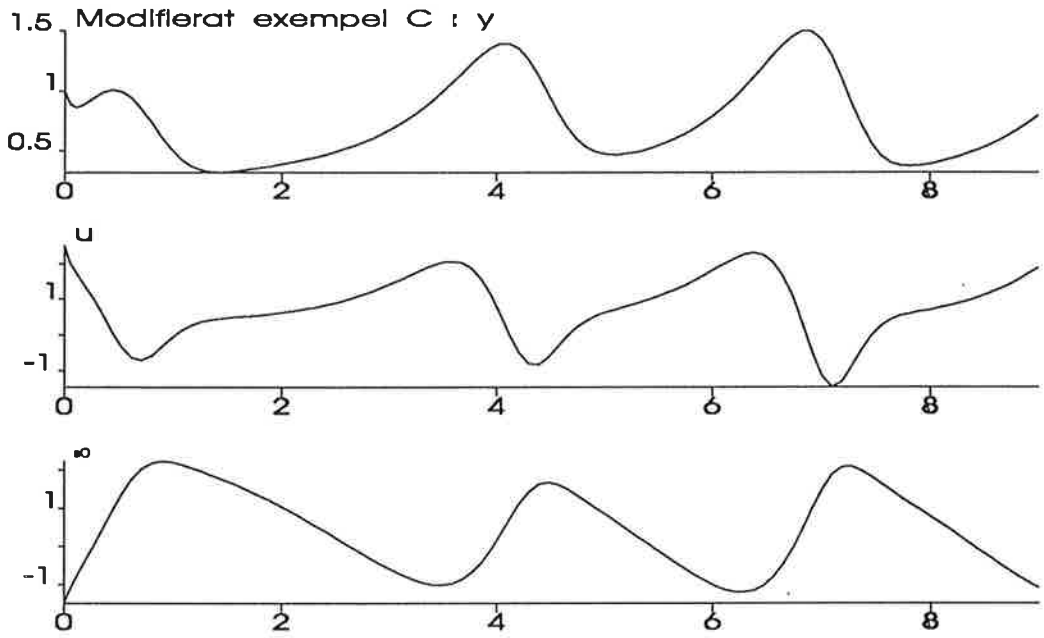


fig.6 Exempel C simulerat med γ=8, μ=0.1, a=1 och r konstant 1.

Som man kan se i *fig.5* och *fig.6* är det slutna systemets stabilitet kritiskt beroende på valet av γ.

På detta exempel appliceras den i föregående exempel designade direkta adaptiva algoritmen. Ingen direkt jämförelse kan göras av γ:s storlek, men man kan dra slutsattser angående robustheten map valet av γ. Vid simmuleringar finner man att den i artikeln föreslagna algoritmen förblir stabil trots val av stora γ. I *fig.7* visas en simulering med γ=1 som jämförelse till den omodifierde algoritmen.

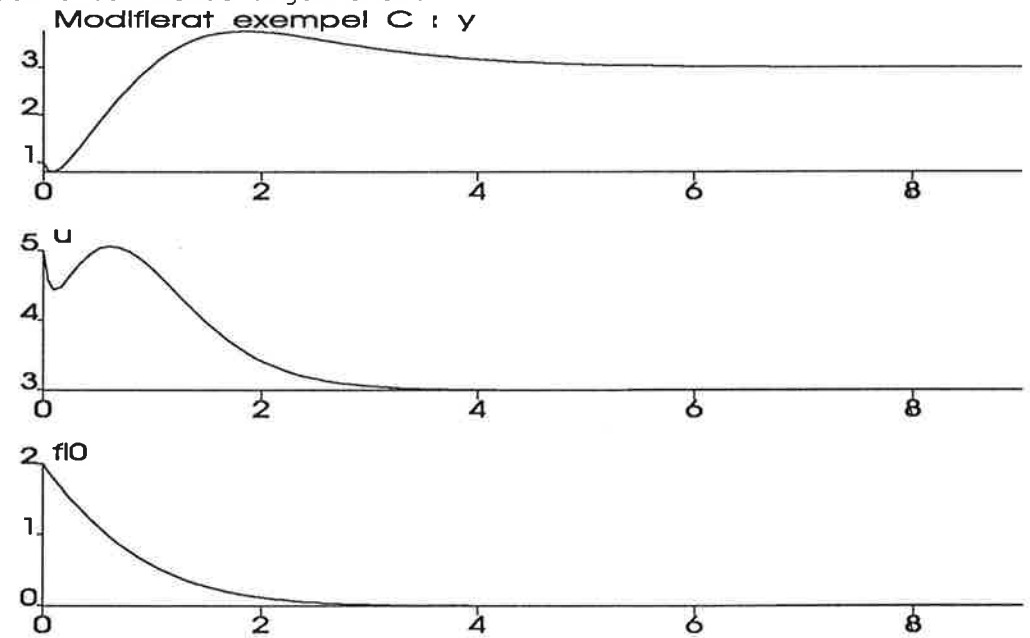


fig.7 Exempel C simulerat med γ=1, a=1, μ=0.1 och r konstant 3.

I *fig.8* visas en simulering där det slutna sytemet förblir stabilt trots att γ valts till 99. Regulatorn är alltså robust map valet av γ.

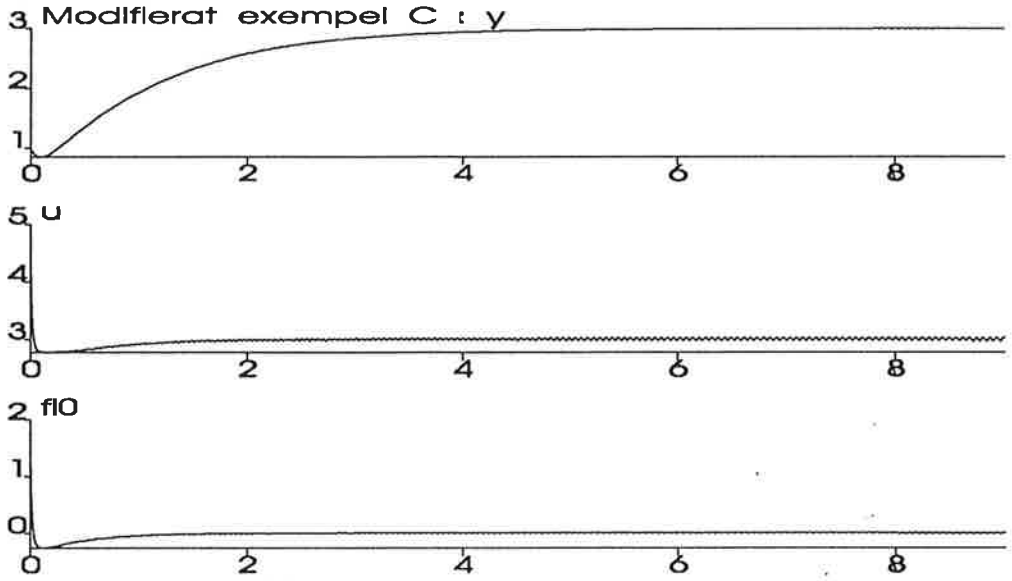


fig.8 Exempel C simulerat med γ=99, a=1, μ=0.1 och r=3.

Slutsatser

Slutsatsen efter att ha studerat artikeln "Robust Adaptive Control: A unified approach" [1] är att de föreslagna metoderna fungerar men att de kräver ingående *a priori* kunskap om processen för att bli praktiskt användbara.
Brister i artikeln är att endast STR-regulatorer behandlas och inlemmas i teorin. Vidare så behandlas inte stokastiska insignaler och störningar. Anmärkningsvärt är också att för de i exemplen specifikt utpekade problemen, som kan uppstå med adaptivreglering, inte ges någon direkt lösning i artikeln.
Det slutliga omdömet blir att artikeln skall ses som en del i den pågående forskningen kring en systematisk och enhetlig behandling av adaptiva system, precis som artikelförfattaren anger i sin slutsats.

Referenser

[1] Petros A. Ioannou och Aniruddha Datta, "*Robust adpative control : A unified Approch*", Proceedings of the IEEE, vol 79, NO. 12 december 1991.

[2] K. J. Åström och B. Wittenmark, "*Adaptive control*", Addison-Wesely, ISBN 0-201-09720-6

Appendix 1

Krav på $\Delta_m(s)$:

I] a kand.

För $\left\| \frac{a\Delta_m(s)}{s+am} \right\|_\infty < 1$

garanteras att det slutna systemet med styrlagen

$u = ay$ förblir stabilt.

$$\left\| \frac{a\Delta_m(s)}{s} \right\|_\infty = \left\| \frac{-2\mu a}{1+\mu s} \right\|_\infty = \sup_\omega \left| \frac{-2\mu a}{1+\mu j\omega} \right| = \sup_\omega f(\omega)$$

$$f(\omega) = \left| \frac{-2\mu a}{1+\mu j\omega} \right| = \frac{2\mu a}{1+(\mu\omega)^2} = \frac{2\mu a}{1+\mu^2\omega^2}$$

$f(\omega)$ symmetrisk
$f(\omega)$ strängt avtagande $\omega > 0$ $\Rightarrow$

$$\Rightarrow \sup_\omega f(\omega) = f(0) = 2\mu a \quad \Rightarrow \quad \underline{2\mu a < 1}$$

II] a okänd. Ansätter $a_m = 0$ och $\omega_m = 0$

i. $\Delta_m(s)$ analytisk i $\mathrm{Re}[s] \geq -\delta_0/2$ för något $\delta_0 > 0$

ii. $\Delta_1 = \| \Delta_m(s)/s \|_\infty^{\delta_0} \leq c$ för något $c > 0$

iii. $\Delta_2 = \| ((s+\delta_0)\Delta_m(s))/s \|_\infty^{\delta_0} \leq c\,\delta_0$ för något $c > 0$

$$\Delta_m(s) = \frac{-2\mu s}{1+\mu s}, \quad \text{pol i } -\frac{1}{\mu} \Rightarrow \Delta_m(s) \text{ analytisk för alla } s \neq -\frac{1}{\mu} \Rightarrow$$

$\Rightarrow \Delta_m(s)$ analytisk för $\mathrm{Re}[s] > -\frac{1}{\mu} \Rightarrow -\frac{\delta_0}{2} \neq -\frac{1}{\mu} \Rightarrow 0 < \delta_0 < \frac{2}{\mu}$ (*

Vilket visar att $\Delta_m(s)$ uppfyller i) om δ_0 uppfyller (*).

Appendix 2

I exempel C används systemet

$$(*)\quad \begin{cases} \dot{x} = -ax + z - u \\ \mu\dot{z} = -z + 2u \\ y = x \end{cases}$$

vilket ger upphov till samma svåra problem som det som användes i artikeln.

En adaptiv styrlag designas mha Stabilitetsteori för det förenklade systemet med $\mu = 0$.

Referensmodell:

$$\begin{cases} \dot{x}_m = -x_m + r \\ y_m = x_m \end{cases} \Rightarrow \quad y_m = \frac{1}{s+1} r$$

Styrlag:

$$u = r - s_0 x$$

$$\dot{s}_0 = \gamma x \varepsilon \quad , \quad \varepsilon = x - x_m$$

Regulatorn insatt i det verkliga systemet ger följande slutna system

$$\begin{cases} \dot{x} = -ax + z - (r - s_0 x) \\ \dot{z} = -\frac{1}{\mu} z + \frac{2}{\mu}(r - s_0 x) \\ \dot{s}_0 = \gamma x (x - x_m) \end{cases}$$

$$\bar{X} = [x \;\; z \;\; s_0]^T \qquad \dot{\bar{X}} = f(\bar{X})$$

$f(\bar{X})$ olinjär $\Rightarrow$ Förenklar problemet genom att söka stabilitet kring jämviktspunkt

Appendix 2 forts

Förenklar Linjäriserings problemet genom att anta $\begin{cases} x_m(0) = r \\ \dot{x}_m = 0 \\ r \text{ konstant} \end{cases}$

Stationära punkt ges då av:

$$\begin{cases} 0 = -ax + z - (1 - S_0 x) \\ 0 = -\frac{1}{\mu} z + \frac{z}{\mu}(1 - S_0 x) \\ 0 = \gamma x (1 - x_m) \end{cases}$$

$X^0 = [\, x^0 \quad S_0^0 \quad z^0 \,]^T = [\, x_m \quad 0 \quad 2x_m \,]$ är en lösning till (*)

Linjärisering kring X^0 ger

$$\Delta X = X - X^0$$

$$\Delta \dot{X} = \left.\frac{\partial f}{\partial X}\right|_{X^0} \Delta x + \frac{df}{dr} \Delta r \Rightarrow$$

$$\Delta \dot{X} = \underbrace{\begin{bmatrix} -1 & 1 & x_m \\ 0 & -\frac{1}{\mu} & -\frac{2}{\mu} x_m \\ \gamma x_m & 0 & 0 \end{bmatrix}}_{A} \Delta x + \begin{bmatrix} -1 \\ \frac{2}{\mu} \\ 0 \end{bmatrix} \Delta r$$

Stabil?

$$\det(sI - A) = \begin{vmatrix} s+1 & -1 & -x_m \\ 0 & s+\frac{1}{\mu} & \frac{2}{\mu} x_m \\ -\gamma x_m & 0 & s \end{vmatrix} = s^3 + s^2\left(1 + \frac{1}{\mu}\right) + s\left(\frac{1}{\mu} - \gamma x_m^2\right) + \gamma \frac{x_m^2}{\mu}$$

Hurwitz kriterie: $F(s) = a_0 s^3 + a_1 s^2 + a_2 s + a_3$

Alla rötter till $F(s)$ har negativ realdel om

I) $a_i > 0$

II) $a_3 > 0$

III) $a_1 a_2 > a_0 a_3$

Här I) $\Rightarrow 1 + \frac{1}{\mu} > 0$ ok! ty $\mu > 0$

II) $\Rightarrow \frac{\gamma x_m^2}{\mu} > 0$ ok! ty $\gamma > 0$

III) $\Rightarrow \quad \gamma x_m^2 < \frac{\mu + 1}{\mu(2 + \mu)}$

Appendix 3.

Simnon programlistningar

Simnon program 1

```
Processen i exempel B

CONTINUOUS SYSTEM procB
" Andra ordningens process (Exempel B sid 1737)
"   .
"   x=ax+z-u
"   .
" mz=-z+2u
"
"   y=x

INPUT  u
OUTPUT y

STATE x    z
DER   dx  dz
TIME t

" System dynamik ------------------------------------------------
dx=a*x+z-u
ddz=(1/m)*(-z+2*u
y=x
" Initiala parameter värden --------------------------------
a : 1
x : 1
m : 0.1
END
```

Simnon program 2

Adaptiv regulator enligt MIT-regeln

```
CONTINUOUS SYSTEM regB
" Adaptiv regulator i exemlep B (Sid 1737)
" baserad på förenklad modell
"
" u=-kx
" .
" k=gamma*eps1*x
"
" eps1=x
INPUT  y
OUTPUT u

STATE k
DER   dk

" System dynamik ----------------------------------------
x=y
eps1=x
dk=IF k<1025 THEN gamma*eps1*x ELSE 0
u=-k*x
" Initiala parameter värden ---------------------------------
gamma : 1
k     : 100
END
```

Simnon program 3

```
Modifierad direkt-STR .

CONTINUOUS SYSTEM regcm
" Direkt adaptiv regulator enligt
" Gradient metoden.    (6.8 sid 1761)
"  .
" fi0= -gamma*eps*fz-gamma*fi0*omega
"   .
" fz= -fz+y
"   .
"  W=-W+u
"    .
"  ms=-delta0*ms+abs(u)^2+abs(y)^2
"
"        fi0*fz-z
"  eps= -----------
"           m^2
"
"  z= -y + W
"
"  m^2=alfa+beta*ns^2
"  ns^2=ms
"  u = fi0*y+r

INPUT  y r
OUTPUT u

STATE fi0   fz    W    ms
DER   dfi0  dfz   dW   dms

dfi0 = -gamma*eps*fz-gamma*fi0*omega
dfz  = -fz+y
dW   = -W+u
dms=-delta0*ms*abs(u)^2+abs(y)^2

eps=(fi0*fz-z)/(m2)
z= -y+W
m2=alfa+beta*ns2
ns2=ms

"********* Styrlagen
u=fi0*y+r
"********* Parametrar
alfa   : .1
beta   : .9
delta0 : 0.01
omega  : 0.98
gamma  : 1

fi0  :2
fz   :0
W    :0
ms   :0
end
```

Simulering av en "försiktig" adaptiv regulator

Lars Arvastson, F-88
och
Thomas Idoffsson, F-88

13 maj 1992

Sammanfattning

Denna rapport behandlar insomnningseffekter på en försiktig regulator. Projektet ingår som en obligatorisk del av kursen Adaptiv Reglering vid Institutionen för Reglerteknik, LTH.
Vi simulerar en första ordningens process med olika parameterval i Simnon. Härur kan vi se hur insomningseffekten beror av processen, hur ofta och hur länge regulatorn sover för olika processer. Vi studerar speciellt beteendet kring stabilitetsgänsen.
Slutsatsen är att regulatorn kan somna både med stabila och instabila processer. För instabila är det dock ovanligt, medan det inträffar ofta för stabila processer. Tiden som regulatorn sover beror också på processen, för en instabil process vaknar regulatorn snabbare jämfört med en stabil.

Dual reglering

Syftet med dual reglering är att uppnå en kompromiss mellan bra reglering och bra estimering. Detta uppnås varken med MRAS eller STR eftersom god estimering kräver att processen exciteras tillräckligt, vilket dessa regulatorer inte tar hänsyn till. Följden kan bli att kovariansmatrisen i Kalmanfiltret växer och osäkerheten ökar, när systemet är dåligt exciterat.
Om trajektorian för reglervärdet är känd kan man genom att minimera en förlustfunktion (Bellman ekvationen) ett visst antal steg framåt i tiden uppnå dual reglering. Detta kräver emellertid så mycket beräkningsarbete att det i praktiken ej går att lösa det.
Om man begränsar sig till att minimera endast ett steg framåt blir räkningarna enkla. Detta leder till sk försiktig reglering (eng. Cautious Control) Problemet är bara att denna regulator kan råka ut för sk insomning. Detta innebär att regulatorn slutar skatta parametrarna samt ger en styrsignal mycket nära noll.

Processen

Vi har använt en process med två parametrar, a och b,

$$y(t+1) = ay(t) + b(t+1)u(t) + e(t+1)$$

där y är utsignalen, u är insignalen och e är vitt brus. Bruset antas vara normalfördelat med väntevärdet noll och variansen R_2. Det gäller även att $e(t)$ är oberoende av $y(t-1), y(t-2)$,..., $u(t-1), u(t-2)$,..., $a(t), a(t-1)$,..., $b(t), b(t-1)$,.... Vi låter a vara en känd konstant och b variera enligt

$$b(t+1) = \Phi b(t) + v(t)$$

$v(t)$ är normalfördelat brus med väntevärdet noll och variansen R_1.

Processkattning

Parametern b skattar vi med ett Kalmanfilter, som får ett enkelt utseende därför att alla variabler är skalärer.

$$\hat{b}(t+1) = \Phi\hat{b}(t) + K(t)(y(t) - ay(t-1) - u(t-1)\hat{b}(t))$$

$$P(t+1) = (\Phi - K(t)u(t-1))P(t)\Phi + R_1$$

$$K(t) = \frac{\Phi P(t)u(t-1)}{R_2 + u(t-1)^2 P(t)}$$

Certain Equivalence (CE) reglering

Den optimala styrsignalen, om parametrarna vore kända, är

$$u(t) = \frac{y_r(t+1) - ay(t)}{b(t+1)}$$

CE reglering fås då b ersätts med motsvarande skattning, dvs

$$u(t) = \frac{y_r(t+1) - ay(t)}{\hat{b}(t+1)}$$

Med denna regulator undviker man fenomenet insomning, istället får man problem då $\hat{b} \approx 0$, styrsignalen blir då extremt känslig för skattningsfel. Därför vill vi även ta hänsyn till hur bra vår skattning är, vilket vi gör med försiktig reglering.

Försiktig reglering

Här strävar vi efter att minimera väntevärdet av det förväntade reglerfelet. Vi vill alltså minimera förlustfunktionen

$$E[(y(t+1) - y_r(t+1))^2 \mid Y_t, U_t]$$

Där Y_t är alla gamla y fram till tiden t och U_t är alla u fram till tiden t. $y(t+1)$ givet $y(t)$ och $u(t)$ är normalfördelad, ty

$$y(t+1) = ay(t) + b(t+1)u(t) + e(t+1)$$

där $ay(t)$ är känd, $b(t+1)u(t)$ är normalfördelad och $e(t+1)$ är också normalfördelad. Eftersom vi har en normalfördelning kan vi utnyttja att $E(X^2) = E(X)^2 + var(X)$. Förlustfunktionen blir nu

$$[ay(t) + \hat{b}(t+1)u(t) - y_r(t+1)]^2 + R_2 + u^2(t)var(\hat{b})$$

Vi deriverar och sätter derivatan till noll för att finna minimat.

$$\frac{\partial E}{\partial u} = 2\hat{b}(t+1)[ay(t) + \hat{b}(t+1)u(t) - y_r(t+1)] + 2var(\hat{b})u(t) = 0$$

Den optimala styrsignalen fås nu som

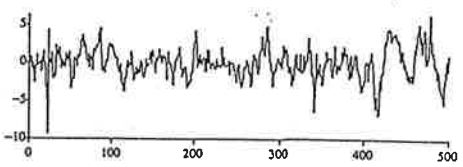
$$u(t) = \frac{\hat{b}(t+1)y_r(t+1) - a\hat{b}(t+1)y(t)}{\hat{b}(t+1)^2 + var(\hat{b})}$$

Skattningen av b och dess varians får vi med Kalmanfiltret, där P(t) är skattningens varians.

Insomning

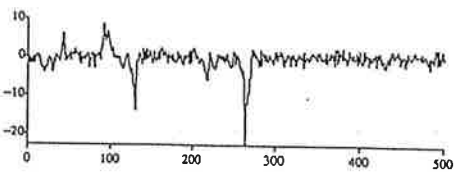
Problemet med insomning uppkommer under vissa förutsättningar. En av dessa är att parametern b måste vara nära noll, så att $\hat{b}$ blir nära noll. För att b skall bli nära noll lagom ofta, låter vi b variera enligt ovan givna formel med $\Phi = 0.98$. Vid simuleringarna jämför vi två processer, $a = 0.9$ och $a = 1.1$, dvs en stabil och en instabil process. För att göra det så lätt som möjligt säger vi att $y_r = 0$. Resultaten nedan är inte representativa eftersom det är avsevärt enklare att få regulatorn till en stabil process att somna jämfört med en instabil, vilket inte framgår av de två simuleringarna.

Stabil process:

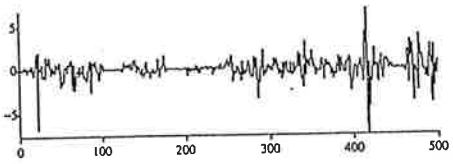


Processen y

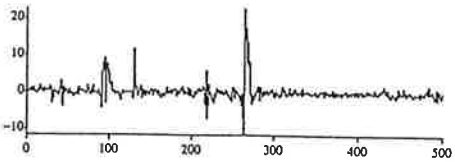
Instabil process:



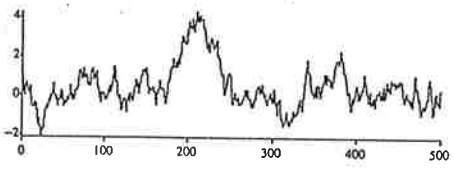
Processen y



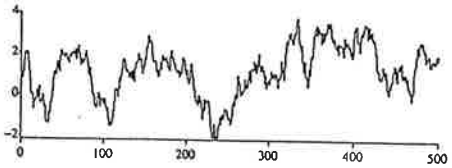
Styrsignalen u



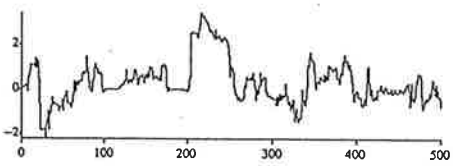
Styrsignalen u



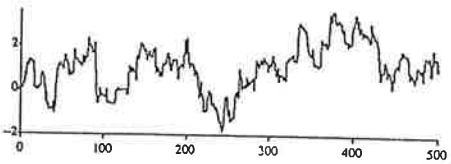
Parametern b



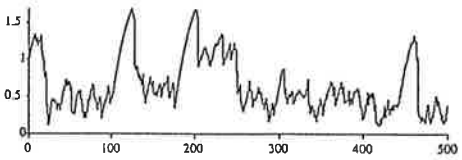
Parametern b



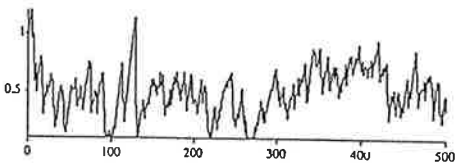
Skattad parameter $\hat{b}$



Skattad parameter $\hat{b}$



Variansen P



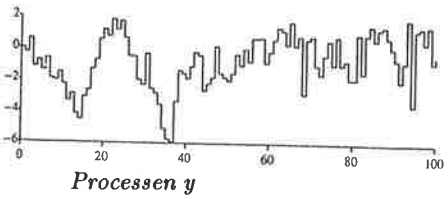
Variansen P

En regulatorn sover när både u och $\hat{b}$ är mycket små. Tittar vi först på den stabila regulatorn ser vi att den sover vid ett par tillfällen, det syns då tydligt hur variansen P snabbt växer. Någon nämnvärd försämring av regleringen syns dock inte eftersom systemet är dämpat. Det instabila systemet sover bara vid ett tillfälle, men tittar vi på y ser vi att vi har stort reglerfel vid två tillfällen. Den första toppen kommer av att regulatorn sover medan den andra beror på att regulatorn har ett litet P samtidigt som $\hat{b}$ är liten.

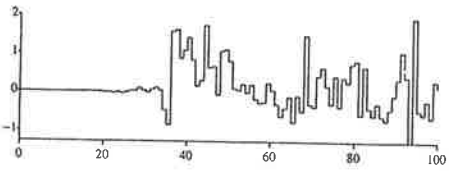
Uppvaknande

För att lättare kunna jämföra hur länge regulatorn sover för de olika processerna ger vi $\hat{b}$ ett så litet startvärde att regulatorn sover från början. Bilderna nedan visar några typiska simuleringar. Den instabila processens regulator vaknar vanligtvis två till tre gånger snabbare än den stabila, i båda fallen sker uppvaknandet mycket snabbt, när det väl har börjat. Simuleringarna visar att regulatorn sover kortare tid för instabila processer jämfört med stabila. Att det är svårt att få regulatorn till ett instabilt system att somna beror inte, som vi ser av simuleringarna, på att den vaknar på en gång. Anledningen är troligen istället att det instabila systemet är bättre exciterat jämfört med det stabila.

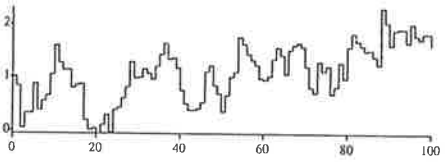
Stabil process:



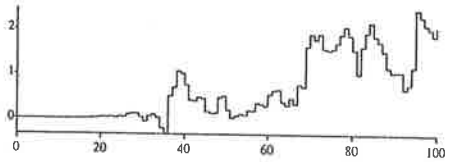
Processen y



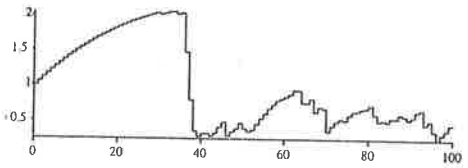
Styrsignalen u



Parametern b

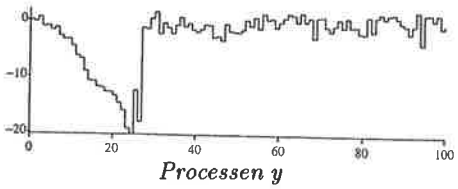


Skattad parameter $\hat{b}$

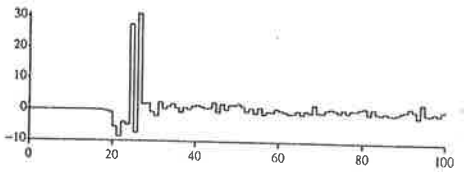


Variansen P

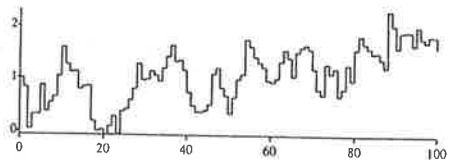
Instabil process:



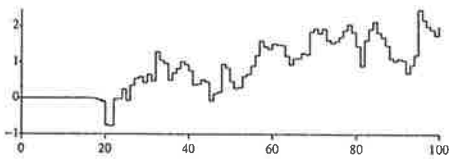
Processen y



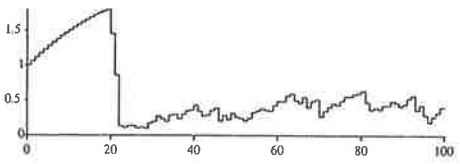
Styrsignalen u



Parametern b



Skattad parameter $\hat{b}$



Variansen P

Slutsatser

Vi har konstaterat att en regulator som reglerar en första ordningens process kan somna för såväl stabila som instabila processer. Den stabila processens regulator somnar dock betydligt oftare än vad den instabila gör, dessutom sover regulatorn för den stabila processen längre än den instabila. Att regulatorn till ett instabilt system somnar så sällan beror inte på att den vaknar meddetsamma. Det beror troligtvis istället på att den instabila processen är bättre exciterad jämfört med den stabila.

Simnonkod

```
discrete system proreg
input  e1 e2
state   z  b  x  y1  u1  P
new    nz nb nx ny1 nu1 nP
time t
tsamp ts
"PROCESS
nb=fi*b+e1
nz=a*z+b*u+e2
y=z
"REGULATOR
ny1=y
nu1=u
nx=fi*x+K*(y-a*y1-u1*x)              "Estimering
nP=(fi-K*u1)*P*fi+R1
K =fi*P*u1/(R2+u1*P*u1)
u=-a*x/(x*x+P)*y                     "Cautious control
"u=-a/x*y                            "Certainty equivalence control
ts=t+h
h: 1
a:0.6
fi:1
R1:0.1
R2:1
end
```

```
discrete system noise1
output e1
tsamp ts
time t
e1=sqrt(R1)*norm(t)
ts=t+h
h:1
R1:0.1
end
```

```
discrete system noise2
output e2
tsamp ts
time t
e2=sqrt(R2)*norm(t)
ts=t+h
h:1
R2:1
end
```

```
connecting system con
time t
e1[proreg]=e1[noise1]
e2[proreg]=e2[noise2]
h:1
end


macro do
syst proreg noise1 noise2 con
store y[proreg] u[proreg] x[proreg] P[proreg] b[proreg]
split 2 1
par a:1.1
par fi:1
init x[proreg]:0.1
init b[proreg]:1
init P[proreg]:1
init z[proreg]:0
simu 0 1000
ashow y
ashow u
end
```

Adaptiv reglering av tappvarmvatten

Utfört av::
Jan Johansson, Teknolog vid LTH
Mats Wennberg, Teknolog vid LTH

Handledare:
Tore Hägglund, Institutionen för Reglerteknik
Karl-Johan Åström, Institutionen för Reglerteknik
Anders Wallenborg, Tour-Andersson

I kursen adaptiv reglering har det under en vecka, i april 1992, utförts ett kortare samarbetsprojekt mellan Tour-Andersson och institutionen för Reglerteknik. Projektet har haft till syftet att undersöka dagens PI-reglering vid uppvärmning av tappvarmvatten och speciellt studera om en adaptiv regulator är lämplig. All dynamik har konstaterats härröra från ställdonet och att där är variationer i den statiska förstärkningen. Vi har ett förslag på en bra adaptiv regulator.

Problemformulering

I figur 1 visas en schematisk bild på en tappvarmvattenkrets, som används för att förse t.ex. ett hyreshus med varmvatten. Kretsen består av en vatten-vattenvärmeväxlare med tillhörande reglerutrustning.

Målet med regleringen är att hålla den utgående temperaturen, T4, på sekundärsidan konstant, trots variationer i belastningen. Detta åstadkommes genom att variera fjärrvärmevattenflödet på värmeväxlarens primärsida. Flödet regleras med hjälp av en ventil, SV1. I vårt fall var börvärdet på tappvarmvattentemperaturen 50° C.

Vid reglering av denna processtyp så måste följande beaktas.

- Där är stora stokastiska belastningsändringar medan börvärdet är konstant.
- Långsamma ställdon, 60 s gångtid, i förhållande till dynamiken i processen.
- Endast temperatursignalen är tillgänglig för mätning, det är för dyrt med flödesmätningar.
- Processen är olinjär.
- Viktigt med få och enkla parameterinställningar.
- Vattenförbrukningen är låg under stora delar av dygnet, vilket ger dålig excitation av processen.

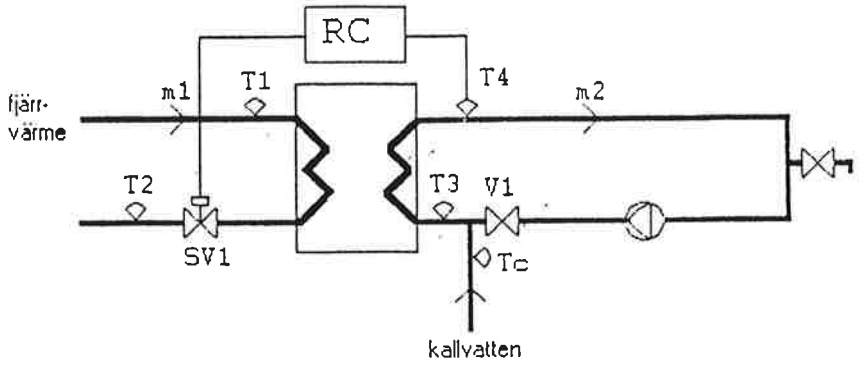
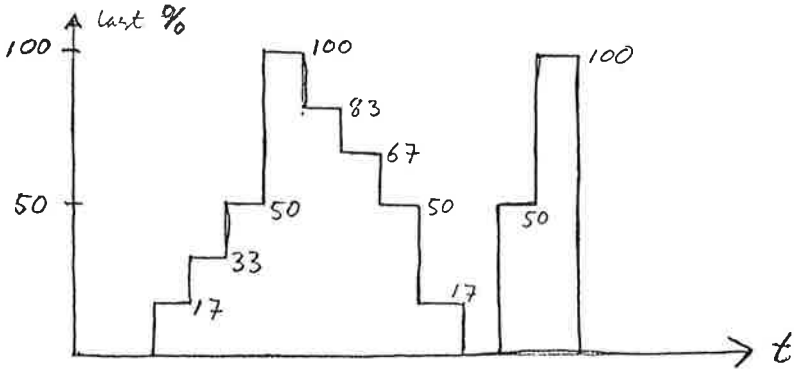


Fig .1. Tappvarmvattenkrets. T1 = temperatur på inkommande fjärrvärme, T2 = temperatur på returen till fjärrvärmenätet, T3 = inkommande temperatur på sekundärsidan, T4 = tappvarmvattentemperatur, Tc = kallvattentemperatur, m1 = massflöde på primärsidan, m2 = massflöde på sekundärsidan. SV1 är en reglerventil med tillhörande ställdon och V1 är en backventil. RC betecknar regulatorn.

- Ventilen har ett minsta reglerbart flöde, vilket leder till on-off reglering som kan ge stabilitetsproblem vid låg last.
- Svårt att trimma regulatorn manuellt vid frekventa laststörningar.

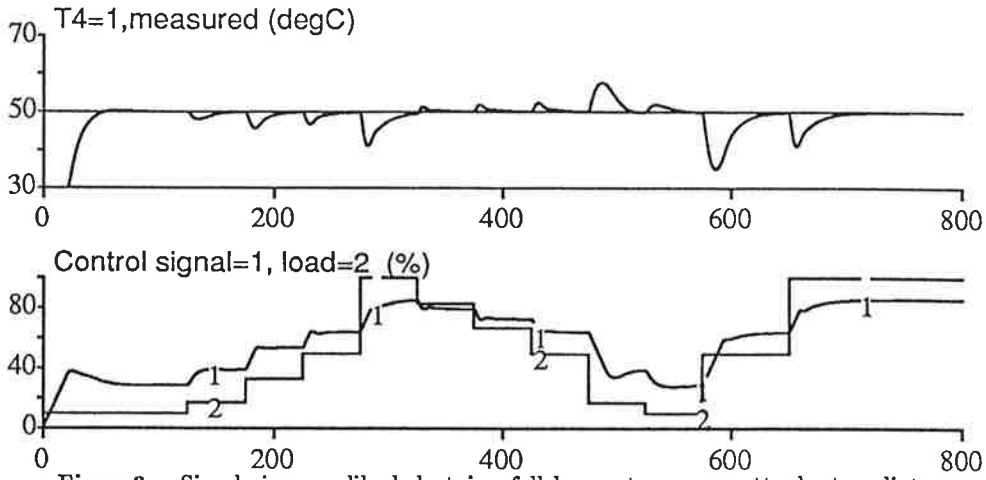
Samtidigt som man är medveten om dessa problem så är där en del prestandakrav som skall uppfyllas. Många av dessa krav är fastlagda enligt svensk byggnorm:

- Man tillåter ett stationärt reglerfel som är $\leq 2°$ C.
- Maximalt reglerfel = $10°$ C vid 50 % belastningsändring.
- Maximal amplitud hos stationär temperatursvängning = $2°$ C då man har 10-30 % belastning på processen. Över 30 % tillåts ingen självsvängning.
- Reglerfelet $< 2°$ C inom 2 minuter från belastningsändring.
- Klara av en belastningsprofil enligt finska provnormer, se figur 2.



Figur 2. Belastningsprofilen enligt finska normer

Problemen som föreligger gör det lämpligt att undersöka om man kan



Figur 3. Simulering av olika belastningsfall hos en tappvarmvattenkrets enligt en finsk laststörningsprofil då man använder en PI-regulator. De uppkommna temperaturvariationerna ses i den övre bilden.

använda sig av en adaptiv metod för att reglera processen.

Simnon-modellen

På TA har man tagit fram simnonprogram för att kunna simulera ett tappvarmvattensystem för 10 hushåll. Vi fick ta del av dessa simuleringprogram för att prova ut en adaptiv regulator. I den modellen som fanns reglerades processen med hjälp av en PI-regulator, som bra klarar av de ovan ställda kraven vid små och medelstora laständringar. Figur 3 visar en simulering då man har lagt på en finsk laststörniningsprofil.

Det syns att man får väldiga temperaturfall vid stora laständringar. Temperaturfallet beror främst på att ställdonet är långsamt och ej hinner med de snabba ändringarna i belastningen men också att regleringen är långsam vid hög last.

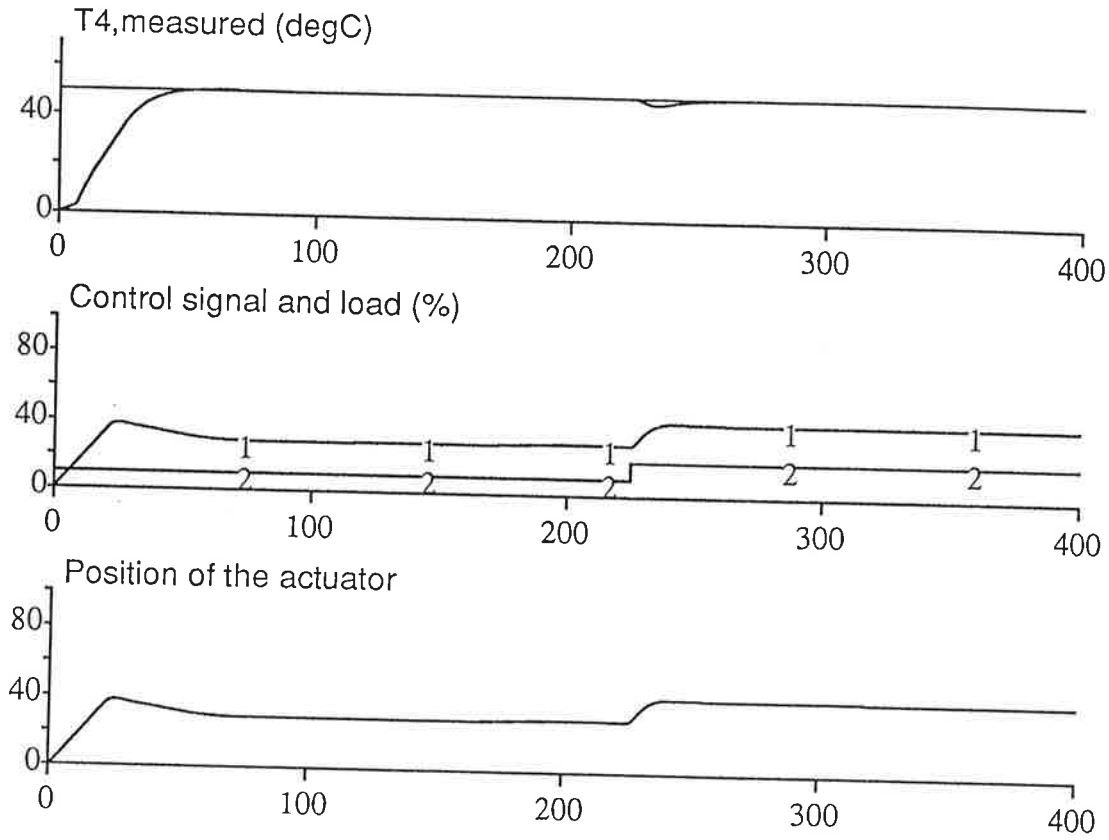
ANALYS

Tidskonstanten

För att kunna analysera modellen var vi tvungna att simulera hur processen uppförde sig vid olika driftsförhållanden. Vi beslöt oss för att börja med att undersöka tidskonstanterna i modellen, dels hos värmeväxlaren och dels hos ställdonet. Det visade sig att värmeväxlaren svarade momentant på flödesändringar medan det fanns en fördröjning i ställdonet på ca 1 sekund, från det styrsignal kom till dess att ställdonet svarade. Således var tidskonstanten densamma vid olika belastningsfall.

Statisk förstärkning

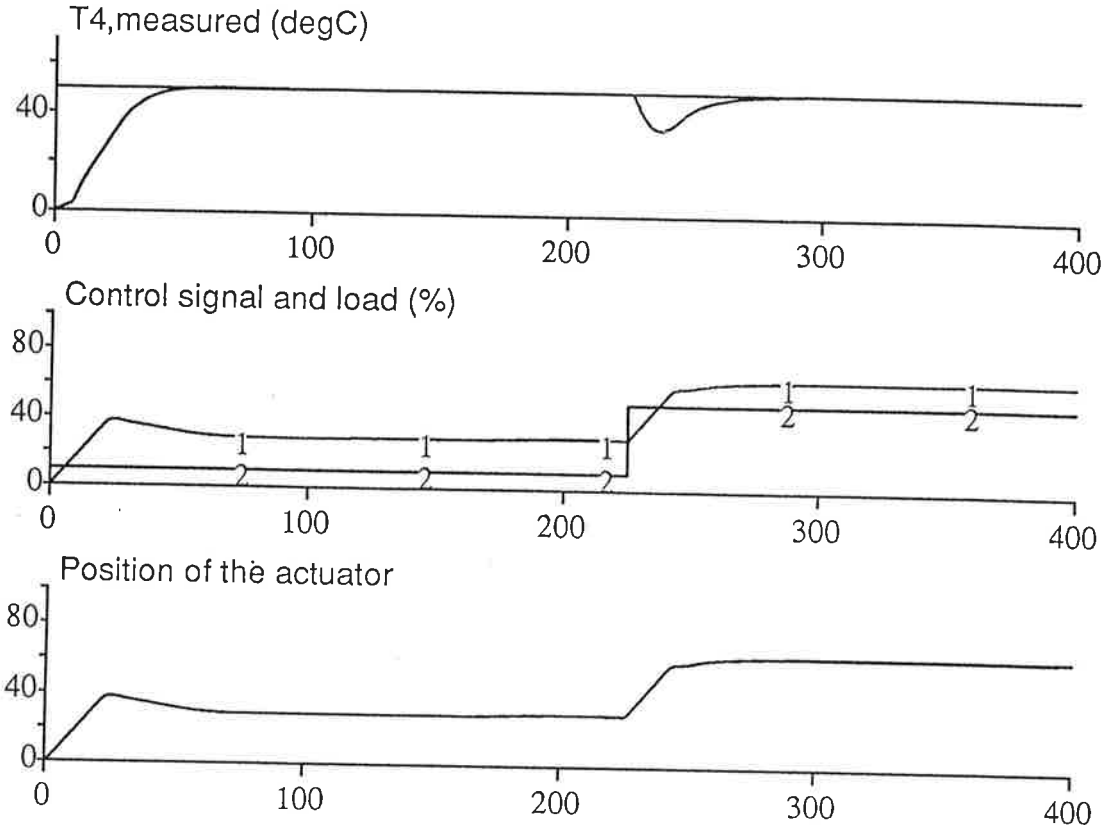
Vi undersökte också hur processen betedde sig då vi från en minimibelastningsnivå på 10% ökade belstningen till 20%, 50% samt 100%. Figur 4, 5



Figur 4. Kontroll av tappvarmvattenkretsens uppförande vid liten belastningsändring

och 6 visar simuleringarna i dessa fall med TA's regulator inkopplad och ett ställdon med 60 sekunders gångtid.

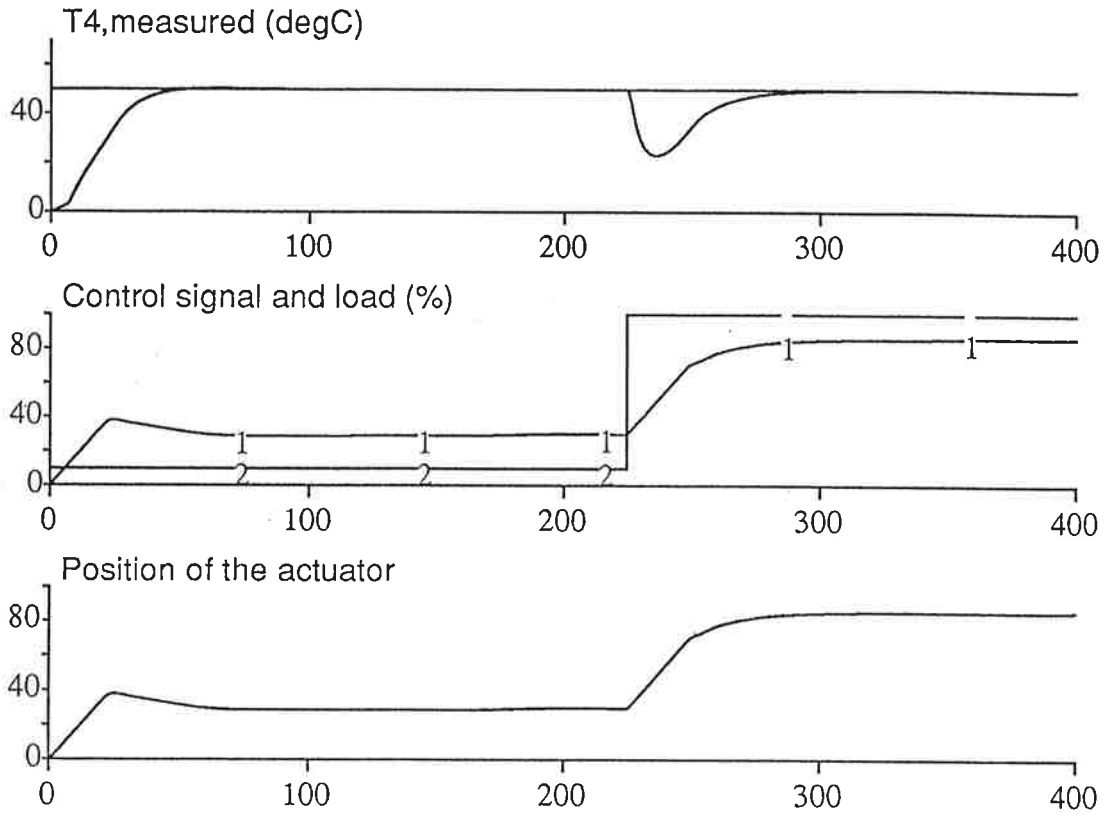
Ur dessa bilder kan ses att förstärkningen förändrades vid olika belastningar. Förhållandet mellan ställdonsposition och last beräknades. Vi tolkade kvoten som ett mått på processens förstärkning. Denna kvot var 3 vid den minimala lasten 10% och minskade sedan till 2.1 vid 20% last, 1.3 vid 50% last och 0.86 vid 100% last. Detta tydde på att processens statiska förstärkning var mindre vid höga laster. För att undersöka detta ännu närmare gjordes simulering, där styrsignalen varierades stegvis utan last. I figur 7 ses att det blir allt mindre förändring i tappvarmvattentemperaturen T4, trots att styrsignalen ökas med lika stort steg hela tiden. Observera dock att T4 aldrig kan bli större än primärvattentemperaturen, vilken i vårt fall är 65° C.



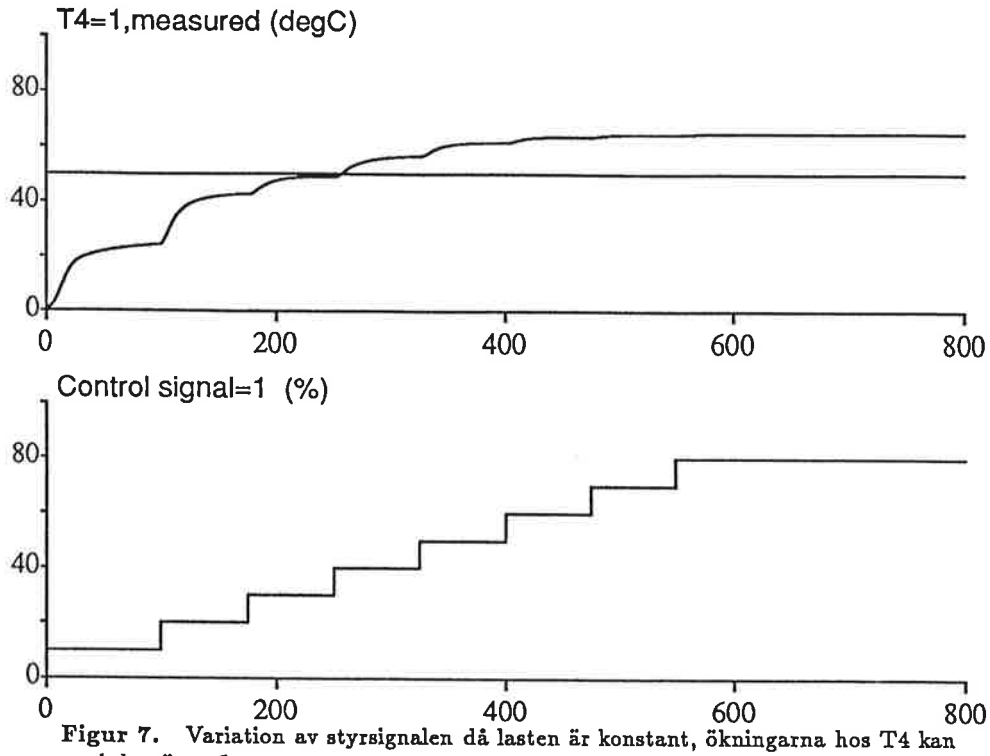
Figur 5. Kontroll av tappvarmvattenkretsens uppförande vid en stor belastningsändring

Olinjäriteter

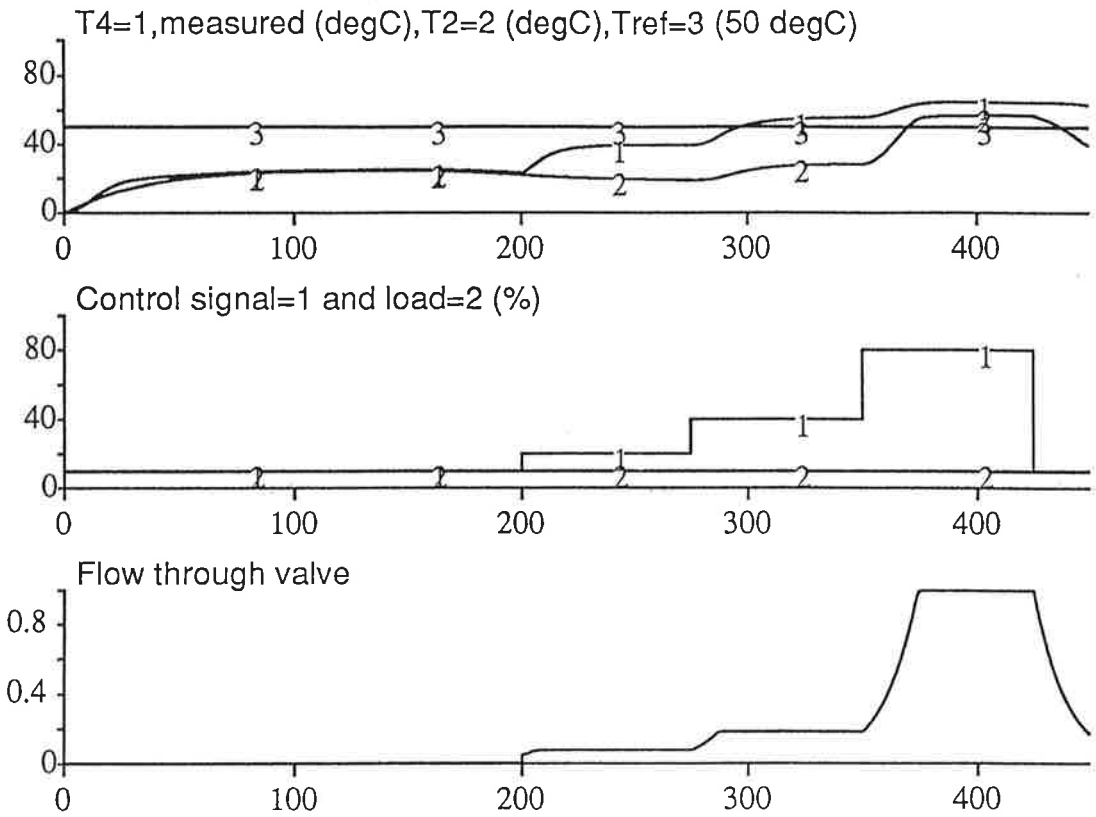
Simnon-koden för systemet undersöktes nu närmare för att se om denna olinjäritet kunde finnas tillgänglig där. Simulering gjordes där flödet genom ventilen förändrades, då vi hade en konstant last men ändrade styrsignalen, se Figur 8. Här syns det tydligt att för låga laster, förändrar sig flödet linjärt genom ventilen, medan det för höga laster är en exponentiell förändring.



Figur 6. Kontroll av tappvarmvattenkretsens uppförande från liten- till full belastning



Figur 7. Variation av styrsignalen då lasten är konstant, ökningarna hos T4 kan ses i den övre figuren.



Figur 8. Varierande styrsignal och konstant last för att kunna se olinjäriteten hos ventilen SV1.

I Simnon-koden fanns detta exponentiella beteende hos ventilen med. Detta uppförande hos ventilen innebär att det för höga laster ej behövs stora styrsignaler till ställdonet för att få stora förändringar i T4. Vid närmare undersökning av koden för värmeväxlaren hittades ytterligare en olinjäritet i form av att värmeövergångskoefficienterna för primär- och sekundärsidan var av en potens < 1, av respektive massflöde. Detta förklarar figur 7 bra. I och med detta icke-linjära beteende hos processen bekräftades vår föraning om att en adaptiv regulator, för att reglera processen, kunde var lämplig.

MODELLBYGGE

Vi ställde nu upp en modell, där tappvarmvattentemperaturen, T4, antogs bero på ställdonets position, här kallad Pos.

$$T4 = K * G * Pos + bias$$

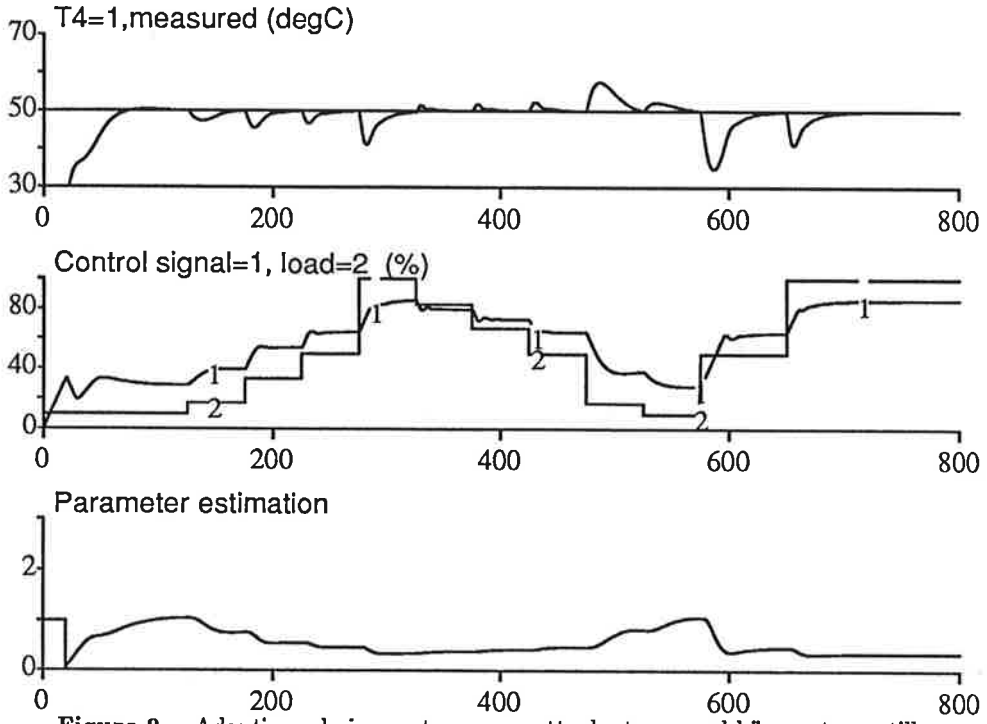


Figure 9. Adaptiv reglering av tappvarmvattenkretsen, med hänsyn tagen till en biasnivå

där $K*G$ är processens överföringsfunktion och G(0)=1. För att vara säker på att några höga frekvenser ($> 1Hz$) ej fanns med, lågpassfiltrerades T4 och Pos. Simnonkoden för lågpassfiltret återfinns i Bilaga 1. Efter lågpassfiltreringen såg modellen ut som,

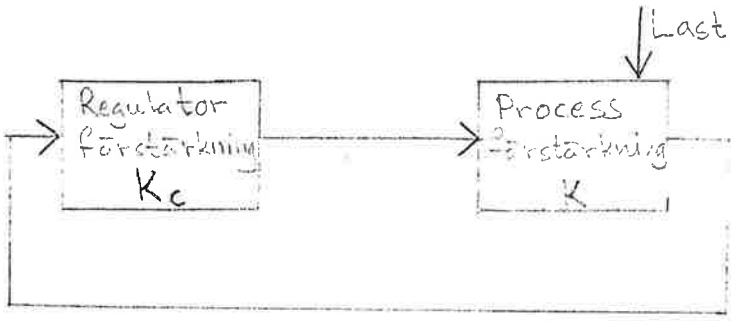
$$T4 = K * Pos + bias$$

K är processens förstärkning och är alltså den parameter vi vill kunna skatta. Biastermen var vi tvungna att ha med, för vi kunde ej vara säkra på att T4=0 då $Pos = 0$. Bias-termen kunde ej filtreras bort, då processen i stort sett alltid var dåligt exciterad (ingen last under stora delar av dygnet) och excitationen fanns i huvudsak vid de lägre frekvenserna ($< 0.01Hz$). Bias-termen valdes till 20, vilket motsvarar omgivningens temperatur.

Skattning och Simulering

För att skatta processens förstärkning K, använde vi oss av en rekursiv minsta-kvadratskattare med glömskefaktor=0.99. Simnonkoden för denna skattare återfinns i Bilaga 2. I Figur 9 ses simuleringen av hur den skattade processförstärkningen varierar. De 100 första sekunderna är ett uppstartsförlopp, men sedan fås ett jämnt och fint uppförande hos den estimerade förstärkningen.

Vid den minimala lasten på 10 %, ses att processens förstärkning skattats till ca 1. Vad gäller regulatorn, som TA använder sig av, har den en förstärkning KC=3. Det återkopplade systemet kan därför skrivas enligt



Då processens förstärkning skattats till 1, vid den minimala lasten (10 %), fås sambandet

$$KC * K = 3 * 1 = 3$$

Den adaptiva regulator, som nu användes var densamma som TA hade tagit fram (PI-regulator), med den modifikationen att regulatorns förstärkning ej var konstant utan berodde på den skattade processens förstärkning enligt sambandet $KC_{ny} = 3/K$, där KC_{ny} är den adaptiva regulatorns nya förstärkning och K processens skattade förstärkning. Detta ger att vid höga laster, då processen har en liten förstärkning, vrids regulatorns förstärkning upp för att kompensera för processens lägre förstärkning. Vidare ses i Figur 9 resultatet av simuleringen med den adaptiva regulatorn inkopplad. Ställdonet har 60 sekunders gångtid. Integratordelens återställningstid har gjorts 10 ggr mindre än i det ursprungliga programmet, för att förhindra att ställdonsventilen står och slår allt för mycket. Vid jämförelse mellan Figur 9 och Figur 3 ses ett likartat uppförande. Den adaptiva regulatorn är dock litet snabbare vid belastningsökningar men litet långsammare vid belastningsminskningar än TA:s regulator med sin konstanta förstärkning.

Vi prövade också en modell av processen, som ej innehöll en bias-term. I Figur 10 syns skattningen av processens förstärkning, vilken nu är ca 2 vid den minimala lasten 10 %. Detta gav nu, enligt ovan, en adaptiv regulatorförstärkning på $KC_{ny} = 6/K$. I Figur 10 visas också simuleringen, vilken uppvisar ett likartat uppförande både vad gäller T4 och styrsignal till ställdonet vid jämförelse med Figur 3 och Figur 9.

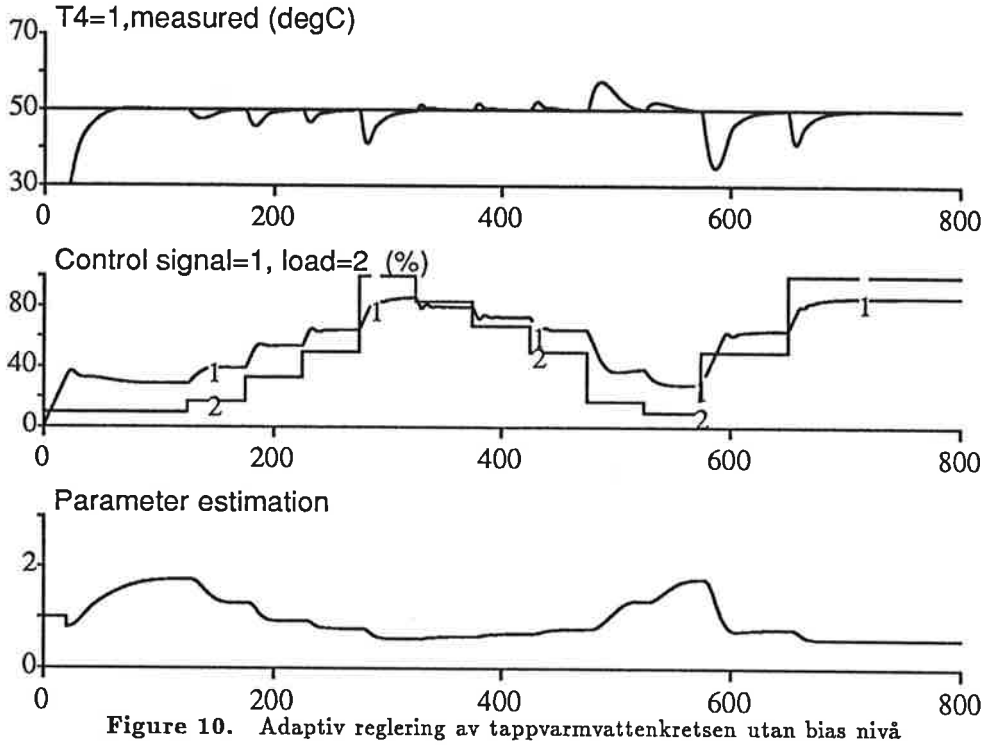


Figure 10. Adaptiv reglering av tappvarmvattenkretsen utan bias nivå

Som avslutning provades ett ställdon med 10 sekunders gångtid, dels på TA:s regulator och dels på den adaptiva regultorn, baserad på modellen med bias-term. Simuleringen med TA:s regulator syns i Figur 11. I Figur 12 ses simuleringen med den adaptiva regulatorn, vars återställningstid för integratordelen gjorts 10 ggr mindre av samma skäl som tidigare.

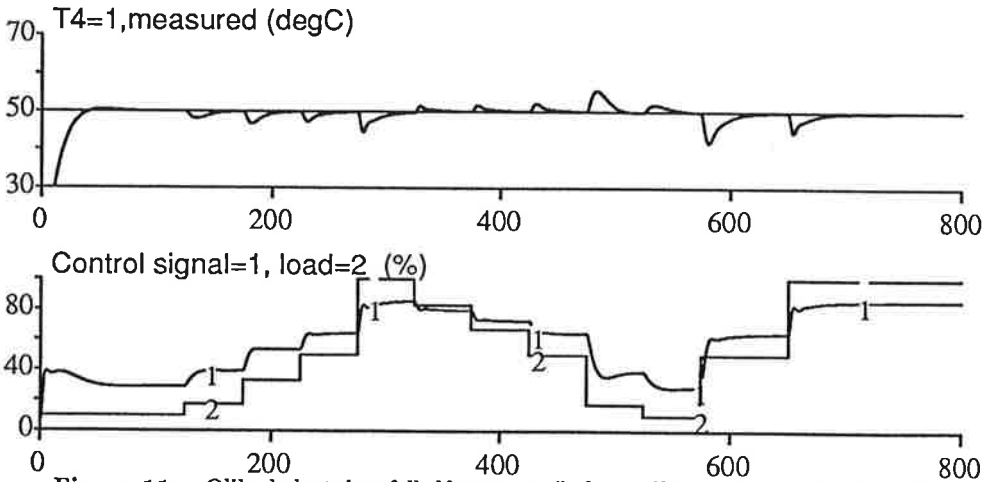


Figure 11. Olika belastningsfall då man använder en PI-regulator och ett snabbt ställdon

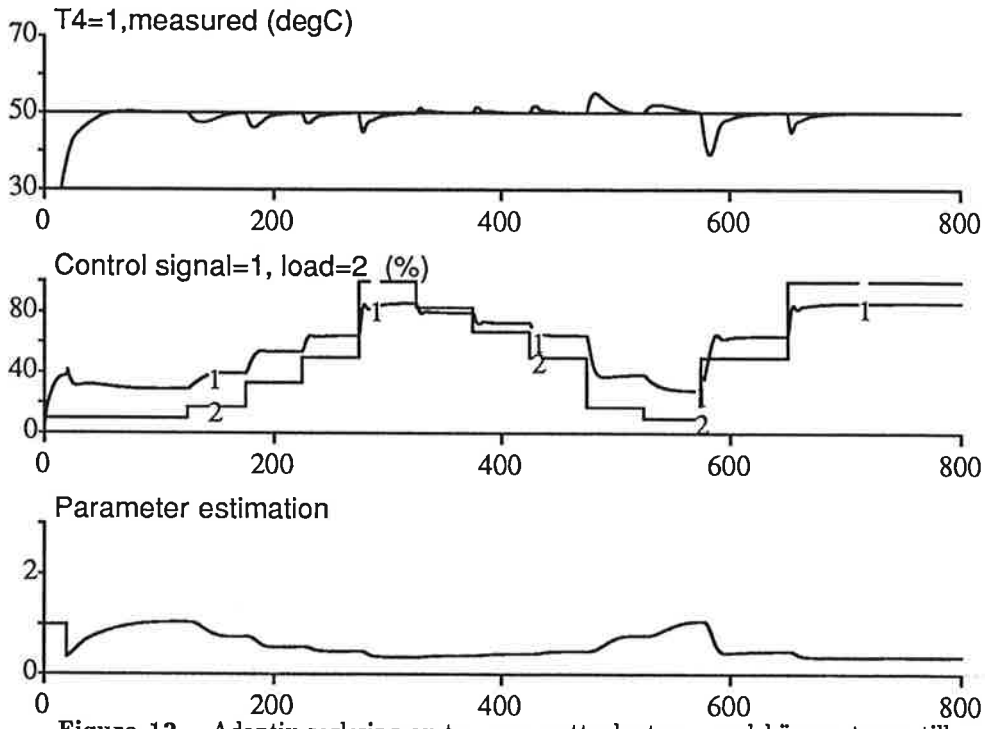


Figure 12. Adaptiv reglering av tappvarmvattenkretsen, med hänsyn tagen till en biasnivå, då man använder ett snabbt ställdon

Båda simuleringarna uppvisar ett likartat beteende jämfört med tidigare simuleringar, dock gör det snabbare ställdonet att tappvarmvattentemperaturen återkommer fortare till börvärdet än med ett långsammare ställdon. Liksom tidigare ses att den adaptiva regulatorn är snabbare vid belastningssökningar men långsammare vid belastningsminskningar jämfört med TA:s regulator med sin konstanta förstärkning.

Sammanfattning och slutsatser

Hela kretsens uppförande bestäms till stor del av ställdonets gångtid, vilket då begränsar systemets prestanda. TA:s PI-regulator med konstant förstärkning klarar de uppställda kraven bra upp till medelstora laster. Det går alldeles utmärkt att använda sig av en adaptiv regulator, för att reglera tappvarmvattensystemet. På grund av det långsammare ställdonet kan vi inte förbättra regleringen nämnvärt med den adaptiva regulatorn. Simuleringarna visar dock att vi kan skatta processförstärkningen bra och att den adaptiva regulatorn därför har en potential att förbättra regleringen om ett snabbare ställdon användes. I vårt fall använde vi oss av en enkel rekursiv minsta-kvadratskattare för att skatta den uppställda modellens förstärkning, vilken sedan utnyttjades i regulatorns förstärkning.

Det som kan undersökas närmare är bl.a bias-termen, inverkan av annan sampligstid samt annan glömskefaktor hos skattaren. Det går säkert också att finna ett bättre lågpassfilter.

Referenser

TOUR-ANDERSSON, *Simnonkod för simulering av tappvarmvattensystem*, Malmö, Sverige.

ÅSTRÖM, K. J and B. WITTENMARK (1989): *Adaptive control*,, Addison Wesley..

Bilaga 1

```
continuous system filt


input T4 pos
output T4f posf

time t
state y1 y2 u1 u2
der   ny1 ny2 nu1 nu2

ny1=-1.25663706143592*y1-0.3947841760435*y2+T4
ny2=y1
T4f=0.3947841760435*y2


nu1=-1.25663706143592*u1-0.3947841760435*u2+pos
nu2=u1
posf=0.3947841760435*u2

end
```

```
discrete system skatt

input T4 Tref u
output th

state theta P
new  ntheta nP

time t
tsamp ts

e=T4-u*theta


w=P*u
den=w*u+lam


u1= (theta+w*e/den)
theta1=if t>20 then u1 else theta
th=theta1
ntheta=theta1

u2=if abs(T4-Tref)>alfa then (P-w*w/den)/lam else P
P1=if t>20 then u2 else P
nP=P1
ts=t+h

h:0.1
alfa:1
lam:0.99

end
```

Projekt i kursen Adaptiv Reglering
Reglering av tappvarmvattentemperatur

Jörgen Malmborg

Department of Automatic Control
Lund Institute of Technology
Juni 1992

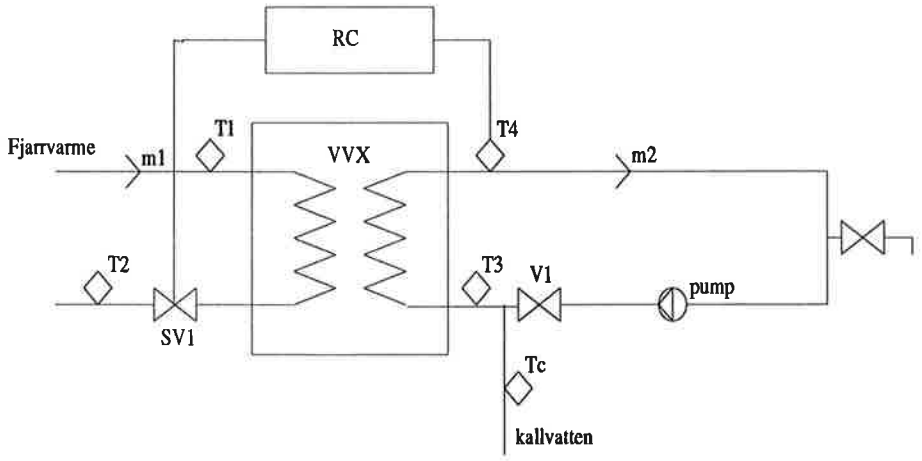


Figure 1. Tappvarmvattenkrets. T1=temperatur på inkommande fjärrvärme, T2=temperatur på returen till fjärrvärmenätet, T3=inkommande temperatur på sekundärsidan, T4=tappvarmvattentemperatur, Tc=kallvattentemperatur, m1=massflöde på primärsidan, m2=massflöde på sekundärsidan. SV1 är en reglerventil med tillhörande ställdon och V1 är en backventil. RC betecknar regulatorn.

1. Problembeskrivning

Fig. 1 visar en schematisk bild över en tappvarmvattenkrets. Målet med regleringen är att hålla tappvattentemperaturen (T_4) konstant när uttaget av varmvatten (m_2) ändras. Detta åstadkommes genom att reglera flödet genom ventilen (SV1) på värmeväxlarens primärsida. Svårigheten hos problemet är att överföringsfunktionerna för värmeväxlaren och ventilen är olinjära funktioner av belastningen. Temperaturvariationer hos kallvattnet och variationer i tryck och temperatur hos flödet fjärrvärmenätet spelar också roll. Effekten av dessa variationer har inte undersökts i projektet.

Modell

Problemkonfigurationen är som i fig. 1. Man kan mäta temperaturer på vattnet innan och efter värmeväxlaren både på primär och på sekundärsidan. Några flöden vill man inte mäta (flödesmätare är dyra).

För att simulera tappvarmvattensystemet användes en simnonmodell fig. 2 [Tour-Andersson, 1992] som TA har tagit fram.

När den adaptiva regulatorn beräknades användes en förenklad modell. Här bortser man från sensordynamiken som är relativt snabb. Blocken Act, Valv, VVX och Recirc slåss samman till ett enda olinjärt block. Den förenklade modellen blir som i fig. 3.

I identifieringsfasen är det överföringsfunktionen för detta nya VVX-block som skattas för olika belastningsfall.

Krav på prestanda

Svensk byggnorm ställer krav på prestanda ([Johansson and Wennberg, 1992]):

- Stationära reglerfelet ska vara mindre än 2°C.
- Vid 50% belastningsändring får man ha ett reglerfel på max 10°C.
- Vid mer än 30% belastning tillåts ingen temperatursvängning. Vid lägre belastning får amplituden hos svängningarna högst vara 2°C.

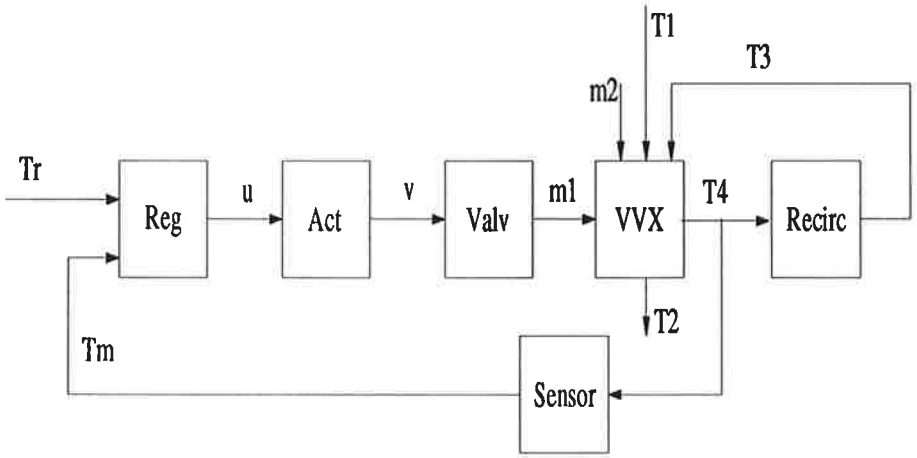


Figure 2. SIMNON-modell. Reg=regulatorn, Act=ställdonet, Valv=ventilen på primärsidan, VVX=värmeväxlaren, Recirc=modell för recirkulationen av varmvattnet, sensor=temperaturmätare.

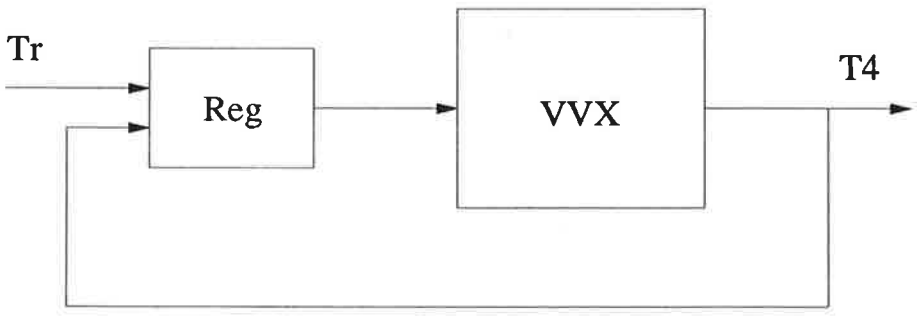


Figure 3. Förenklad modell

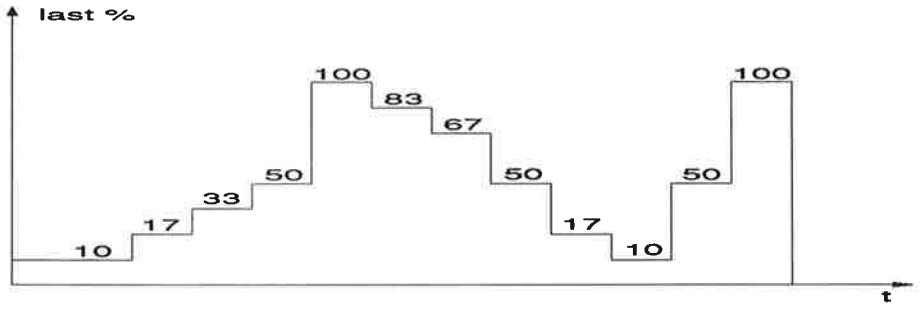


Figure 4. finsk belastningsprofil

- Reglerfelet ska var mindre än 2°C inom 2 minuter från belastningsändringen.
- Man ska klara en belastningsprofil enligt finska byggnormer. Fig. 4

2. Identifiering

Modell av värmeväxlaren

Första steget var att välja en modell för värmeväxlaren och sedan identifiera parametrarna i denna modell. Jag provade några olika modeller. Till att börja med ansatte jag ett första ordningens system med tidsfördröjning. Skattningar av denna typ av överföringsfunktion visade att tidsfördröjningen var försumbar. Jag gick därför över till en enklare modell för värmeväxlare inklusive ställdon och ventil av typen:

$$G_{vvx} = \frac{K_{vvx}}{1 + sT_{vvx}} \tag{1}$$

Modellen har två parametrar K_{vvx} och T_{vvx}, som varierar som funktioner av belastningen. Detta ger enkla beräkningar av regulatorparametrarna.

Variationer i förstärkning och tidskonstant

Från ekvation 1 får vi:

$$\begin{aligned} Y(s) + sT_{vvx}Y(s) &= K_{vvx}U(s) \\ y(t) &= -T_{vvx}\dot{y}(t) + K_{vvx}u(t) \\ y(t) &= [T_{vvx}\ K_{vvx}][-\dot{y}(t)\ u(t)]^T \end{aligned} \tag{2}$$

För att ta fram hur K_{vvx} och T_{vvx} varierar använde jag en kontinuerlig minstakvadratskattare [Åström and Wittenmark, 1988]:

$$\begin{aligned} \frac{d\hat{\theta}}{dt} &= P(t)\phi(t)e(t) \\ e(t) &= y(t) - \phi^T(t)\hat{\theta}(t) \\ \frac{dP(t)}{dt} &= \alpha P(t) - P(t)\phi(t)\phi^T(t)P(t) \\ \phi^T(t) &= [-\dot{y}_f(t)\ u_f(t)] \end{aligned} \tag{3}$$

Denna skattare minimerar kriteriet:

$$V(\theta) = \int_0^t e^{-\alpha(t-s)}(y(s) - \phi^T(s))\theta)^2 ds \tag{4}$$

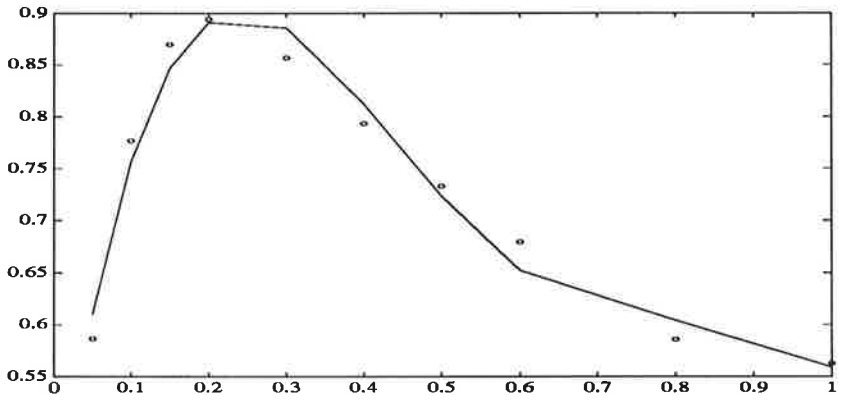


Figure 5. variationen hos processens förstärkning

Filtrerade mätdata

Som regressorer använde jag $\dot{y}_f(t)$ och $u_f(t)$ som är filtrerade y(t) respektive u(t). Filtret är av bandpasstyp för att få bort eventuell bias och frekvenser ovanför de som är intressanta för processen. Som bias kommer här ju in bl.a. kallvattnets temperatur. Det använda filtret har överföringsfunktionen:

$$\begin{aligned} G_f &= \frac{sT_1}{1+sT_1} * \frac{1}{(1+sT_2)^4} \\ T_1 &= 50 \\ T_2 &= 0.5 \end{aligned} \tag{5}$$

Glömskefaktor

Normalt skulle man ju inte behöva använda någon glömskefaktor α när man kör med konstant belastning. Jag körde emellertid estimeraren från start med ventilen stängd och då är processen anorlunda. Med glömskefaktorn α=0.02 konvergerade parametrarna snabbt efter uppstartningsfasen.

Exitation

För att åstadkomma tillräcklig exitation lät jag T_{ref} variera enligt:

$$T_{ref} = 50 + 4 * mod(60, 30) \tag{6}$$

Huvudsakliga energin hos denna signal ligger på frekvenser intressanta för processen och inom filtrets passband.

Resultat av identifieringen

Jag ställde in olika belastningsnivåer och estimerade K_{vvx} och T_{vvx} för dessa nivåer. Figurerna 5 & 6 visar variationerna hos värmeväxlarens förstärkning och tidskonstant. De heldragna linjerna är en anpassning av ett 4:e grad-spolynom till estimerade data. Dessa polynom används sedan av den adaptiva regulatorn vid beräkning av K & T_i. Polynomens grafer är kantiga därför att jag inte använt så många punkter vi plottningen.

I fig. 5 ser man att förstärkningen har en topp vid en belastning på 0.2. Om man använder en konstant regulator är denna topp begränsande och den totala förstärkningen i kretsen blir lägre både för större och mindre belastningar. Här ser man möjligheten till förbättringar genom att använda en regulator med belastningsberoende parametrar.

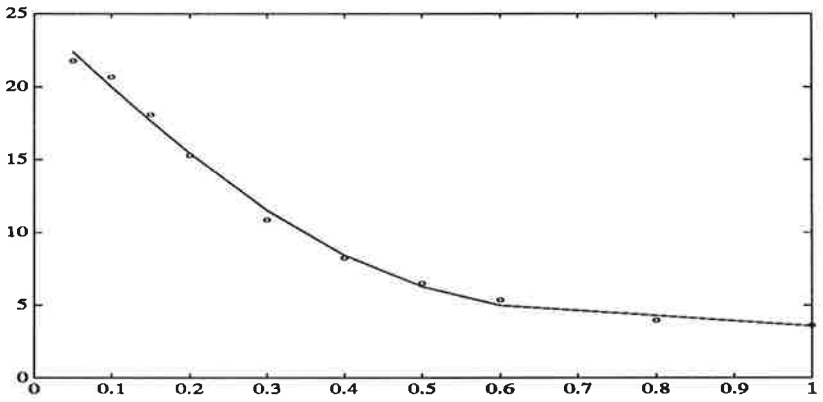


Figure 6. variationen hos processens tidskonstant

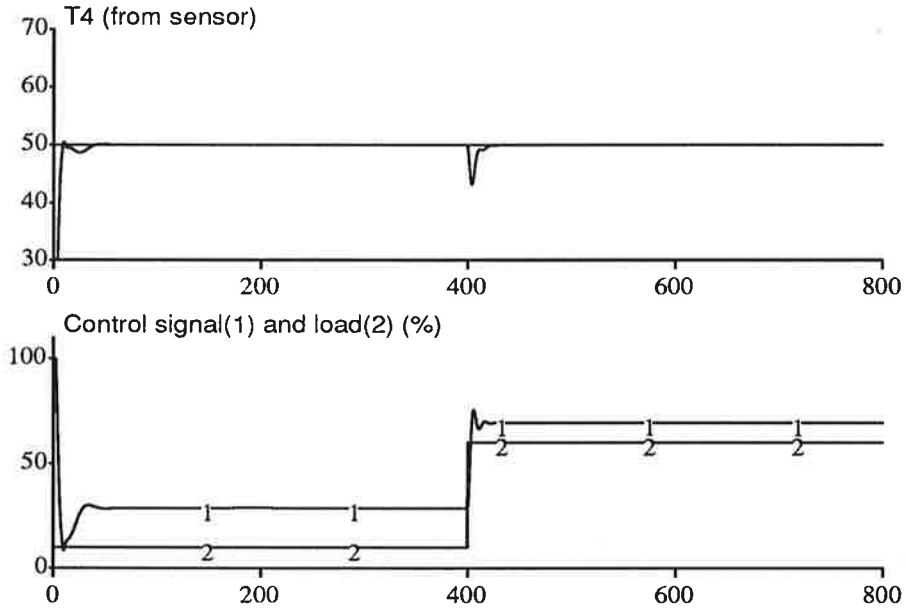


Figure 7. Stegsvar för en belastningsändring från 0.1 till 0.6. Ställdon med en gångtid på 0.6s

3. Inställning av PI-regulator

För att ta fram en adaptiv regulator började jag med att leta regulatorparametrar som gav ett bra resultat för belastningen 0.6. Då jag inte ville att begränsningar hos ställdonet skulle inverka använde jag mig av ett snabbt ställdon med en gångtid på 0.6s. Detta val är i och för sig orealistiskt men det ger en gräns för vad som kan åstadkommas. Fig. 7 visar stegsvaret för en belastningsändring från 0.1 till 0.6. Regulatorns överföringsfunktion är:

$$G_{PI} = K(1 + \frac{1}{sT_I}) \tag{7}$$

Från detta experiment valde jag de nominella parametervärdena till K=5 och T_I=5. Denna regulator ger ett snabbt och ganska dämpat uppförande. Regulatorn som TA:s SIMNON-modell använder har parametervärdena K=3 och T_I=10.

Filtrering av styrsignalen

Speciellt vid experiment med snabba ställdon visar det sig att ställdonet kommer att stå och slå mellan max hastighet framåt och max hastighet bakåt. För att komma till rätta med det problemet lågpassfiltrerade jag styrsignalen. Nu kunde jag få ett snällt uppförande på styrsignalen och ställdonets rörelser. Valet av filter är inte kritiskt. Det jag använde hade överföringsfunktionen:

$$G_f = \frac{3}{3+s} \tag{8}$$

4. Gain Scheduling

Inställning av regulatorn sker nu med gain scheduling. Värmeväxlarens tidskonstant och förstärkning som funktion av belastningen fås av polynomen ovan. Därefter beräknas regulatorparametrarna K och T_I så att samma dynamik för det slutna systemet fås för alla belastningar.

Beräkning av T_I och K

Slutna systemets överföringsfunktion blir:

$$\begin{aligned} G_c &= \frac{K(1+\frac{1}{sT_I})(\frac{K_{vvx}}{1+sT_{vvx}})}{1+K(1+\frac{1}{sT_I})(\frac{K_{vvx}}{1+sT_{vvx}})} \\ &= \frac{\frac{KK_{vvx}}{T_IT_{vvx}}(1+sT_I)}{s^2+s(\frac{1}{T_{vvx}}+\frac{KK_{vvx}}{T_{vvx}})+\frac{KK_{vvx}}{T_IT_{vvx}}} \end{aligned} \tag{9}$$

Den karakteristiska ekvationen blir:

$$s^2 + s\frac{1}{T_{vvx}}(1+KK_{vvx}) + \frac{KK_{vvx}}{T_IT_{vvx}} \tag{10}$$

$$s^2 + a_1 s + a_0 \tag{11}$$

$$\begin{aligned} a_1 &= \frac{1}{T_{vvx}}(1+KK_{vvx}) \\ a_0 &= \frac{1}{T_{vvx}}(\frac{KK_{vvx}}{T_I}) \end{aligned} \tag{12}$$

De nominella värdena på a_1 och a_0 tog jag från experiment vid en belastning på 0.6.

$$\begin{aligned} \bar{a}_1 &= \frac{1}{\bar{T}_{vvx}}(1+\bar{K}\bar{K}_{vvx}) = \frac{1}{5.37}(1+5*0.680) = 0.82 \\ \bar{a}_0 &= \frac{1}{\bar{T}_{vvx}}(\frac{\bar{K}\bar{K}_{vvx}}{\bar{T}_I}) = \frac{1}{5.37}(\frac{5*0.680}{5}) = 0.13 \end{aligned} \tag{13}$$

Vi har ω_0=0.36 och ζ=1.15. Gain scheduling görs sedan så att samma a_1 och a_0 fås för alla belastningar. För att få den önskade karakteristiska ekvationen ska man ställa in regulatorparametrarna på:

$$\begin{aligned} K &= \frac{a_1T_{vvx}-1}{K_{vvx}} \\ T_I &= \frac{KK_{vvx}}{a_0T_{vvx}} \end{aligned} \tag{14}$$

Med insatta värden på $\bar{a}_1$ och $\bar{a}_0$ får vi:

$$\begin{aligned} K &= \frac{1}{K_{vvx}}\left(\frac{T_{vvx}}{\bar{T}_{vvx}}(1+\bar{K}\bar{K}_{vvx})-1\right) \\ T_I &= \bar{T}_I\frac{(1+\bar{K}\bar{K}_{vvx})-\frac{\bar{T}_{vvx}}{T_{vvx}}}{\bar{K}\bar{K}_{vvx}} \end{aligned} \tag{15}$$

Hur får vi tag i K_{vvx} och T_{vvx}?

För inställningen behövs nu antingen en direkt skattning av K_{vvx} och T_{vvx} eller en skattning av belastningen som ger oss K_{vvx} och T_{vvx} via polynomen.

Försök med en finsk belastningsprofil visar att skattningarna av K_{vvx} och T_{vvx} normalt inte konvergerar tillräckligt snabbt för att ge acceptabelt uppförande hos T_4. Det hjälper inte om man tillför konstgjord exitation med börvärdesändringar.

I alla mina simuleringar utom den sista, fig. 14 har jag därför använt det korrekta värdet på belastningen.

En alternativ metod är att ur temperaturmätningar beräkna belastningen. Antag att vi lyckas med att hålla T_4 konstant. Då kommer också det recirkulerade vattnet att ha konstant temperatur. Vi bör därför kunna bortse från dynamik och tidfördröjning i Recirc. Vattnet som kommer tillbaka har temperaturen $0.9 * T_4$. Ur en temperaturrelation får man fram andelen kallt vatten i det som kommer in till värmeväxlaren:

$$\begin{aligned} T_3 &= x * T_c + (1-x) * 0.9 * T_4 \\ x &= \frac{T_3 - 0.9T_4}{T_c - 0.9T_4} \end{aligned} \tag{16}$$

Nu vet vi hur mycket vatten vi fyllt på och känner därför belastningen. Sista figuren visar resultatet av en simulering med en adaptiv regulator där gainscheduling görs med hjälp av den skattade belastningen. Resultatet verkar tillfredställande. Man märker ingen skillnad mot om man använder det korrekta värdet på belastningen.

5. Jämförelser mellan nuvarande regulator och den adaptiva

Snabbt ställdon

Fig. 8 och fig. 9 visar resultatet av experiment med finsk belastningsprofil och ett snabbt ställdon med en gångtid på 0.6s. Dessa figurer visar gränsen för vad som kan åstadkommas om man inte var beroende av långsamma ställdon. Det är tveksamt om modellen gäller för denna korta gångtid.

Långsamt ställdon

Fig. 10 och fig. 11 visar resultatet av experiment med finsk belastningsprofil och det normala ställdonet med gångtiden 60s. Skillnaden mellan den adaptiva regulatorn och den konstanta är inte så stor. Ändringar i regulatorn har nästan ingen betydelse då kretsens uppförande helt domineras av det långsamma ställdonet.

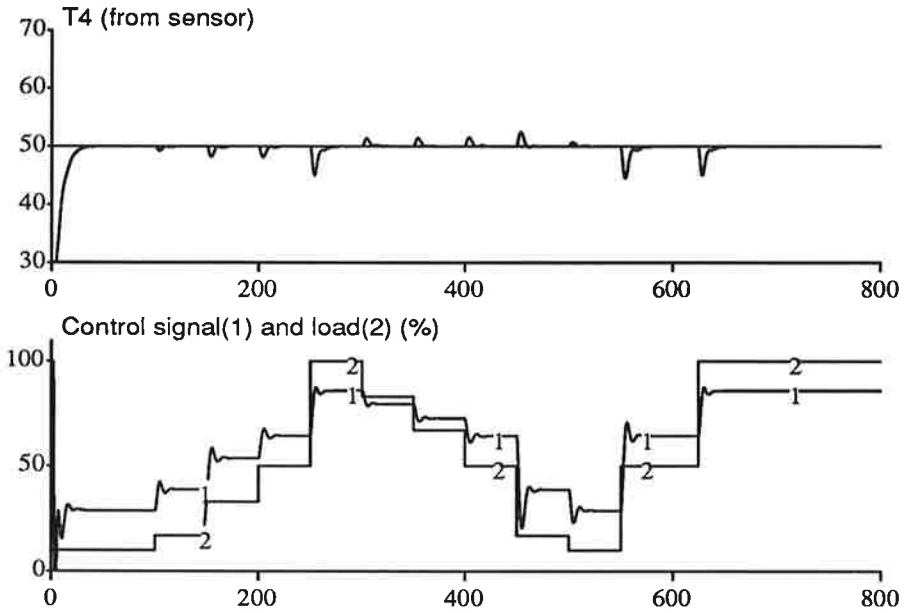


Figure 8. gångtid=0.6, adaptiv regulator

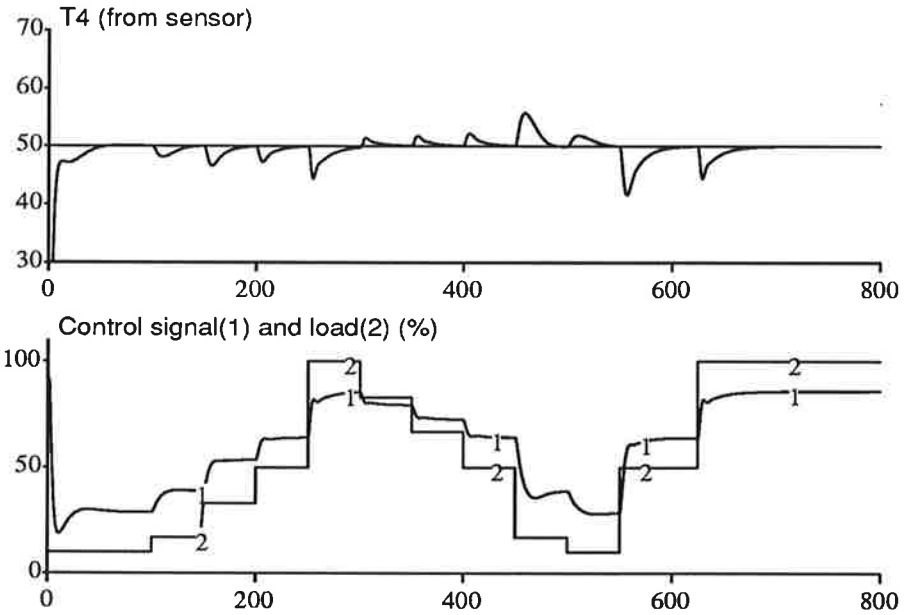


Figure 9. gångtid=0.6, konstant regulator

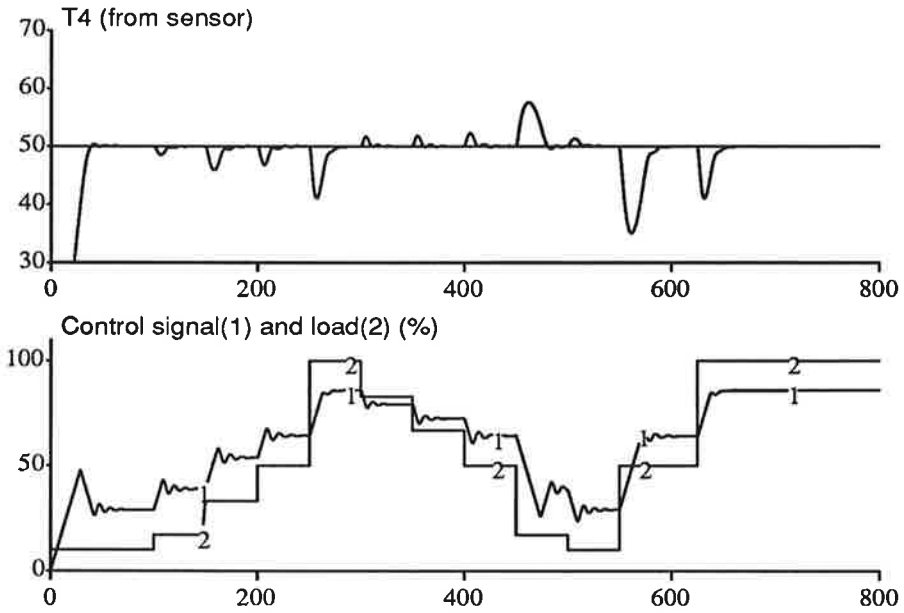


Figure 10. gångtid=60, adaptiv regulator

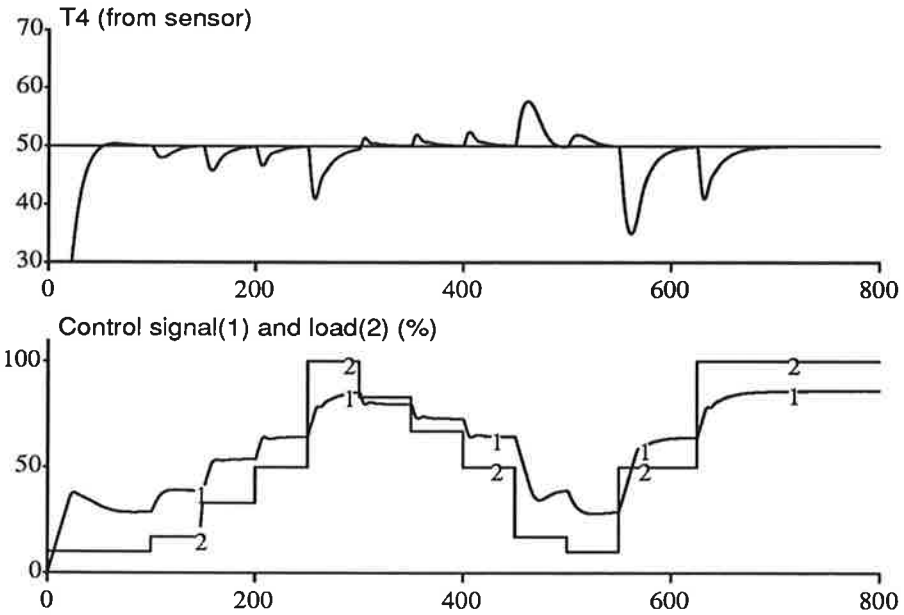


Figure 11. gångtid=60, konstant regulator

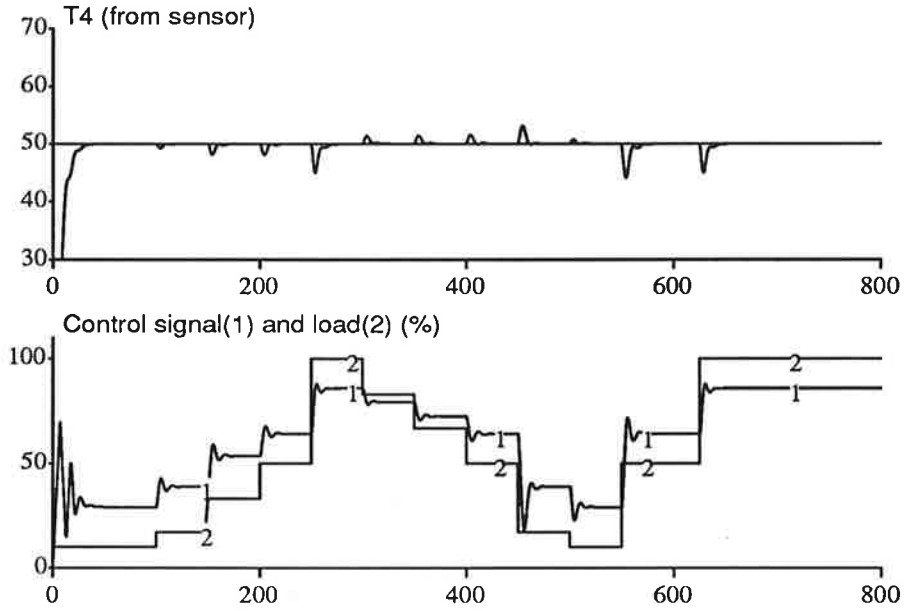


Figure 12. gångtid=10, adaptiv regulator

Medelsnabbt ställdon

Fig. 12 och fig. 13 visar resultatet av experiment med finsk belastningsprofil och ett något snabbare ställdon. Gångtiden är 10s. Den adaptiva regulatorn ger ett överlägset uppförande. Man kan också notera att det inte lönar sig med ställdon med kortare gångtid än 10s. Den sista figuren, fig 14 är resultatet av en simulering med en adaptiv regulator där belastningen skattas ur temperaturmätningar.

6. Slutsatser

Simuleringar har visat att det finns mycket att vinna på att använda en adaptiv regulator. Man uppfyller med bred marginal alla prestandakrav vid experiment med den finska belastningsprofilen.

En annan viktig punkt är att det räcker med ställdon med en gångtid på 10s. Om gångtiden minskas ännu mera tjänar man inget extra.

Inga flödesmätningar behöver göras. Det räcker med ytterliggare en temperaturmätning.

Ej analyserat

Jag har inte gjort några undersökningar om hur tryck- och temperaturvariationer i primärflödet påverkar. Vad gäller variationer i kallvattentemperaturen kommer dessa inte att spela någon roll om de är någorlunda långsamma. Då tar I-delen hand om dem.

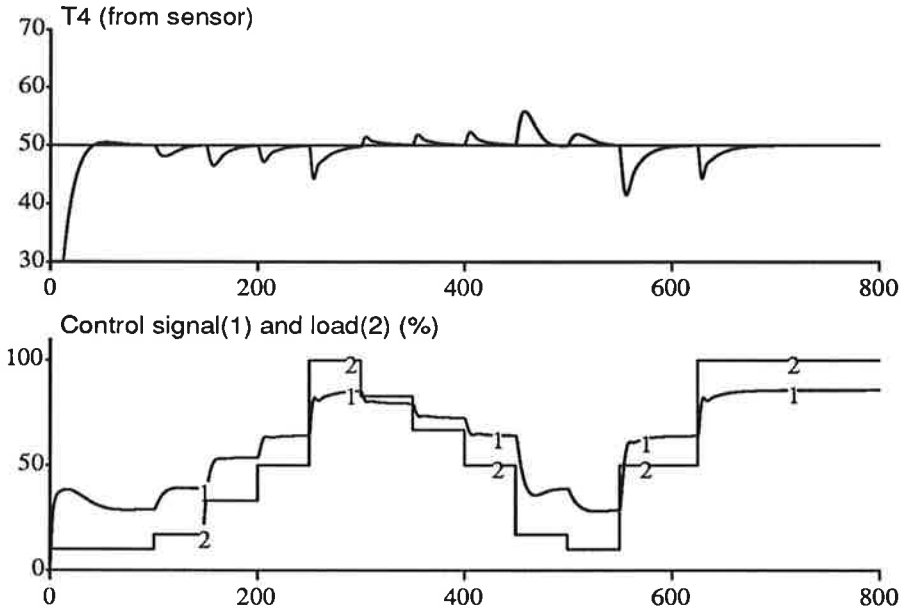


Figure 13. gångtid=10, konstant regulator

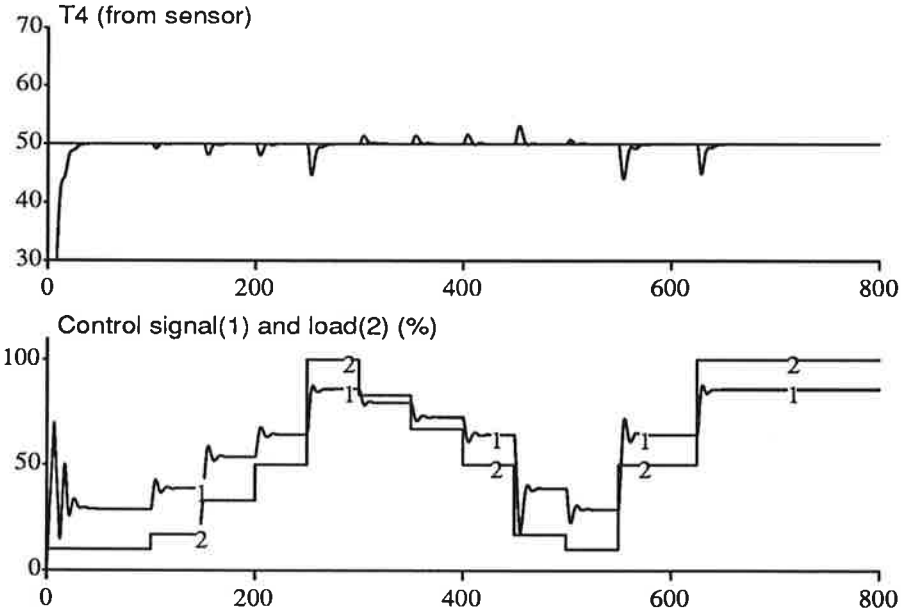


Figure 14. gångtid=10, adaptiv regulator, beräkning av belastning från temperaturdata

7. Referenser

[Johansson and Wennberg, 1992] Johansson, J. and Wennberg, M. (1992). *Adaptiv reglering av tappvarmvatten.*

[Tour-Andersson, 1992] Tour-Andersson (1992). *SIMNON-modell för tappvarmvattensystem.*

[Åström and Wittenmark, 1988] Åström, K. and Wittenmark, B. (1988). *Adaptive Control.* Addison Wesley.

Autotuning med fuzzyteknik
Projekt i Adaptiv reglering
och
Datorimplementering av reglersystem

Lars Hermansson, F-87
och
Stefan Gustafsson, F-87

Maj 1992

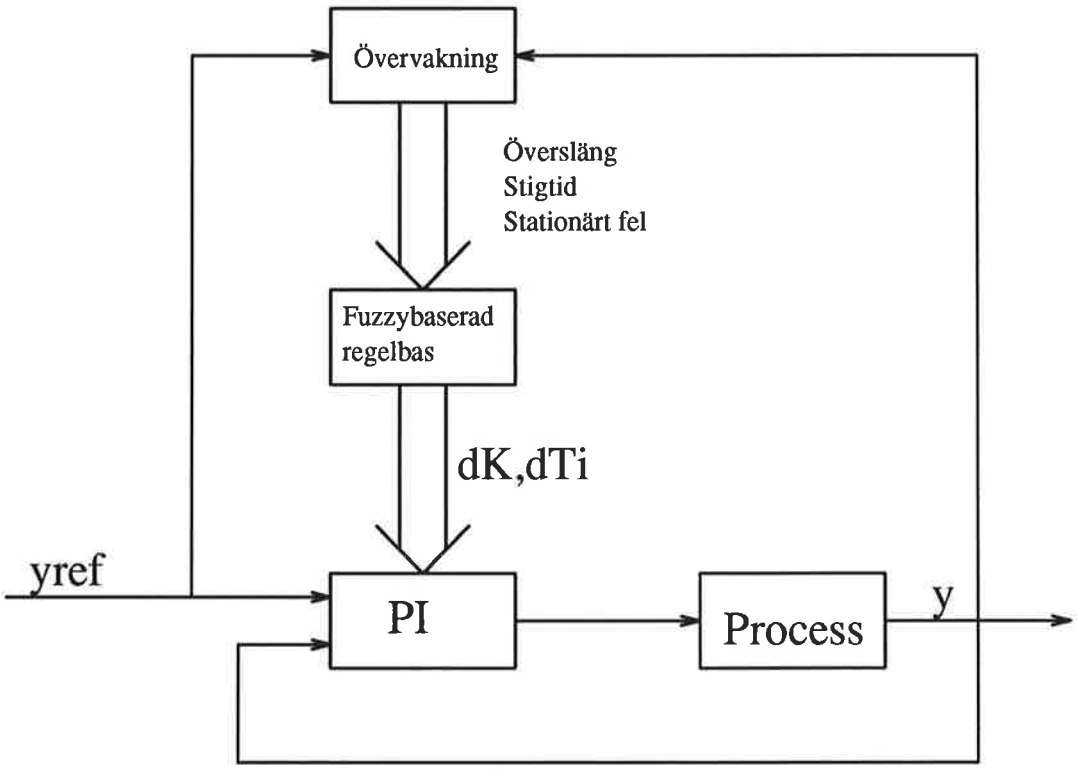
Innehåll

1 Inledning

Den här rapporten handlar om ett kombinerat projekt i kurserna Adaptiv reglering och Datorimplementering av reglersystem. Vi har med hjälp av fuzzy teknik implementerat en självinställande PI-regulator, där den automatiska parameterinställningen försöker efterlikna en mänsklig operatörs. Regulatorn har implementerats på PC i Modula-2 och Tilshell.

2 Bakgrund

Vi har implementerat en självinställande PI-regulator med hjälp av fuzzy teknik. Idèerna har vi hämtat från artikeln *An Analytical Framework for Knowledge-Based Tuning of Servo Controllers*, skriven av Clarence de Silva. Arbetet har inspirerats av att den praktiska erfarenheten vid inställning av servoregulatorer finns tillgänglig i form av mjuka regler och inte som direkta matematiska algoritmer. Detta har lett till en regulatortyp som inte använder fuzzy logik i direkt reglering, utan på en högre nivå vid inställning av PI-regulatorn. Parametrarna ändras med hänsyn till tre uppförandeattribut, vilka mätes i övervakningsenheten, varefter förändringarna av parametrarna fås med hjälp av ett antal regler i fuzzyregelbasen, se figur 1. De nya parametrarna laddas ner i regulatorn. Parameteruppdateringen sker betydligt mer sällan än uppdateringen av styrsignalen.



Figur 1: Blockschema över regulatorn

3 Fuzzy

3.1 Inledning till fuzzyteknik

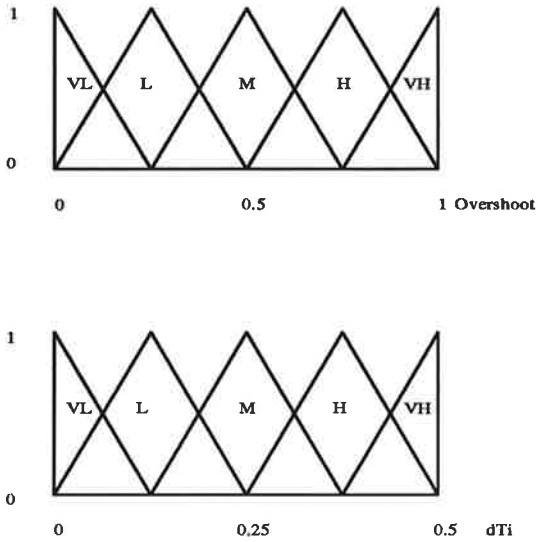
Vårt arbete bestod av att konstruera en självinställande PI-regulator med hjälp av fuzzy teknik. Fuzzy reglering, vilken baseras på fuzzy logik, är en form av kunskapsbaserad reglering som innefattar den ingenjörsmässiga delen av reglertekniken i form av en samling lingvistiska lagar. Dessa lagar innehåller suddiga ("fuzzy") villkor i stil med "ganska långsamt", "mycket bra" etc. Suddighet bör här inte förväxlas med ord som vaghet, osäkerhet, generalitet osv. Konventionell fuzzy reglering innebär att styrsignaler beräknas direkt med hjälp av lingvistiska regler. Denna typ av reglering är något av en modenyck just nu, och särskilt populär i Japan där man använder metoden i bland annat tvättmaskiner, tunnelbanetåg och skakningskompenserade videokameror. Vi har dock implementerat en annan tillämpning av tekniken, en tillämpning man finner då man studerar en mänsklig operatör som ställer in en PID-regulator.

3.2 Inställning av PI-regulator

En operatör som har uppdraget att ställa in eller justera en konventionell PID-regulator skrider till verket ungefär enligt följande: Hon (vi antar att det är en kvinnlig operatör) börjar med att ställa in några parametervärden som hon tror kan vara ganska vettiga (Hon kanske någotsånär vet vilken storleksordning de bör ha), varefter hon skickar in ett steg till processen. Hon studerar stegsvarets översläng, justerar parametrarna med hänsyn till denna.Vid oacceptabelt stor översläng ska förstärkningen minskas och integraltiden ökas (i fortsättningen betecknade K och Ti). Dessa förändringars storlek beror helt naturligt på hur stor överslängen är; ju större överslängen ser ut, desto kraftigare förändring av K och Ti. Därefter kanske hon skickar in ett nytt steg, justerar lite igen med avseende på överslängen, testar igen med ett nytt steg. Den här gången kanske överslängen är acceptabel, hon går då över till att betrakta stigtiden, justera parametrarna med hänsyn till denna, skicka in ett nytt steg etc. På detta sätt kommer vår operatör förhoppningsvis fram till en regulator som reglerar processen bra. Vi har försökt att efterlikna operatörens inställningsstrategi i vår regulator. Det sekventiella och upprepade mönstret motsvaras av så kallad regelupplösning, vilken kort och gott innebär att man ställer in en parameter i taget. Det verkar vidare vara ganska vettigt att använda suddig logik som motsvarighet till operatörens mjuka, lingvistiska regler. Notera att vi inte använder fuzzy tekniken till direkt reglering, utan på en högre nivå, för att ställa in regulatorparametrarna i den konventionella regulatorn. Vi har (i likhet med många mänskliga operatörer) nöjt oss med att ställa in regulatorns förstärkning och integraltid.

3.3 Parameterändring med fuzzylogik

Fuzzy logik är en reell logik där en logisk variabel antar ett reellt värde mellan 0 och 1. Detta kan jämföras med den klassiska, diskreta, logiken där en logisk variabel antar endera 0 eller 1. Detta innebär att en utsaga är antingen sann eller falsk, inga mellanting. Den fuzzy logiken byggs upp med sk medlemsskapsfunktioner, se figur 2. VL betyder här Very Low etc. Till exempel så är en översläng på 0.2 till graden 0.8 "låg" och till graden 0.2 "mycket låg". Som jämförelse kan nämnas MIT-regeln där parametrarna regleras med hänsyn till kvadraten av avvikelsen från en referensmodell. I vårt fall ändrar vi parametrarna med hänsyn till tre



IF Overshoot is VH THEN
  dTi is VH
IF Overshoot is H THEN
  dTi is H
IF Overshoot is M THEN
  dTi is M
IF Overshoot is L THEN
  dTi is L
IF Overshoot is VL THEN
  dTi is VL

Figur 2: Exempel på medlemskapsfunktioner och lingvistiska regler

uppförandeattribut; översläng, stigtid och stationärt fel. Detta för att efterlikna den manuella expertinställningen. Inställningen av parametrarna sker med relativa förändringar. Det nya parametervärdet fås som $K*(1+dK)$ etc. Åtgärder för respektive uppförandeattribut framgår av tabellen nedan.

Uppförandeattribut	K	Ti
Översläng	Minska	Öka
Stigtid	Öka	Öka
Stationärt fel	Öka	Minska

Vi utvecklade vår regelbas med hjälp av Tilshell som är en professionell utvecklingsmiljö för utveckling av fuzzyregulatorer. Programmet körs från arbetstationer och sköts till största delen med muskommandon, med vilkas hjälp man definierar sitt system och sätter ihop regelbasen. Av andras erfarenhet vet vi att utseendet av medlemsskapsfunktionerna inte har så stor betydelse, varför vi använt oss av triangulära sådana. Vi har valt följande gränser för uppförandeattributen:

- Översläng: 0 och 1, dvs maximalt 100
- Stigtid: 1 och 2, relaterat till en maximal tillåten stigtid.
- Stationärt fel: 0 och 1.

3.4 Reglersystemets uppbyggnad

Övervakningsenheten är implementerad i det sekventiella programmeringsspråket Modula2, som arbetar i realtid. Definitionsmoduler framgår av avsnittet om modulstrukturer. Tilshell har inbyggt ett kommando som kompilerar systemet och regelbasen till färdigt program i programmeringsspråket C. C-koden flyttas därefter över till PC med programmmet Kermit och länkas samman med Modula-koden till ett exekverbart progaram. Sammanlänkningen görs med programmet Blink. Efter lite trixande har vi nu fått en färdig regualator som

vi kan använda för att styra någon process via en PC. Avslutningsvis kopplade vi upp vår regulator mot realtidssimnon för att testa beteendet. Vi såg då att man måste ha olika relativa förandringar för de bägge parametrarna samt för de olika uppförandeattributen. Till exempel så måste man vid en mycket hög stigtid fördubbla K för att få tillräcklig effekt. Detta beror ju på att ett långsamt system i regel har för låg förstärkning, och en liten relativ förändring av ett lågt värde ger ju en låg absolut förändring. Vi märkte också att vid stationärt fel måste man vara försiktig vid ökningen av K, annars riskerar man att K drar iväg för högt och ger upphov till kraftig översläng och eventuellt instabilitet. Efter ytterligare lite justeringar av den här typen fick vi till en regulator som vi tyckte att vi kunde vara nöjda med.

4 Implementering

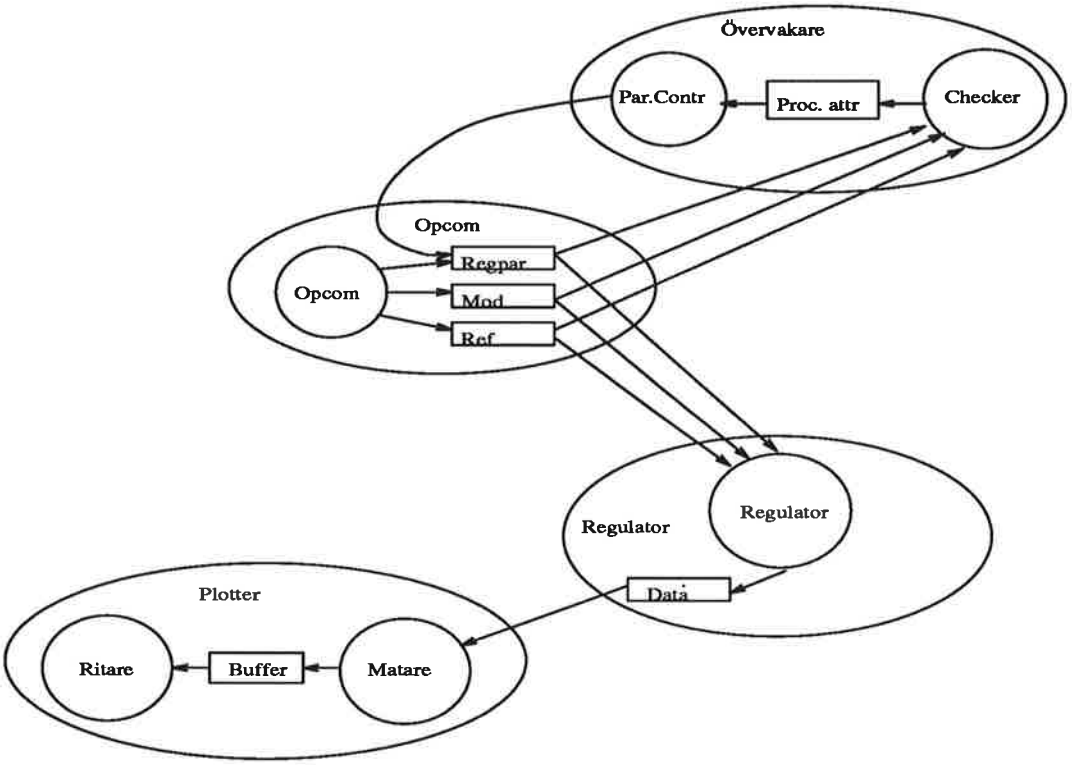
4.1 Processtruktur

Programmet är uppbyggt av fyra moduler Opcom, Regulator, Plotter och Supervisor, se figur 3. Plotter innehåller två processer, en process Matare som hämtar plotdata från Regulator och lägger dessa värden i en buffer och en process Ritare som hämtar data från buffer och plottar dem. Modulen Regulator innehåller en process som har hand om själva regleringen. Regulatorprocessen hämtar värden på regulatorparametrar, referensvärde och mod från Opcom och lägger sedan de parametrar som skall plottas i monitorn Data. Den automatiska parameterinställningen sköts av modulen Supervisor som består av två processer. Processen Checker övervakar systemets uppförande och bestämmer översläng, stigtid och stationärt fel. Processen lägger sedan dessa värden i monitorn Proc.attr varifrån processen Par.Contr hämtar dem. Par.Contr beräknar vilka ändringar som behövs göras av regulatorparametrarna och skickar sedan de nya regulatorparametrarna till Opcom, där de läggs i monitorn RegPar. Beskrivningen av modulen Opcom i processgrafen nedan är något förenklad. Modulen består i själva verket av flera processer men vi har för enkelhetens skull bara ritat en. Det kanske ska tilläggas att all kommunikation mellan de olika processerna i programmet sker via monitorer.

4.2 Modulstruktur

Nedan följer de olika definitionsmodulerna.

```
DEFINITION MODULE Opcom
EXPORT QUALIFIER RegParType,GetRegPar,SetRegPar,WaitForEnd,InitOpcom,
                 ModeType,GetMode,GetRef,SetRef;
TYPE RegParType=RECORD
                  K,Ti,Td:REAL:
                  h:CARDINAL;
                END;
ModeType=(off,Man,Auto);
PROCEDURE GetRegPar(VAR P:RegParType);
PROCEDURE SetRegPar(Parameterers:ARRAY OF REAL);
PROCEDURE WaitForEnd;
PROCEDURE GetMode(VAR m:ModeType);
PROCEDURE SetRef(r:REAL);
```

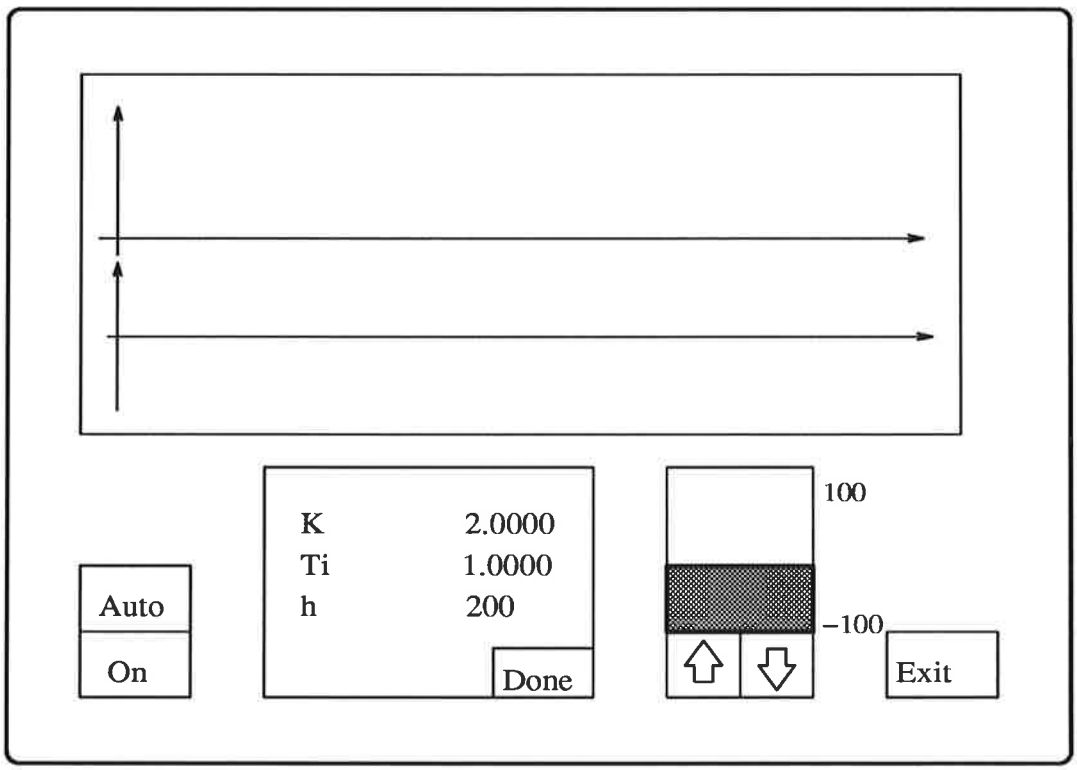


Figur 3: Processgraf

```
PROCEDURE GetRef(VAR r:REAL);
PROCEDURE InitOpcom;
END Opcom.

DEFINITION MODULE Regulator;
EXPORT QUALIFIER RegData,GetData,InitRegulator;
TYPE RegData = RECORD
                 y,yr,u,K,Td,Ti:REAL;
                 h :  CARDINAL;
               END;
PROCEDURE GetData(VAR d:RegData);
PROCEDURE InitRegulator; END Regulator.

DEFINITION MODULE Plotter;
EXPORT QUALIFIER InitPlotter;
PROCEDURE InitPlotter; END Plotter.

DEFINITION MODULE Supervisor;
EXPORT QUALIFIER InitSupervisor;
PROCEDURE InitSupervisor;
END Supervisor.
```

4.3 Operatörskommunikation

Figur 4 visar en förenklad bild av hur datorns skärm ser ut. I den övre grafen plottas regulatorparametrarna K och Ti och i den undre grafen plottas referensvärde,styrsignal och utsignal. Regulatorn kan köras i två moder nämligen autotuning och konventionell PI. Man väljer mod genom att klicka med musen i den övre delen av knappen i det nedre vänstra hörnet. För att stänga av respektive sätta på regulatorn klikar man i den undre delen av samma knapp. När regulatorn är avstängd eller körs i PI-mod kan man ändra regulatorparametrarna. Detta görs genom att man klickar på parametrarnas värden, skriver det nya värdet och slår return. När man ändrat de parametrar man ville ändra klickar man i rutan Done och de nya värdena registreras. Referensvärdesändringar gör man genom att klicka i stapeldiagrammet på önskad nivå eller genom att klicka på pilarna under stapeldiagrammet. Observera att automatisk inställning av regulatorparametrarna sker först då regulatorn befinner sig i autotuning-mod och man gör en börvärdesändring. För att få en bra parameteruppsättning kan man behöva göra fler än en stegändring. När man så är nöjd med den automatiskt inställda parametervärdena går man över i PI-mod och PI-regulatorn körs då med dessa värden. Programmet avslutas om man klickar i knappen Exit i det nedre högra hörnet.



Figur 4: Förenklad bild av operatörskommunikationen

5 Resultat

Vi provade vår regulator genom att köra den mot en process i realtidssimnon. Processens överföringsfunktion var $G(s) = (s = 1)^{-4}$. Resultatet kan studeras i tabell 5. Av tabellen

framgår att regulatorn efter endast ett fåtal justeringar ställt in sig hyggligt. För att nå ännu bättre prestanda måste man lägga ner betydligt mer tid och arbete vid konstruktionen av fuzzyreglerna. Även valet av maximal stigtid och tillåten översläng påverkar regulatorns inställning.

Startvärden	Efter 1:a tuningen	Efter 2:a	Efter 3:e	Efter 4:e
K=7	K=1.18	K=0.40	K=0.78	K=0.80
Ti=1	Ti=0.96	Ti=2.18	Ti=2.59	Ti=2.27
K=4	K=2.47	K=1.70	K=1.64	K=1.32
Ti=4	Ti=5.72	Ti=7.96	Ti=3.98	Ti=4.50
K=1	K=1.23	K=0.93	K=0.93	
Ti=7	Ti=3.18	Ti=4.04	Ti=4.04	
K=0.69	K=0.43	K=0.69		
Ti=1.84	Ti=1.85	Ti=1.76		

Rekommenderade värden : K = 0.69 Ti = 1.84

Figur 5: Resultat av körning mot process

6 Slutsats

Vi anser att vi har lyckats bygga en hyfsat bra självinställande PI-regulator med tanke på tiden projektet skulle ta. En lämplig utvidgning av projektet vore att göra de i avsnittet ovan föreslagna förbättringarna.

Referenser

[1] Clarence W. de Silva. *An analytical framework for knowledge-based tuning of servo controllers.* Applic. Artif. Intell. Vol. 4, No. 3, pp 177-189, 1991.

[2] Karl Johan Åström, Björn Wittenmark. *Adaptive Control.* Addison Wesley, 1989.

[3] Karl Erik Årzen. *Fuzzy Control.* Institutionen för reglerteknik, Lunds tekniska högskola.

[4] Lars Nielsen. *Computer implementation of control systems.* Institutionen för reglerteknik, Lunds tekniska högskola.

Simulering och analys av adaptiv motorstyrning med omodellerad stördynamik

ett projekt i kursen Adaptiv Reglering av
Margot Tischbierek, E-86

Lund, den 1 juni 1992

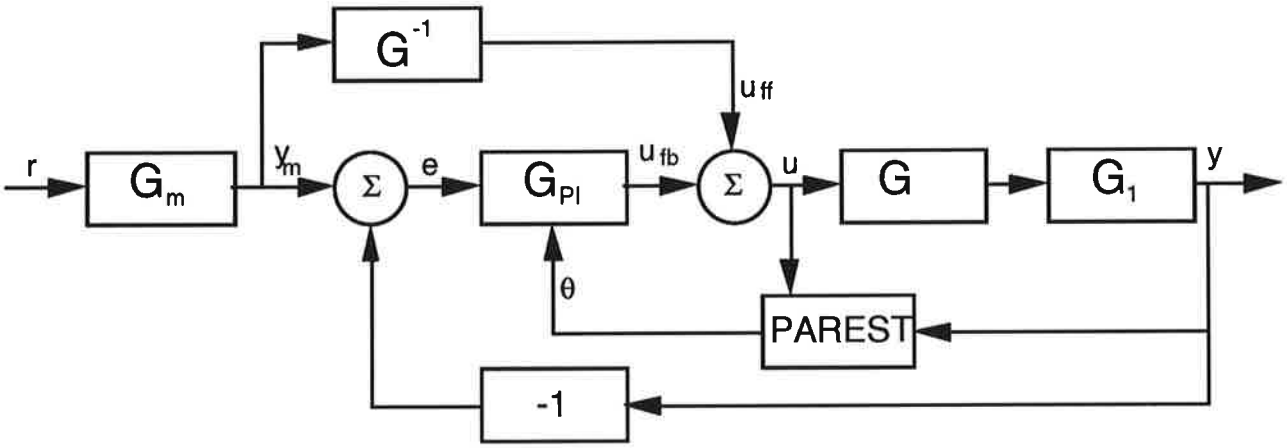
Innehållsförteckning

1 Inledning

Denna rapport behandlar en enkel reglering av en dc-motor. Av kursansvarige Karl-Johan Åström erhöll jag regulatorkod för det relativt realistiska antagandet, att dc-motorn kan modelleras som en integrator. Detta redovisas mer i detalj i kapitel 2. I kapitel 3 analyseras och verifieras det slutna systemets stabilitetsegenskaper, då processen har omodellerad dynamik. I kapitel 4 simuleras systemet i Simnon och utvärderas för de båda fallen då motorns tröghetsmoment skattas. Kapitel 5 består av en kort sammanfattning.

2 Systembeskrivning

Koden jag erhöll, implementerar en återkopplad PI-regulator med framkoppling. Den senare är till för att kunna åstadkomma önskat svar från referenssignal oberoende av PI-regulatorns designfrekvens. Den tidsdiskreta parameterestimatorn skulle emellertid ersättas av en tidskontinuerlig sådan. Dessutom kunde jag ändra på filtreringen av insignalerna till denna. Jag valde att simulera, dels med det filter som ingick i koden ursprungligen, dels med en lågpassfiltrering på respektive insignal till estimatorn. För att efterlikna verkligheten en aning, skulle jag lägga till en högfrekvent och dåligt dämpad störddynamik i processen. Systemets uppbyggnad visas nedan i figur 2.1.



Figur 2.1 Systemstruktur

Figurens överföringsfunktioner, de båda parameterestimatorerna samt härledning av regulator finns dokumenterade i appendix A.

3 Analys av rotort för det slutna systemet

En jämförelse av det slutna systemets kar. poly:

$$s^4 + 2\zeta_0\omega_0 s^3 + \omega_0^2 s^2 + 2\zeta\omega\omega_0^2 s + \omega^2\omega_0^2$$

och två komplexkonjugerade polpar

$$(s^2 + 2\zeta_1\omega_1 s + \omega_1^2)(s^2 + 2\zeta_2\omega_2 s + \omega_2^2) =$$

$$= s^4 + 2(\zeta_1\omega_1 + \zeta_2\omega_2)s^3 + (\omega_1^2 + 4\zeta_1\omega_1\zeta_2\omega_2 + \omega_2^2)s^2 +$$

$$+ 2\omega_1\omega_2(\zeta_1\omega_2 + \zeta_2\omega_1)s + \omega_1^2\omega_2^2$$

ger:

$$\begin{cases} \zeta_1\omega_1 + \zeta_2\omega_2 = \zeta_0\omega_0 & (1) \\ \omega_1^2 + 4\zeta_1\omega_1\zeta_2\omega_2 + \omega_2^2 = \omega_0^2 & (2) \\ \omega_1\omega_2(\zeta_1\omega_2 + \zeta_2\omega_1) = \zeta\omega\omega_0^2 & (3) \\ \omega_1^2\omega_2^2 = \omega^2\omega_0^2 & (4) \end{cases}$$

Om ω_2, ζ_2 relativt ζ_1, ω_1 är de snabbare rötterna så gäller följande för gränsen till instabilitet:

$\zeta_2 = 0$:

$$\begin{cases} \zeta_1\omega_1 = \zeta_0\omega_0 & (1') \\ \omega_1^2 + \omega_2^2 = \omega_0^2 & (2') \\ \zeta_1\omega_1\omega_2^2 = \zeta\omega\omega_0^2 & (3') \\ \omega_1^2\omega_2^2 = \omega^2\omega_0^2 & (4') \end{cases}$$

$$(3' \,|\, 1') \Rightarrow \omega_2^2 = \frac{\zeta\omega\omega_0}{\zeta_0} \quad (5')$$

$$(5' \Rightarrow 4') \Rightarrow \omega_1^2 = \frac{\omega^2\omega_0^2}{\omega_2^2} = \frac{\zeta_0\omega\omega_0}{\zeta} \quad (6')$$

$$(5', 6' \Rightarrow 2') \Rightarrow \omega\omega_0\left(\frac{\zeta_0}{\zeta} + \frac{\zeta}{\zeta_0}\right) = \omega_0^2 \Rightarrow$$

$$\Rightarrow \boxed{\frac{\omega}{\omega_0} = \left(\frac{\zeta_0}{\zeta} + \frac{\zeta}{\zeta_0}\right)^{-1} = \frac{\zeta\zeta_0}{\zeta^2 + \zeta_0^2}} \quad (7)$$

där denna formel uttrycker vid vilket ω systemet övergår från att vara stabilt till instabilt.

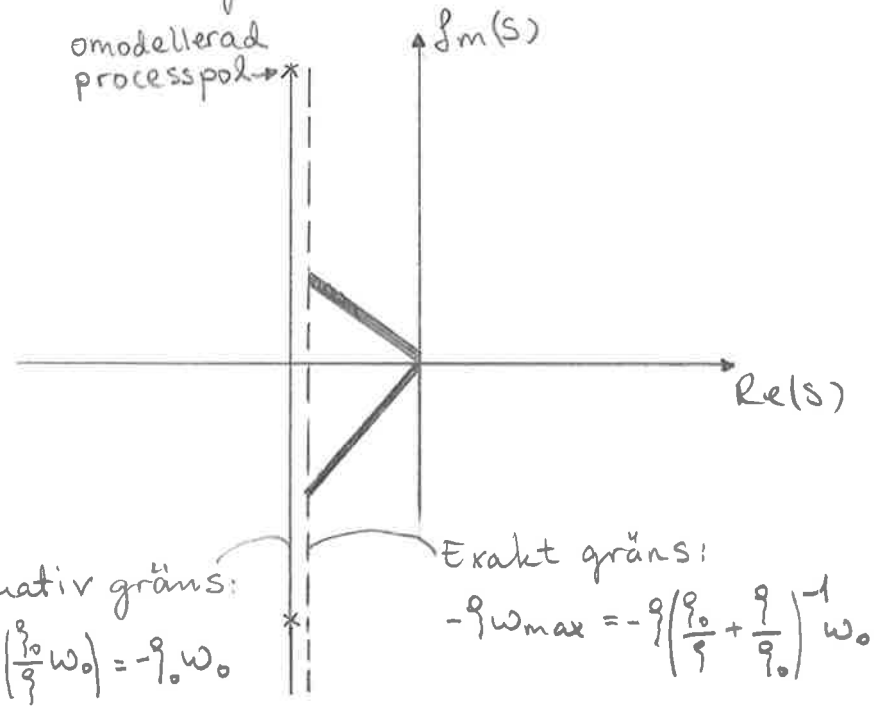
Formel (7) kan approximeras

$$\kappa_0 = \frac{\omega}{\omega_0} = \left(\frac{\zeta_0}{\zeta} + \frac{\zeta}{\zeta_0}\right)^{-1} = \left(\frac{\zeta_0}{\zeta}\left(1 + \left(\frac{\zeta}{\zeta_0}\right)^2\right)\right)^{-1} = \frac{\frac{\zeta}{\zeta_0}}{1+\left(\frac{\zeta}{\zeta_0}\right)^2} < \frac{\frac{\zeta}{\zeta_0}}{\left(\frac{\zeta}{\zeta_0}\right)^2} = \frac{\zeta_0}{\zeta}$$

$\frac{\zeta_0}{\zeta} < 1$ för dåligt dämpad högfrekvent stördynamik

Figur 3.2 visar approximationens avvikelse ifrån det verkliga värdet α_0. Dvs för $\frac{\zeta}{\zeta_0} > 5$ gäller approximationen bra.

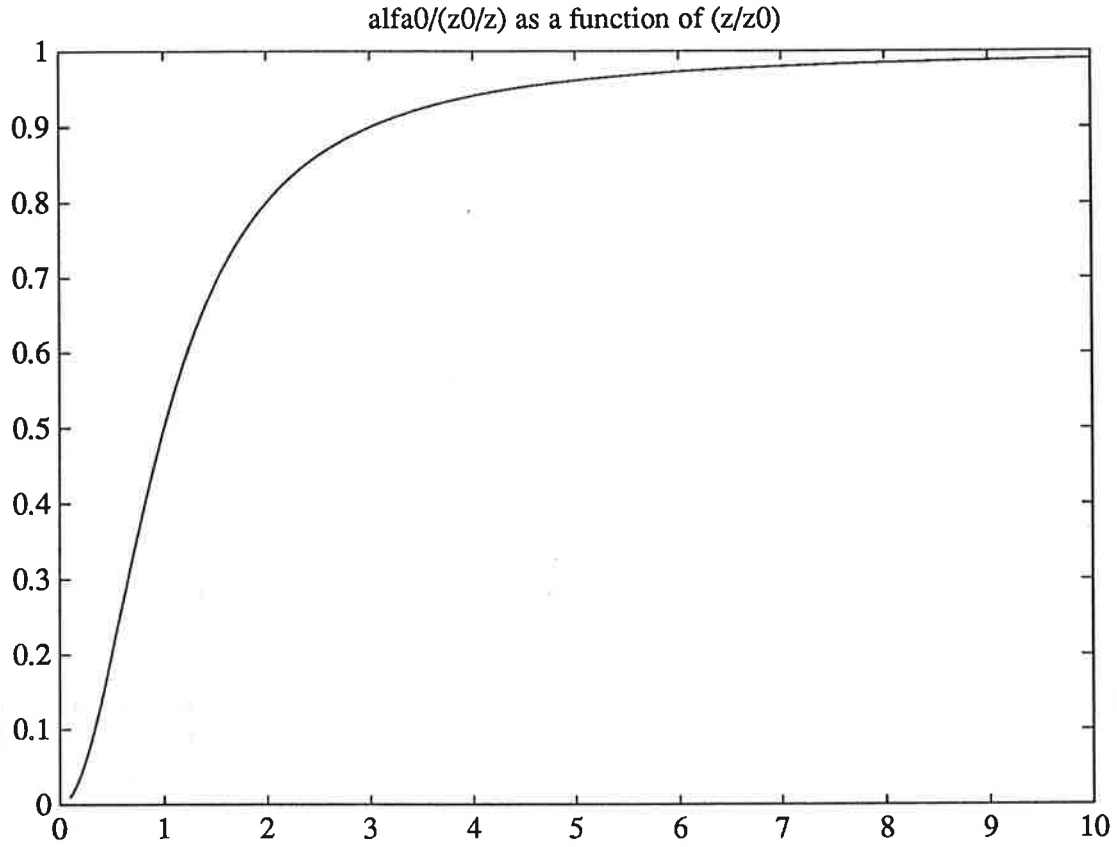
I figur 3.1 visar de grövre ritade strecken tillåten polplacering för nominella slutna systemet, som ger stabilitet.



De komplexkonjugerade störpolernas rundring mot positiva reella axeln undersöktes. Figur 3.3 visar ett linjärt beroende mellan realdelen av stördynamikens poler och regulatorns önskade slutna design-frekvens.

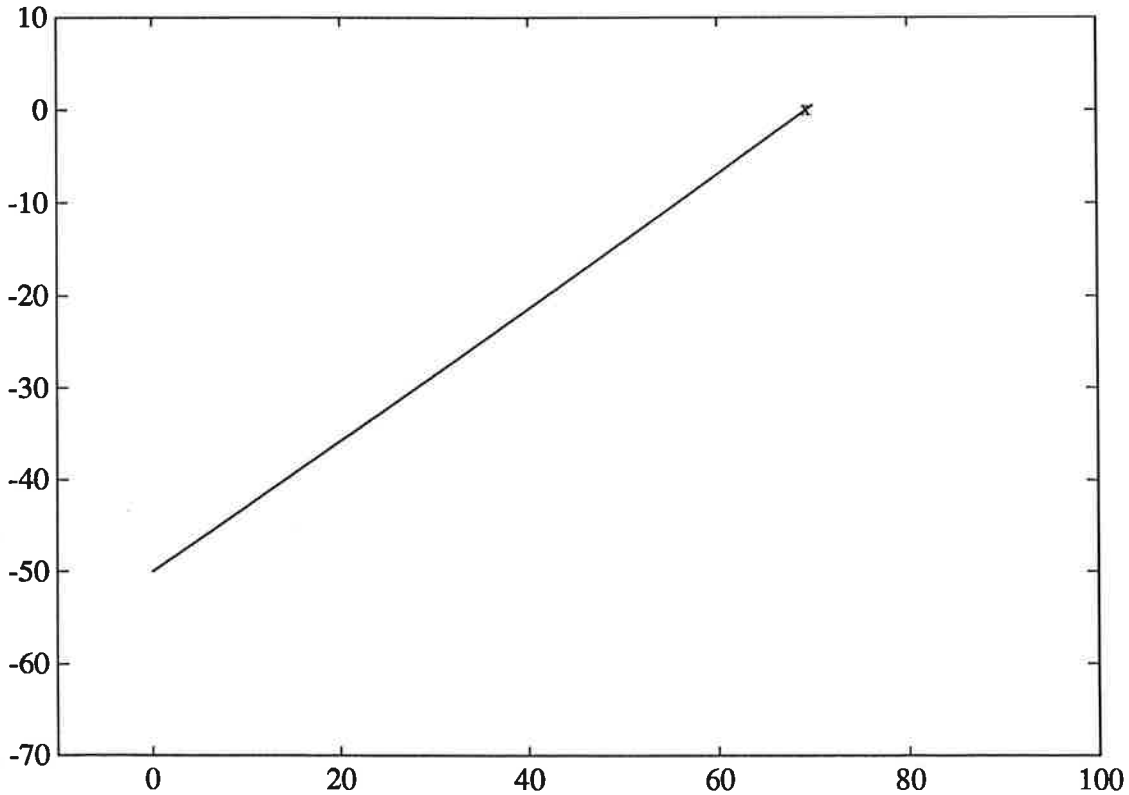
För exemplet: $\begin{cases} \omega_0 = 500 \text{ rad/s} \\ \zeta_0 = 0.1 \\ \zeta = 0.707 \end{cases} \Rightarrow \omega = \frac{0.1 \cdot 0.707}{0.1^2 + 0.707^2} \cdot 500 = 69.334$

ser man i figur 3.3 att realdelen av stördynamikens poler är noll vid $\omega \simeq 69$ och systemet bör därför bli instabilt för större värden på ω.



Figur 3.2

Realdelen av stördynamikens poler som funktion av regulatorns designfrekvens



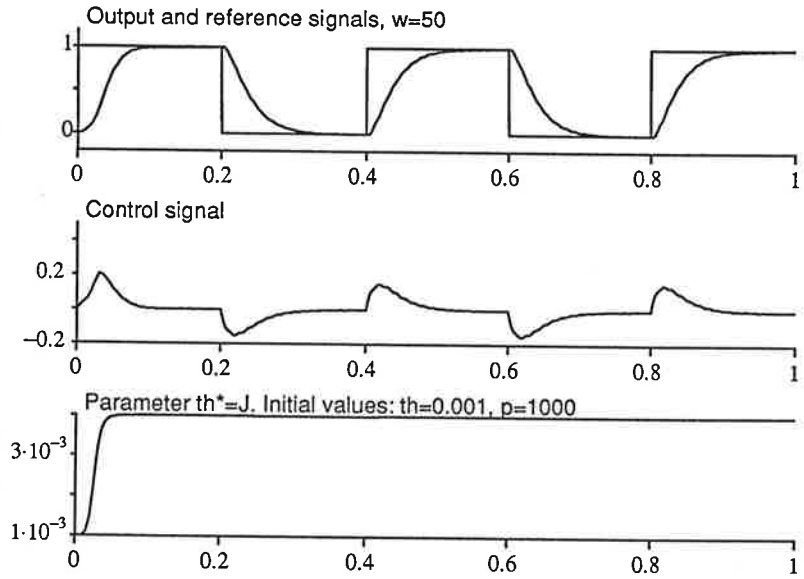
Figur 3.3

4 Simulering av systemet och utvärdering av de båda sätten att estimera motorns tröghetsmoment

I kapitel 2 med tillhörande appendix A beskrives systemstrukturen. Kapitel 3 analyserar systemets önskade slutna frekvens som en funktion av störpolernas dämpning och snabbhet. Se nedan.

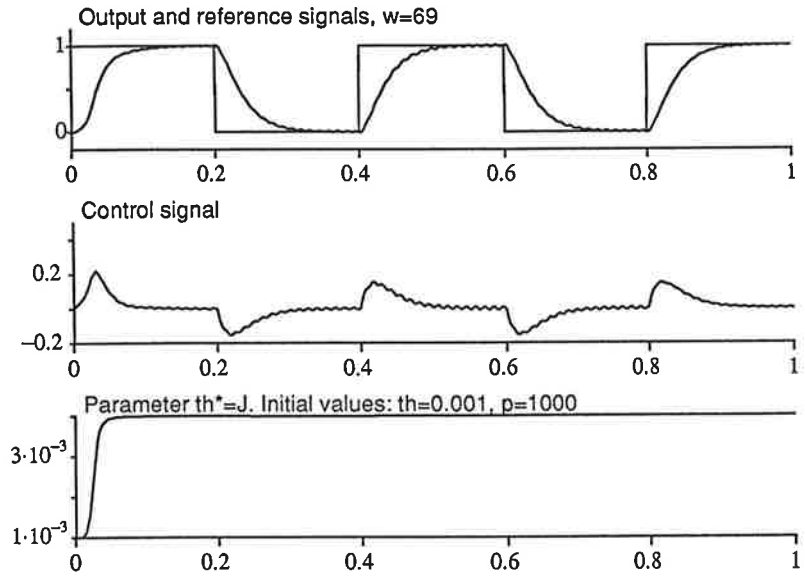
$$\frac{\omega}{\omega_o} = \frac{\zeta\zeta_o}{\zeta_o^2 + \zeta^2} \quad \begin{bmatrix} \zeta = 0.707 \\ \zeta_o = 0.1 \\ \omega_o = 500 \end{bmatrix} \Rightarrow \boxed{\omega_{limit} = 69.334}$$

Ovanstående värden användes i följande simuleringar med ett maximalt ω på 69 rad/s. Programkoden återfinnes i appendix B. Jag börjar med att titta på parameterestimatorn, där θ=J och endast processens insignal filtreras. Figur 4.1 visar en simulering där θ initialt är lägre än det verkliga, som är 0.004.

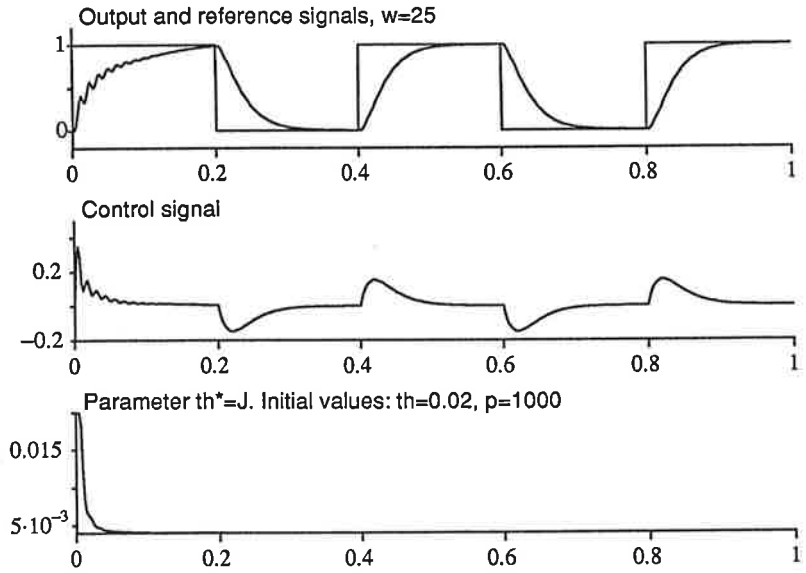


Figur 4.1

Man kan se att tröghetsmomentet svänger in sig till rätt värde relativt fort. Om man då ökar ω från 50 till 69 rad/s visar figur 4.2 att systemet ligger nära gränsen för instabilitet. θ svänger emellertid in lika snabbt som i föregående fall. Förutsättningarna ändras på ett markant sätt, om man närmar sig med ett initialt θ som är större än det verkliga. I figur 4.3 ser man att utsignalen ifrån processen till en början är mer oscillativ, trots att systemets snabbhet har minskats till 25 rad/s. Då det skattade tröghetsmomentet är för stort, blir K för stort, vilket leder till det oscillativa beteendet i början av simuleringen.

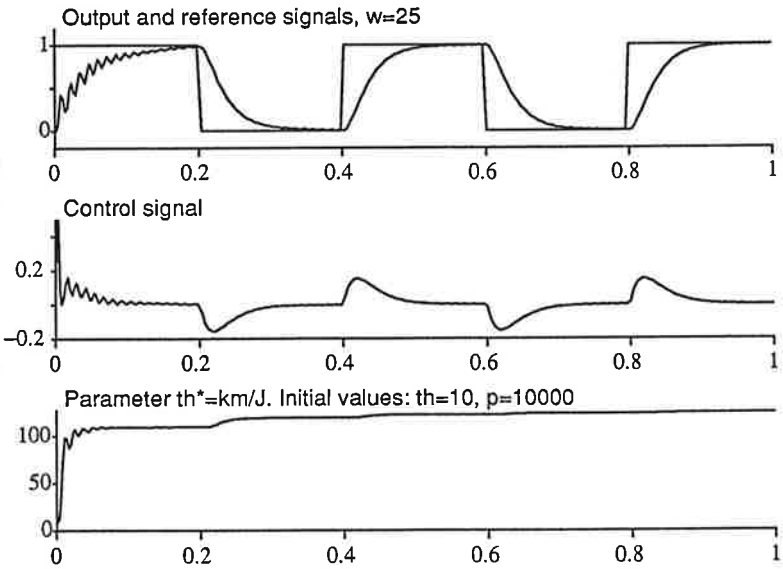


Figur 4.2

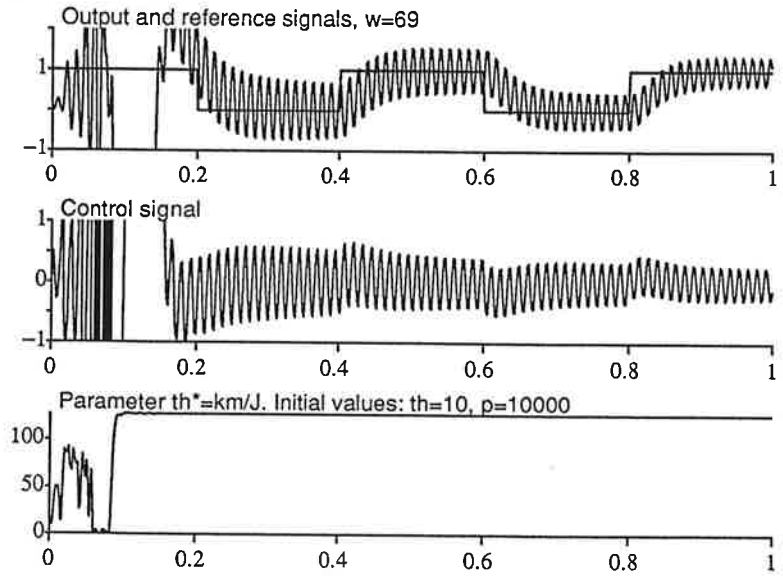


Figur 4.3

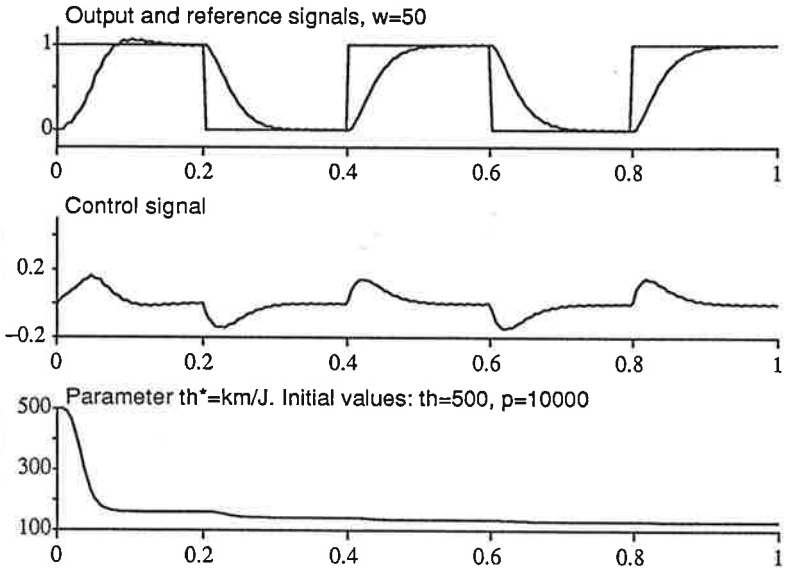
Den andra estimatorn (parest2) filtrerar både in- och utsignalen ifrån processen. Tröghetsmomentet beräknas som J=km/θ, vilket ger ett verkligt θ på 125 då motorkonstanten km=0.5. Figur 4.4 visar simuleringen av motsvarigheten till figur 4.3. Dvs om man börjar med ett θ som är lägre än det verkliga, eller ett J som är högre än det verkliga, så blir utsignalen ganska oscillativ även för låga ω. Med ökande frekvens ökar även oscillationernas amplitud, vilket ses i figur 4.5. Detta beror på att störpolerna exciteras lättare, dvs θ närmar sig noll och J och K närmar sig oändligheten.



Figur 4.4



Figur 4.5



Figur 4.6

Ovanstående bild visar en simulering där θ initialt är större än det verkliga. Utsignalen är som väntat mindre oscillativ, medan parameterskattaren är aningen långsammare. Detta kan kanske härröra från att skattaren behöver energi i systemet, såsom oscillationer eller till/frånslagningar, för att snabbt hitta till rätt värde på parametern.

5 Sammanfattning

Rapporten behandlar framför allt två avsnitt i samband med adaptiv motorstyrning. Den första delen analyserar vad omodellerad högfrekvent och resonant dynamik har för begränsande inverkan vid val av PI-regulatorns design-frekvens. För detta ändamål har en formel härletts. Den andra delen består av simuleringar i Simnon för två olika kontinuerliga skattare av motorns tröghetsmoment. Den första skattaren LP-filtrerar processens utsignal med hög brytfrekvens för motverkan av eventuell drift. Däremot passerar processens in- och utsignal ett första ordningens LP-filter för den andra skattaren. Ett högre ordningens filter borde kanske ha implementerats, eftersom simuleringarna för de båda skattarna inte avviker speciellt ifrån varandra. Vad som däremot gäller för de båda skattarna är, att valet av den skattade parameterns initialvärde har viss betydelse i början av den simulerade utsignalen beroende på hur parametern påverkar regulatorn.
Felmeddelandet "beräkningsfel" gav lite problem vid simuleringen. Det avhjälptes emellertid med en förkortad simuleringstid.
Avslutningsvis kan tilläggas att det varit roligt att arbeta med detta projekt.

Man lär så länge man lever,
men man lever så länge man lär.

Appendix A:1

Processmodell: $G = \frac{km}{Js}$

Process: $G_p = G \cdot G_1 = \frac{km}{Js} \cdot \frac{\omega_0^2}{s^2 + 2\zeta_0\omega_0 + \omega_0^2}$

Referensmodell: $G_m = \frac{\omega_m^2}{s^2 + 2\zeta_m\omega_m s + \omega_m^2}$

$$\hat{G}^{-1} = \frac{Js}{km} = \frac{s}{\omega_m} \cdot \frac{J\omega_m}{km}$$

Återkoppling: $G_{PI} = K\left(1 + \frac{1}{T_I s}\right)$

slutna systemet:

$$u = u_{ff} + u_{fb} = \hat{G}^{-1} G_m u_c + G_{PI}(y_m - y)$$

$$y = G_p \cdot u = G_p \hat{G}^{-1} G_m u_c + G_p G_{PI}(y_m - y) =$$

$$= G_1 G \hat{G}^{-1} G_m u_c + G_1 G G_{PI}(y_m - y)$$

$$(1 + G_{PI} G_1 G)\, y = G_1 G \hat{G}^{-1} G_m u_c + G_{PI} G_1 G\, y_m =$$

$$= G_1 G \hat{G}^{-1} G_m u_c + G_{PI} G_1 G G_m u_c =$$

$$= (G\hat{G}^{-1} + G_{PI} G) G_1 G_m u_c$$

$$y = \frac{(G\hat{G}^{-1} + G_{PI} G)}{(1 + G_{PI} G_1 G)} G_1 G_m u_c = \left[\text{om} \begin{cases} G_1 = 1 \\ \hat{G}^{-1} = \frac{1}{G} \end{cases} \right] = G_m u_c$$

slutna systemets kar. poly:

(1) ... $1 + G_{PI} G_1 G = 1 + K\left(1 + \frac{1}{T_I s}\right) \cdot \frac{\omega_0^2}{s^2 + 2\zeta_0\omega_0 s + \omega_0^2} \cdot \frac{km}{Js} = 0$

nominella kar. poly:

$$s^2 + \frac{km}{J} K\left(s + \frac{1}{T_I}\right) = s^2 + \frac{km}{J} k\, s + \frac{km\,K}{J\,T_I}$$

jfr. med: $s^2 + 2\zeta\omega s + \omega^2$ ger $\begin{cases} K = \frac{2\zeta\omega J}{km} \\ T_I = \frac{2\zeta}{\omega} \end{cases}$ insatt i (1) ger:

slutna systemets slutliga kar. poly:

$$s^4 + 2\zeta_0\omega_0 s^3 + \omega_0^2 s^2 + 2\zeta\omega\omega_0^2 s + \omega^2\omega_0^2$$

Appendix A:2

Regulatorn:

Framkoppling:

$$\begin{cases} X_1(s) = Y_m(s) = G_m R(s) = \dfrac{\omega_m^2}{s^2 + 2\zeta_m \omega_m s + \omega_m^2} R(s) \\ X_2(s) = \dfrac{s}{\omega_m} X_1(s) \\ u_{ff}(s) = \dfrac{J\omega_m}{k_m} X_2(s) \end{cases}$$

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & \omega_m \\ -\omega_m & 2\zeta_m\omega_m \end{bmatrix} x + \begin{bmatrix} 0 \\ \omega_m \end{bmatrix} r \\ u_{ff} = \begin{bmatrix} 0 & \dfrac{J\omega_m}{k_m} \end{bmatrix} x \end{cases}$$

Återkoppling:

$$\begin{cases} u_{fb}(s) = K\left(1 + \dfrac{1}{T_I s}\right) = K \cdot E(s) + I(s) \\ I(s) = \dfrac{K}{T_I s} E(s) \\ E(s) = X_1(s) - Y(s) \end{cases}$$

$$\begin{cases} u_{fb} = K \cdot e + i \\ di = \dfrac{K}{T_I} \cdot e \\ e = x_1 - y \end{cases} \quad \text{med} \quad \begin{cases} K = \dfrac{2\zeta\omega J}{k_m} \\ T_I = \dfrac{2\zeta}{\omega} \end{cases} \quad \text{insatt ger:}$$

$$\begin{cases} u_{fb} = \dfrac{2\zeta\omega J}{k_m} \cdot e + i \\ di = \dfrac{J\omega^2}{k_m} \cdot e \\ e = x_1 - y \end{cases}$$

För att motverka integral-uppvridning läggs en term till di.

$di = \dfrac{J\omega^2}{k_m} e + (u - v)/T_t$ där $\begin{cases} v = u \text{ om } |v| < i_{max} \text{ annars} \\ \quad |v| = i_{max} \\ T_t = \dfrac{T_I}{n} \end{cases}$

Appendix A:3

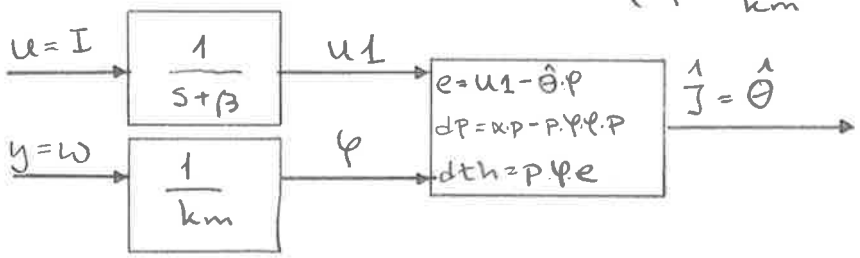
De båda estimatorernas uppbyggnad:

parest: $I = \frac{J}{km} \cdot \frac{dw}{dt}$

$$\frac{1}{s} I = \frac{J}{km} \cdot w$$

med integratorn ersatt av LP-filtret $\frac{1}{s+\beta}$ för att hindra offset-drift.

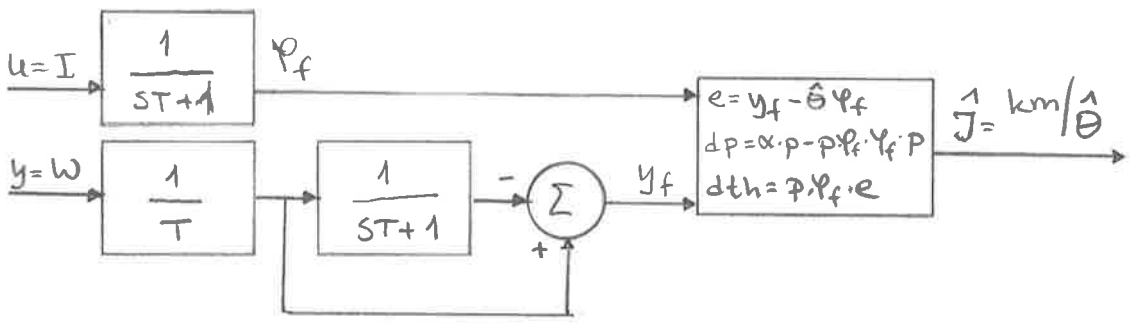
$$\frac{1}{s+\beta} I = J \cdot \frac{w}{km} \quad \text{där} \begin{cases} \bar{y} = \frac{1}{s+\beta} I = \frac{u}{s+\beta} = u1 \\ \hat{\theta} = \hat{J} \\ \varphi = \frac{w}{km} \end{cases}$$



parest2: $\frac{dw}{dt} = \frac{km}{J} I$; båda insignalerna LP-filtreras

$$\frac{1}{sT+1} \cdot sw = \frac{km}{J} \cdot \frac{1}{sT+1} I$$

$$\begin{cases} y_f = \frac{s}{sT+1} w = \frac{1}{T} \frac{sT+1-1}{sT+1} w = \frac{1}{T} w - w_f \\ w_f = \frac{1}{T} \frac{1}{sT+1} w \\ \varphi_f = \frac{1}{sT+1} I \\ \hat{\theta} = \frac{km}{J} \end{cases}$$



Appendix B:1


```
CONTINUOUS SYSTEM proc

INPUT u
OUTPUT wm J
STATE x1 x2 x3
DER dx1 dx2 dx3

dx1 = - 2*z0*w0*x1 - w0*w0*x2 + w0*w0*u
dx2 = x1
dx3 = if simple then u else x2
wm = km*x3/J

"consants
simple:0
z0:0.1
w0:500
km:0.5
"J:0.004
J=0.004

END


CONTINUOUS SYSTEM reg
"PI regulator for motor control
"special version with feedforward and gain scheduling
"filename PIF.t

input y r J
output u
state i  x1   x2
der  di dx1  dx2

"Gain calculations
k=2*z*w*J/ki
Ti=2*z/w
kff=J*wm/ki
Tt=Ti/n

"Feedforward
dx1=wm*x2
dx2=wm*(-x1-2*zm*x2+r)
uff=kff*x2

"Feedback
e=x1-y
v=k*e+i+uff
u=if v<-imax then -imax else if v<imax then v else imax
di=k*e/Ti+(u-v)/Tt
ufb=u-uff

"parameters
imax:30    "current limit
ki:0.5
w:50
z:0.707
wm:50
zm:1
n:5

END
```

Appendix B:2


```
MACRO figx5
"dcmotor-simulation
SYST proc reg parest con
SPLIT 3 1
AXES H 0 1.0 V -0.2 1.2
TEXT 'Output and reference signals, w=50'
STORE wm[proc] u[reg] r[reg] uff ufb Je
PLOT wm[proc] r[reg]
INIT th:0.2
INIT p:1000
"INIT th:0.001
"INIT p:1000
SIMU 0 1
AXES V -0.2 0.5
TEXT 'Control signal'
SHOW u
AXES V 0.004 0.02
TEXT 'Parameter th*=J. Initial values: th=0.2, p=1000'
SHOW Je
END


CONNECTING SYSTEM con

"connecting system for motor simulation

TIME t

r=1
u[proc]=u[reg]
y[reg]=wm[proc]
r[reg]=IF mod(t,tp)<tp/2 THEN uo else 0
J[reg]=IF estim THEN Je[parest] ELSE J[proc]
"J[reg]=Je[parest]
"u[filt]=u[proc]
u[parest]=u[proc]
y[parest]=wm[proc]
estim:1
uo:1
tp:0.4
END


CONTINUOUS SYSTEM parest

"File called parest.t
"This parameter estimator estimates the total moment of inertia
"Author M Tischbierek 920505

INPUT  u y
OUTPUT Je
STATE  u1 th p
DER    du1 dth dp

"Step 1 PARAMETER ESTIMATION

"1.1 Compute the residual
e=(u1-y/ke*th)

"1.2 Compute fi
fi=y/ke

"1.3 Update estimates
```

Appendix B:3


```
dth=p*fi*e

"1.4 Update covariances
dp=alfa*p-p*fi*fi*p

"1.5 Filter u
du1=u-beta*u1

"1.6 Update regression vector fi

"1.7 Update moment of inertia
Je=th

"Parameters
pi:3.1415926

"Specification parameters
alfa:0.5            "alfa>=0
beta:1e-4           "filterconstant

"Initial values
th:0.0189
u1:0
ke:0.5
p:10

end
```

Appendix B:4


```
MACRO simx5
"dcmotor-simulation
SYST proc reg parest2 con2
SPLIT 3 1
AXES H 0 1 V -1 2
TEXT 'Output and reference signals, w=50'
STORE wm[proc] u[reg] r[reg] uff ufb th[parest2]
PLOT wm[proc] r[reg]
INIT th:10
INIT p:10000
SIMU 0 1
AXES V -1 1
TEXT 'Control signal'
SHOW u
AXES V 0 128
TEXT 'Parameter th*=km/J. Initial values: th=10, p=10000'
SHOW th[parest2]
END


CONNECTING SYSTEM con2

"connecting system for motor simulation

TIME t

r=1
u[proc]=u[reg]
y[reg]=wm[proc]
r[reg]=IF mod(t,tp)<tp/2 THEN uo else 0
J[reg]=IF estim THEN Je[parest2] ELSE J[proc]
"J[reg]=Je[parest2]
"u[filt]=u[proc]
u[parest2]=u[proc]
y[parest2]=wm[proc]
estim:1
uo:1
tp:0.4
END


CONTINUOUS SYSTEM parest2

"File called parest2.t
"This parameter estimator estimates the total moment of inertia
"Author M Tischbierek 920514

INPUT  u y
OUTPUT Je
STATE th p wf fif
DER dth dp dwf dfif

"Step 1 PARAMETER ESTIMATION where th=ke/J

"1.1 Compute the residual
e=yf-fif*th

"1.2 Compute fi

"1.3 Update estimates
dth=p*fif*e

"1.4 Update covariances
dp=alfa*p-p*fif*fif*p
```

Appendix B:5


```
"1.5 Filter y
dwf=-wf/T+y/(T*T)
yf=y/T-wf

"1.6 Update regression vector fi, but a filtered one
dfif=-fif/T+u/T

"1.7 Update moment of inertia
Je=ke/th

"Parameters
pi:3.1415926

"Specification parameters
alfa:2          "alfa>=0

"Initial values
th:500
p:100
ke:0.5
T:1e-2

end
```

ROTORT.M

```
z0=0.1
w0=500
z=0.707
w=69.334
aa=[1 2*z0*w0 w0^2 2*z*w*w0^2 w^2*w0^2]
r=roots(aa)
realval(1)=max(real(r(find(abs(r)==max(abs(r))))))
axis([-10, 100, -70, 10])
plot(w,realval(1),'x')
hold on
w=0:0.5:70
for k=1:length(w)
aa=[1 2*z0*w0 w0^2 2*z*w(k)*w0^2 w(k)^2*w0^2]
r=roots(aa)
realval(k)=max(real(r(find(abs(r)==max(abs(r))))))
end
plot(w,realval)
```

Neural Network Control of a Nonlinear Process based on a Steepest Descent Backpropagation Technique

Mattias Gyllerup,
undergraduate student at the Department of Automatic Control,
Lund Institute of Technology, Lund, SWEDEN

June 10, 1992

1 Introduction

The title of this paper[1] is chosen in order to somewhat de-mystify the concept of neural networks. They're actually nothing more than quadratic error minimizer using the salient features of the non linear network architecture. Although the present theoretical in-depth knowledge of the neural nets is so far quite scarse one should not be afraid to experiment cautiously with them.

The main purpose of this paper is to reproduce a neural network simulation based on a control scheme proposed in an article by Fu-Chuang Chen[1]. The reproduction is somewhat simplified by putting a greater constraint on the non linear process which in this case means that the control designer has a greater knowledge about the process than in the original article.

Apart from the main purpose there have also crystallized further objectives during the process of model building and simulation. These were not stated initially but are important aspects to take into consideration.

The first and most urgent objective is to give critics on the original article. It is doubtlessly written with the best of intentions but shows just how hard a reproduction can be. I still have some questions as to what the author Mr. Chen really has done. These question will be described throughout the paper as they arise and summarized in the concluding section.

The second objective is to discuss if a neural network really is needed in the posed problem. This discussion takes the form of trying out two other types of regulators.

2 About the process

The nonlinear process to be controlled as proposed in [1], is

$$y_{t+1} = 0.8sin(2y_t) + 1.2u_t$$

where u_t is the input signal and y_t is the output signal.

If this process for some reason is unknown to the control engineer we may apply the described neural net control system[1]. Although it's more realistic to gather some a priori knowledge about the process before actually controlling. Hence I prefer to test whether a simple digitally implemented proportional integrating controller or an ordinary self tuning

[1]This paper is the result of a project carried out within the course *Adaptive Control* and was supervised by Prof. Karl-Johan Astrom

regulator perhaps are as likely to manage the process. If so, then a neural network is far to advanced for the application. The desired

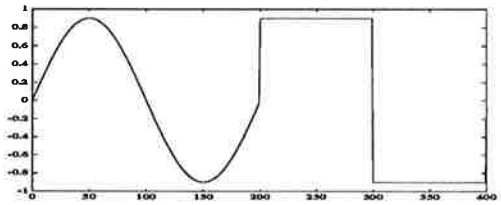


Figure 1: The command signal

command signal, uc, that the process is supposed to track is shown in figure 1. Notice however that this signal differs from the signal given in the original paper by a factor of 0.7 in amplitude. This is done due to the fact that I couldn't manage to reproduce the simulations to reach the same results as in the paper with an amplitude of 1.3, see figure 2. Changing the amplitude to 0.9 made the system act properly.

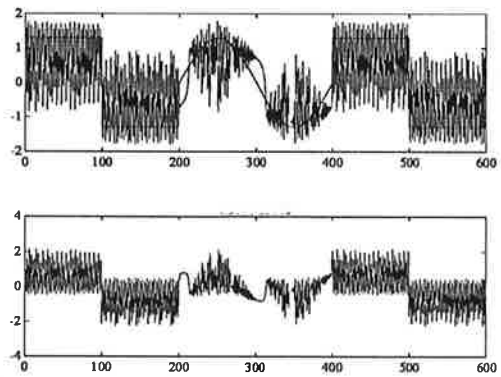


Figure 2: After 60000 samples: The "scary" scenario when trying to reproduce the original paper using an amplitude of 1.3

3 Controlling with a PI-controller

As mentioned in the previous section I want to make sure that an ordinary controller won't do the job. I have therefore implemented a polynomial PI-regulator in the following way:

The process can be written in the general form

$$y_{t+1} = a \sin 2y_t + bu_t$$

Linearization around zero gives

$$y_{t+1} = ay_t + bu_t + \delta$$

Ignoring δ and using z as a forward shift operator leaves us with

$$y_t(z-a) = bu_t \Rightarrow$$

$$y_t = \frac{b}{z-a}u_t \Rightarrow$$

$$A = z - a, B = b$$

Straight forward calculations for the following control law

$$Ru_t = Tuc_t - Sy_t$$

gives us a deadbeat regulator with PI-features if

$$AR + BS = A_0Am$$

where $A_0 = z$, $A = z$, $R = z - 1$, $S = \frac{1+a}{b}z - \frac{a}{b}$ and $T = S$.

However, it turns out that T cannot equal S due to instability in the system. Instead T takes the form of a first order polynomial without a zero order term.

The simulations carried out in simnon code shows that the PI-regulator does not quite manage to handle the process, figure 3. This gives me an incentive to try an adaptive approach.

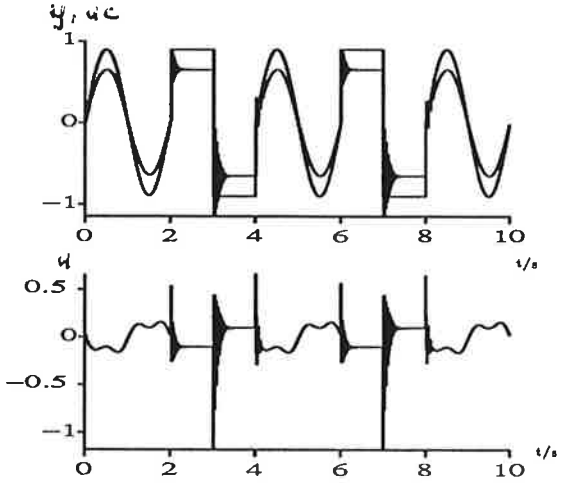


Figure 3: Controlling with a PI-regulator

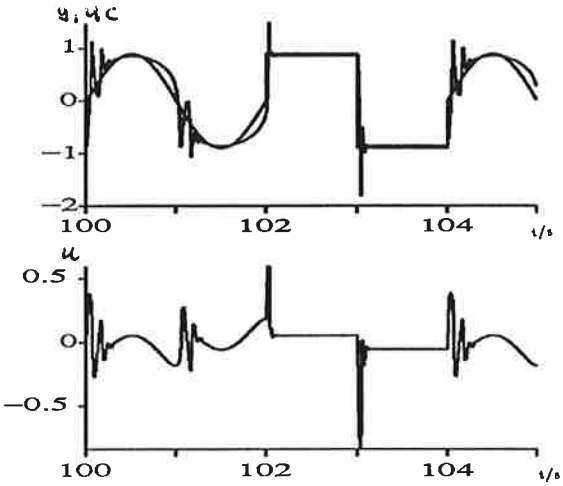


Figure 4: Controlling with a self tuning regulator

4 Self Tuning Regulator

A simple test letting a self tuning regulator control the non-linear process shows that after 10 000 samples it will not get any better than in figure 4. The result indicates that perhaps a neural net could come in handy.

5 The Neural Control System

In the two previous sections I have shown that the process cannot be controlled with simple regulators. In order to make them work, gain-scheduling or non linear feedback are possible ways to carry on. In doing so more thorough identification of the process is needed before designing and implementing any of the above. The neural cotrol system hereby provide a unique alternative since it not only controls the process but also identifies it during the control task. This means that we can apply a 'general' neural regulator to any type of unknown process and make things happen, although this is of course not the whole truth.

Figure 5 shows how the system is built. Notice that this system is somewhat different from what the original article suggests, at least pictorial. In the original illustration the neural net is not fed by the process output and hence cannot produce any function estimates. I believe that this is only a typographic error but it is somewhat misleading and makes things more difficult to understand. Further

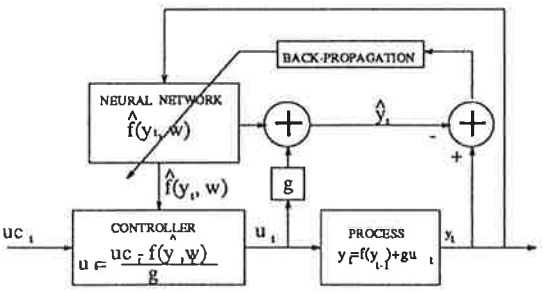


Figure 5: System design

differences is due to my simplification assuming that g is a known constant, and that I prefer to subtract $\hat{y}_t$ from y_t instead of the re-

verse.

5.1 The Neural Network

I assume that the reader is familiar with neural networks and will only submit a brief description on my notations in order to eliminate any confusion.

- t always denotes time dependence. It's either put in parenthesis or as an index.
- index i and j always helps denoting variabels

The neurons used in the model are of two basic classes: The tan-sigmoid without bias and the linear with or without bias. Figure 6 shows the two neuron types. The transfer functions for the neurons are differentiable and monotonicly increasing which offers a way to avoid traps in neuron minima. Only network minima, though local they may be, are possible. The network used in this specific control

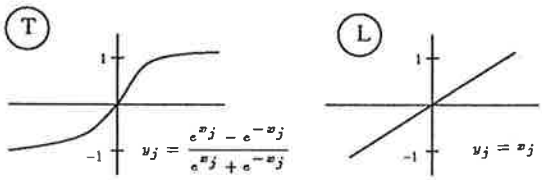


Figure 6: The neuron label and it's transfer function for a left: tan-sigmoid neuron, and right: linear neuron

system is connected as shown in figure 7. It contains 20 tan-sigmoid neurons divided into two hidden layers and uses linear neurons in the input and output layer. The reason for using linear neurons in the output layer is that tan-sigmoid neurons limits the output range to operate only between -1 and 1. The linear neurons does not pose such limitations and can therefore be used to bias the output signal to an appropriate level.

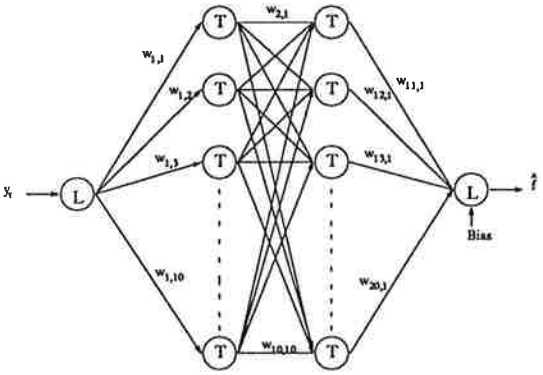


Figure 7: The neural network

5.2 Back-Propagation Algorithm

The best way to look upon the backpropagation neural network as I see it, is to think of the system as a general function approximator. The backpropagation learning rule offers a way to change the weights and biases so that the sum squared error of the network reaches a minima. It is nothing more than the ordinary steepest descent approach implemented to facilitate the structure of the network.

The most apparent drawback is that the error-surface may contain a local minima and that the network will lock on to it. Thus the gradient method does not come with any guarantee to find the global minima.

I will now show how the backpropagation algorithm is obtained for this specific network. The total sum of inputs x_j to a single neuron at level j in the network structure can be expressed as

$$x_j = \sum_i y_i w_{ij} \tag{1}$$

The output y_j from the neuron is a function of the neurons total input x_j. In the case of a

tan-sigmoid neuron the output becomes

$$y_j = \frac{e^{x_j} - e^{-x_j}}{e^{x_j} + e^{-x_j}} \quad (2)$$

The error between the desired and the actual neuron output is defined as

$$E = \frac{1}{2}(d_j - y_j)^2$$

The reason for this is the well known properties of quadratic functions. The intention is to minimize the error E by gradient descent approach in changing the weigths and we therefore require it's partial derivatives. They are then to be propagated back from the output layer to the input layer of the network. For ease of exposition I will show all the calculations required to reach the final formula.

$$\frac{\partial E}{\partial y_j} = d_j - y_j$$

$$\frac{\partial E}{\partial x_j} = \frac{\partial E}{\partial y_j}\frac{dy_j}{dx_j}$$

Taking the derivative of the tan-sigmoid function (2) and substituting into the above equation gives

$$\frac{\partial E}{\partial x_j} = (d_j - y_j)(1 - y_j^2)$$

This equation shows how a change in the input x_j will affect the error E. Taking it from here to see how a change in the weights and from the layers behind will affect the error is now easy since the input is a linear combination of them, see equation (1).

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial x_j}\frac{\partial x_j}{\partial w_{ij}} = \frac{\partial E}{\partial x_j}y_i = (d_j - y_j)(1 - y_j^2)y_i$$

The steepest descent method now gives the following simple rule for updating the weights in order to minimize the error E

$$w_{ij}(t+1) = w_{ij}(t) + \mu\frac{\partial E}{\partial w_{ij}}$$

5.3 Implementing the system

The task of the neural network is to be able to produce a good estimate of how the process will respond to different sets of input signals. This feature is then used in order to give the process the input it requires to follow the command signal uc.

$$u_t = -\frac{\hat{f}(y_t, w_t)}{g} + \frac{uc_t}{g} \quad (3)$$

The output of the process will due to this feedback depend on the weights of the neural network and serve as the desired output of the neural network.

$$y_{t+1} = f(y_t, w) + g\{-\frac{\hat{f}(y_t, w_t)}{g} + \frac{uc_t}{g}\}$$

If (3) is substituted into the following equation (4), which describes the systems process estimate, this estimate will become independent of the weights w_t and will equal the desired command signal uc.

$$\hat{y}_{t+1} = \hat{f}(y_t, w_t) + gu_t \quad (4)$$

The next step is to define the output error to be

$$E_t = \frac{1}{2}e_t^2 = \frac{1}{2}(y_t - uc_t)^2$$

The error derivative due to the weights will now beocme

$$\frac{\partial E_t}{\partial w_{ij}} = \frac{\partial y_t}{\partial w_{ij}}e_t$$

where

$$\frac{\partial y_t}{\partial w_{ij}} = \{\frac{df(y_t)}{dy_t} + \frac{\partial u_t}{\partial y_t}\}\frac{\partial y_t}{\partial w_{ij}} + \frac{\partial u_t}{\partial w_{ij}}$$

However the author of the original article ignores the first term in the expression within the braces and thus, with help from (3), gets

$$\frac{\partial E_t}{\partial w_{ij}(t)} = -\frac{\partial \hat{f}(y_t, w_t)}{\partial w_{ij}(t)} e_t \qquad (5)$$

where

$$\frac{\partial \hat{f}(y_t, w_t)}{\partial w_{ij}(t)} = (1 - y_j(t)^2) y_i(t) \qquad (6)$$

This derivative is not expressively stated in the original paper which I think is a sad misstake. Instead it is referred to as using a backpropagation technique with "...some minor modifictions..". It would be interesting to see if this is the key difference between my system and the original work that makes it impossible for me to fully reproduce the simulations.

Equation (5) is the same as the one appearing in the backpropagation section and the updating rule will therefore be the same.

5.4 Results

After experimenting with different types of learning rate, μ, I found that μ in the proximity of 0.5 works fairly well. If it is set to be 0.95 as in the original paper a wild oscillatory behaviour can be observed every time the command signal turns into a square wave and sometimes instability occurs. Figures 8 through 10 shows an example of how the control system behaves when it behaves well.

In the original paper a discussion on whether the neural net is actually learning the process dynamics or not concludes that if the scalar μ is sufficiently small the error will decrease as the network learns. However the paper does not support any comparison between the process and the tuned network. I think that it's highly relevant to see how well the net manage to identify the process and therefore let figure 11 show the functions for the process with a zero input signal, the tuned net and a single neuron. Notice that in the inter-

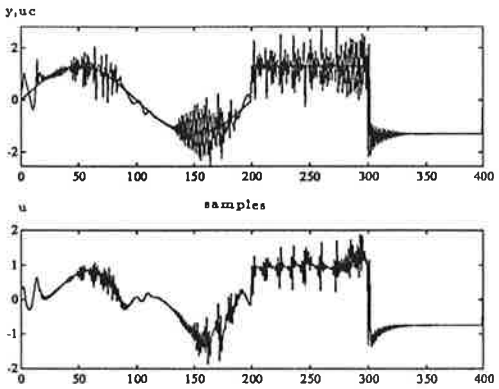


Figure 8: The first 400 samples

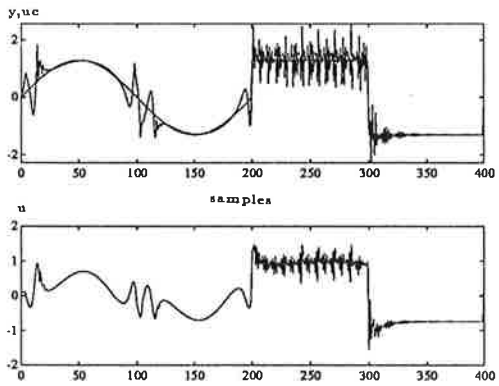


Figure 9: After 2000 samples

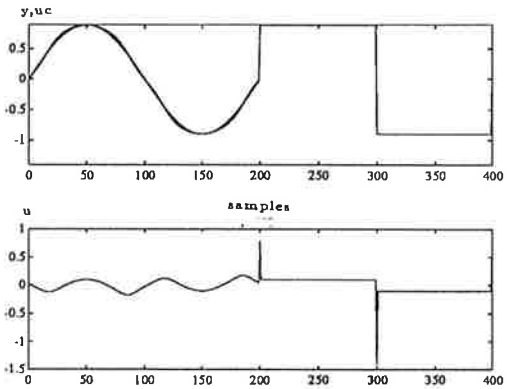


Figure 10: After 100 000 samples

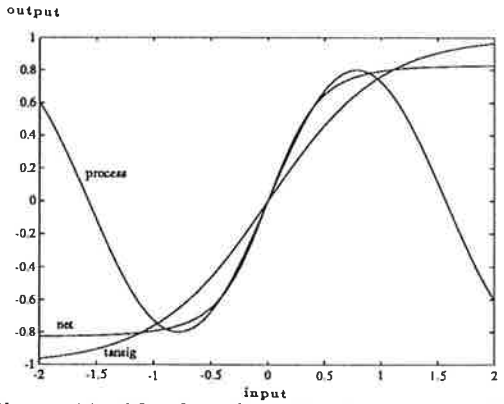


Figure 11: The functions for the process, the tuned net and a single neuron

val [-0.9 0.9] the neural network looks somewhat alike the proces although it's far from perfect. Also notice the resemblance between a single tan-sigmoid neuron function and the process function in the interval [-0.9 0.9]. Perhaps this is one of the reason why the neural control system works in this interval but not in the outside interval.

6 Conclusions

Having shown that the non linear process described in [1] cannot be controlled by neither a simple PI-regulator nor a self tuning regulator I try out the neural network regulator designed in [1] but fail to reproduce the exact results.

There are two major issues of concern that I would like to point out as essential to the reproduction.

- The key formula for the updating rule is not given explicitly in [1]. This is not only ignorant to the reader but makes it impossible to know if the reproduction is carried out correctly.
- The rate at which μ decreases is not specified in [1] which makes it hard to reproduce the rate of convergence.

The overall conclusion that I draw from my experience in working with neural networks in control systems is that the system is heavily dependent on receiving a good initial μ-value. Having constantly failed to reproduce the experiment with the same magnitude on the μ-scalar as in the original description I'm still in doubt as to whether the article properly describes the authors approach.

Finally I would like to point out that one might have divided feelings about working with neural networks. First of all it's very hard to understand what's really going on although you're still amused by the academic challenge the approach offers. Secondly you're having a hard time trying to figure out how on earth this would work in real life. You don't have to be a Nobel Prize Winner to understand that when confronting a unknown nonlinear process with a desire to control it, one seeks thorough in-depth knowledge of the process. And when this knowledge is achieved a tailored system is much more suitable than an on-line identification from scratch. However, as a process identifier only, the neural net may work very well.

References

[1] Chen, Fu-Chuang, *Back-Propagation Neural Networks for Nonlinear Self-Tuning Adaptive Control,* IEEE Control Systems Magazine, Vol 10, No. 3, 1990

A Neural Network Matlab Code

```
function [usave, ysave, ucsave] = bpreg(epoch)

% epoch is the number of samples to work through
% the function returns the control signal, the process output
% and the reference signal for each sample

% Copyright (c) 1992 by Mattias Gyllerup and
% Department of Automatic Control,
% Lund Institute of Technology, Lund, SWEDEN

%%%%%%%%%%%%%%%%%%%%%%
%% initial setup of parameter space %%
%%%%%%%%%%%%%%%%%%%%%%

%amplitude of uc
a=0.9;

% lambda factor, after 20000 samples lr=lr0/50
la=-log(1/50)/20000;

%period length
T=200;

%network parameters
R=1;
Q=1;
S1=1;
S2=10;
S3=10;
S4=1;

% init weight values
W1=rands(S1,R);

W2=rands(S2,S1)*0.1;

W3=rands(S3,S2)*0.1;
size(W3)
W4=rands(S4,S3)*0.1;
B4=rands(S4,1)*0.1;

%error'goal
eg = 0.00002;

%learning rate
lr0 = 0.5;
lr = lr0;

%init desired process output
uc=0;
%init control signal
u=0;
%init process output
y=0;

%repeat the following code until epcoh is reached
for t=1:epoch

%%%%%%%%%%%%%%%%%%%%
%% step 1:estimate the functions %%
%%%%%%%%%%%%%%%%%%%%

% feedthrough without bias in hidden layer
%obs andras dessa funktioner maste aven bp phase andras
A1 = y;
A2 = tansig(W2*A1);
A3 = tansig(W3*A2);
A4 = purelin(W4*A3,B4);

% estimate the nonlinear functions
fhat=A4;
ghat=1.2; % assume g known for ease !


%%%%%%%%%%%%%%%%%%%%%%%%
%% step 2: generate the control signal %%
%%%%%%%%%%%%%%%%%%%%%%%%

% fetching the reference signal
uc=ref(a,t,T);
ucsave=[ucsave uc];

% calculate the new control signal
u=(uc-fhat)/ghat;
usave=[usave u];

%%%%%%%%%%%%%%%%%%%%%%%%%
%% step 3: calculate the process output %%
%%%%%%%%%%%%%%%%%%%%%%%%%

y=proc(y,u);
ysave=[ysave y];

%%%%%%%%%%%%%%%%%%%
%% step 4:update the neural net %%
%%%%%%%%%%%%%%%%%%%

% error between reference signal and actual output
e = y-uc;

% determine the error to be backpropagated
E = y-ghat*u-fhat;


%SSE = sumsqr(E);

% check if need to change weights
%if SSE ¿ eg

 % backpropagate
   D4 = deltalin(A4,E);
   D3 = deltatan(A3,D4,W4);
   D2 = deltatan(A2,D3,W3);

 % update the weights
   [dW2] = learnbp(A1,D2,lr);
   [dW3] = learnbp(A2,D3,lr);
   [dW4,dB4] = learnbp(A3,D4,lr);
   W2 = W2 + dW2;
   W3 = W3 + dW3;
   W4 = W4 + dW4; B4 = B4 + dB4;


%end %if SSE ¿ eg
```

```
% administration
if rem(t,400) == 0
 lr=lr0*exp(-la*t);
 !rm info.sim
  diary info.sim
  diary on
  t
  lr
  diary off

  save ysave
  save usave
  save ucsave

%  clg
%  subplot(211)
%  plot(ucsave)
%  hold on
%  plot(ysave)
%  hold off
%  plot(usave)


 ysave=[];
  usave=[];
  ucsave=[];

end

end %for t=1:epoch



function ynew = proc(y,u)

% function ynew = proc(y,u)
%
% nonlinear process as described in
% IEEE vol10 no3 1990 page 46


% (c)1992 by Mattias Gyllerup and
% Department of Automatic Control,
% Lund Institute of Technology, Lund, SWEDEN

ynew=0.8*sin(2*y)+1.2*u;


function [uc]=ref(a,t,T)

% function [uc]=ref(a,t,T)
%
% producing consecutive blocks of the reference signal
% given in IEEE vol10 no3 1990 page 47 figure 4
%
% a is the amplitude of the reference signal
% t is the specific time point(s) given in samples
% T is the period length given in samples

% (c)1992 by Mattias Gyllerup and
% Department of Automatic Control,
% Lund Institute of Technology, Lund, SWEDEN

f=1/T;

m=rem(t,2*T);

% sinus shape
m1=find(m¡T);
if ~isempty(m1)
     uc(m1)=a*sin(2*pi*f*t(m1));
end

% upper part of square wave
m2=find(m¿T-1 & m¡T+T/2);
if ~isempty(m2)
     uc(m2)=a*ones(m2);
end

% lower part of square wave
m3=find(m¿(T+T/2)-1);
if ~isempty(m3)
     uc(m3)=-a*ones(m3);
end
```

B PI Regulator Simnon Code

```
DISCRETE SYSTEM PIREG

state u1 u2 y1 y2 uc1 uc2
new nu1 nu2 ny1 ny2 nuc1 nuc2

input y uc
output u

time t
tsamp ts

g=-r1*u1-r2*u2-s0*y-s1*y1-s2*y2+t0*uc+t1*uc1+t2*uc2

u=MIN(MAX(g,-100),100)

nu1=u
nu2=u1
ny1=y
ny2=ny1
nuc1=uc
nuc2=uc1

ts=t+h

h:0.01

r1:-1
r2:0
s0:1.5
s1:-0.67
s2:0
t0:0.60
t1:0
t2:0

END
```

```
DISCRETE SYSTEM proc

INPUT u

OUTPUT y

STATE x
NEW nx

TIME t
TSAMP ts

"difference equation
nx=0.8*sin(2*x)+1.2*u
y=x

"update sample time
ts=t+h

"sample step
h:0.01

"process parameters
a:0.8
b:1.2

END


DISCRETE SYSTEM REF

OUTPUT uc

TIME t

TSAMP ts
"frequency
f=1/l

"omega
w=6.283*f

"correction of period length due to sample rate
l=h*p

m=MOD(t,2*l)
d=a*sin(w*t)
uc=IF m¡l THEN d ELSE IF (m¿l-h AND m¡l+l/2) THEN a ELSE -a

"update sample time
ts=t+h

"sample rate
h:0.01

"period length
p:200

"amplitude
a:0.9

END

CONNECTING SYSTEM con

TIME t

uc[pireg]=uc[ref]
y[pireg]=y[proc]
u[proc]=u[pireg]

END
```

C Self Tuning Regulator Simnon Code

```
DISCRETE SYSTEM str

INPUT uc y

OUTPUT u

"estimate
STATE  theta1  theta2  theta3  theta4
NEW    ntheta1 ntheta2 ntheta3 ntheta4

"regressors
STATE phi1  phi2  phi3  phi4
NEW  nphi1 nphi2 nphi3 nphi4

"P-matrix
STATE p11  p12  p13  p14
STATE p22  p23  p24
STATE p33  p34
STATE p44
NEW  np11 np12 np13 np14
NEW  np22 np23 np24
NEW  np33 np34
NEW  np44

"regulator states
STATE  udelay  ydelay  ucdelay
NEW    nudelay nydelay nucdelay

"filter states
STATE  yf1  yf2
NEW    nyf1 nyf2

"sampling par.
TIME t
TSAMP ts


"  Step 1 ; Estimation


" Some initwork ...
ptheta1=p11*phi1+p12*phi2+p13*phi3+p14*phi4
ptheta2=p12*phi1+p22*phi2+p23*phi3+p24*phi4
ptheta3=p13*phi1+p23*phi2+p33*phi3+p34*phi4
ptheta4=p14*phi1+p24*phi2+p34*phi3+p44*phi4

temp=phi1*ptheta1+phi2*ptheta2
temp=temp+phi3*ptheta3+phi4*ptheta4
```

```
" Caclulate K
k1=ptheta1/(1+temp)
k2=ptheta2/(1+temp)
k3=ptheta3/(1+temp)
k4=ptheta4/(1+temp)



" Filtering
yf=y+a0*am1*yf1+a0*am2*yf2

" Calculate the error
error=yf-phi1*theta1-phi2*theta2
error=error-phi3*theta3-phi4*theta4


" Update the P-matrix
np11=p11-ptheta1*k1
np12=p12-ptheta1*k2
np13=p13-ptheta1*k3
np14=p14-ptheta1*k4
np22=p22-ptheta2*k2
np23=p23-ptheta2*k3
np24=p24-ptheta2*k4
np33=p33-ptheta3*k3
np34=p34-ptheta3*k4
np44=p44-ptheta4*k4


" Update the estimate
ntheta1=theta1+k1*error
ntheta2=theta2+k2*error
ntheta3=theta3+k3*error
ntheta4=theta4+k4*error

"Update the filter
nyf1=y
nyf2=yf1

"update the regressor via shift
nphi1=u
nphi2=phi1
nphi3=y
nphi4=phi3

" assigning the estimates
r0=ntheta1
r1=ntheta2
s0=ntheta3
s1=ntheta4

"Step 2 - Calculating the control signal
u=(t0*uc+t1*ucdelay-s0*y-s1*ydelay-r1*udelay)/r0

"Update the regulator via shift
nucdelay=uc
nudelay=u
nydelay=y

" Update the sample time
ts=t+h

" Constants
h:0.01

a0:1

am1:-1.0646
am2:0.3679

t0:0.1769
t1:0.1264

END


DISCRETE SYSTEM proc

INPUT u

OUTPUT y

STATE x
NEW nx

TIME t
TSAMP ts

"difference equation
nx=0.8*sin(2*x)+1.2*u
y=x

"update sample time
ts=t+h

"sample step
h:0.01

"process parameters
a:0.8
b:1.2

END


DISCRETE SYSTEM REF

OUTPUT uc

TIME t

TSAMP ts
"frequency
f=1/l

"omega
w=6.283*f

"correction of period length due to sample rate
l=h*p

m=MOD(t,2*l)
d=a*sin(w*t)
uc=IF m¡l THEN d ELSE IF (m¿l-h AND m¡l+l/2) THEN a ELSE -a

"update sample time
ts=t+h

"sample rate
```

```
h:0.01

"period length
p:200

"amplitude
a:0.9

END

CONNECTING SYSTEM con

TIME t

uc[str]=uc[ref]
y[str]=y[proc]
u[proc]=u[str]

END
```