Escola Universitària
Politècnica de Mataró

Centre adscrit a:

UNIVERSITAT POLITÈCNICA
DE CATALUNYA

Enginyeria Tècnica Industrial: Especialitat Electrònica Industrial

APLICACIONS DE CONTROL PER RETORN D'ESTAT AMB TARJA D'ADQUISICIÓ DE DADES I VISUAL BASIC

Memòria

PAU SALOMÓ CASANOVAS
PONENT: MIQUEL ROCA

TARDOR 2011

TecnoCampus
Mataró-Maresme

Resum

L'objectiu del projecte consisteix en poder controlar un sistema en temps real mitjançant una targeta d'adquisició de dades. Els resultats més significatius serien les diferències que hi ha segons el tipus de controlador. Els dos sistemes estudiats són un motor i una placa Feedback composta de 2 LAGs i un integrador. El motor respon correctament en la major part dels controladors, ja que és un sistema lent però el millor control seria el retorn d'estat complet. I en el sistema del Feedback respon millor amb un control per retorn d'estat amb observador predictiu, ja que és un sistema ràpid i no es pot capturar correctament les variable d'estat.

Resumen

El objetivo del proyecto consiste en poder controlar un sistema en tiempo real a través de una tarjeta de adquisición de datos. Los resultados más significativos serían las diferencias que hay según el tipo de controlador. Los dos sistemas estudiados son un motor y una placa Feedback compuesta de 2 LAGs y un integrador. El motor responde correctamente en la mayor parte de controladores ya que es un sistema lento, de todas formas el mejor control para este sistema seria el retorno de estado. El sistema Feedback responde mejor con un control de retorno de estado con observador predictivo, ya que es un sistema rápido i no se pueden capturar correctamente las variables de estado.

Abstract

The target of our project is to control a real time system through an data acquisition card. Most relevant results are the differences between types of controllers.

An engine and a Feedback board, made of 2 LAG's and an integrator, are the two systems studied in this project. The engine responds properly with most of the drivers because it is a slow system. However, the best driver/controller for this system is the state return. The Feedback system works well with a state return controller with predictive observer, since it is a fast system and it is not possible to correctly catch the state variables.

Índex.

Índex de figures.

Índex de taules.

Glossari de termes.

TCM	Tecno Campus Mataró
NI	Nacional Intruments
A/D	Analògic / Digital
SCADA	Supervisory Control And Data Acquisition (Control de supervisió i adquisició de dades)
DAQ	Adquisició de dades
IRQ	Petició Rutina Interrupcions
Timer	Temporitzador
INT	Integrador
Mp	Pols màxim
Ts	Temps d'estabilització
ITAE	Integral Time absolute error (Error de temps absolut integral)
Ω	Velocitat
σ	Posició

1. Objectius.

1.1. Propòsit.

Desenvolupar varies aplicacions a traves del programa "Visual Basic" per controlar un sistema a traves de una targeta d'adquisició de dades.

1.2. Finalitat.

Estudiar la resposta de les targetes d'adquisició treballant en temps real, a partir de controladors per retorn d'estat.

1.3. Objecte.

Estudiar el comportament de controladors per retorn d'estat a traves de targetes d'adquisició de dades de Nacional Instruments i el seu comportament treballant en temps real. Estudiar les respostes de diferents controladors segons el seu temps de mostreig i amb les seves formes diferents d'implementació.

1.4. Abast.

S'especifica la manera de crear una aplicació de control per retorn d'estat i la seva configuració per poder comunicar amb diferents targetes d'adquisició de dades. S'explica com calcular controladors per retorn d'estat i la seva implementació segons el tipus de sistema.

2. Estudi de les targetes d'adquisició de dades.

2.1. Funcionament de les targetes NI PCI-6014.

La targeta NI PCI-6014 és una targeta d'adquisició de dades a traves del bus PCI. En aquest punt es pretén explicar de forma breu, les possibilitats que tenen aquestes targetes en el moment de controlar un sistema.

2.1.1. Introducció a la targeta d'adquisició NI PCI-6014.

Aquest tipus de targetes s'utilitzen per poder capturar dades analògiques i digitals externes amb un computador, per poder analitza-les i interactuar amb el món exterior.

La targeta NI PCI-6014 és una targeta fabricada per l'empresa Nacional Instruments dotada d'un bus PCI. Aquesta targeta té una sèrie de característiques principals:

- 1 Convertidor A/D amb 16 entrades multiplexades

- 2 Convertidors D/A

- 8 Entrades i sortides digitals TTL

- 2 Temporitzadors de 24 bits

Aquesta targeta és possible utilitzar-la amb diferents plataformes com: LabVIEW, LabWindows/CVI, Measurement Studio per Visual Basic, Visual Studio .NET i Matlab. En els següents apartats es tracten com utilitzar la targeta en algunes d'aquestes plataformes.

2.1.2 Mètodes d'utilització / programació

Molt genèricament, la utilització de la targeta en les plataformes: Matlab, Labview, Visual Basic, Visual Studio .NET.

2.1.3. Utilització a traves de Matlab.

Matlab és una aplicació utilitzada per calcula matrius, com el seu nom indica "MAT" de Matriu i "LAB" de laboratori. Aquest entorn de desenvolupament matemàtic integrat s'utilitza un llenguatge propi denominat Llenguatge M. Aquesta aplicació té la particularitat que pots anar instal·lant connectors denominats "Toolbox" que significa

caixa de eines. Dins de les innumerables Toolbox que existeixen, hi ha dues que son les més significatives que son Simulink i Guide.

- Simulink és un entorn on podem simular diferents sistemes i interactuar amb el hardware.
- Guide és un entorn de programació on es poden crear interfases d' usuari.

En aquest capítol tractarem d'explicar el funcionament del Simulink amb els toolbox de NI. Per poder utilitzar Simulink és necessari tenir comprar el ToolBox Data Acquisition Toolbox, que es pot trobar a: http://www.mathworks.es/products/daq/tryit.html

En primer lloc s'obre el Matlab i el simulink, es troba dins de la llibreria del simulink un apartat anomenat "Real time Windows Target" Fig.2.1. En aquesta llibreria es pot incorporar a una nova fulla de simulink entrades i sortides analògiques.

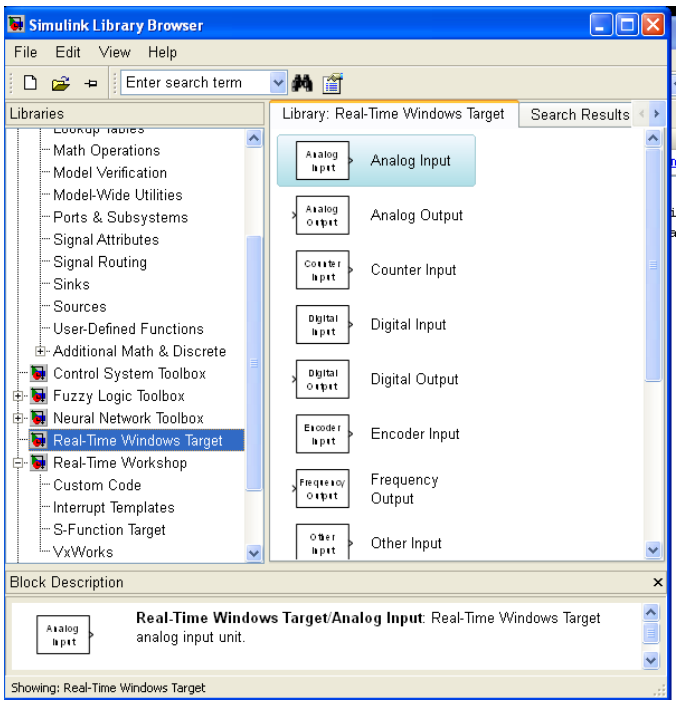


Fig. 2.1 Real time Windows target

Es selecciona la targeta dins del apartat de Nacional Instruments que es troba a la configuració de cada mòduls Fig. 2.2.

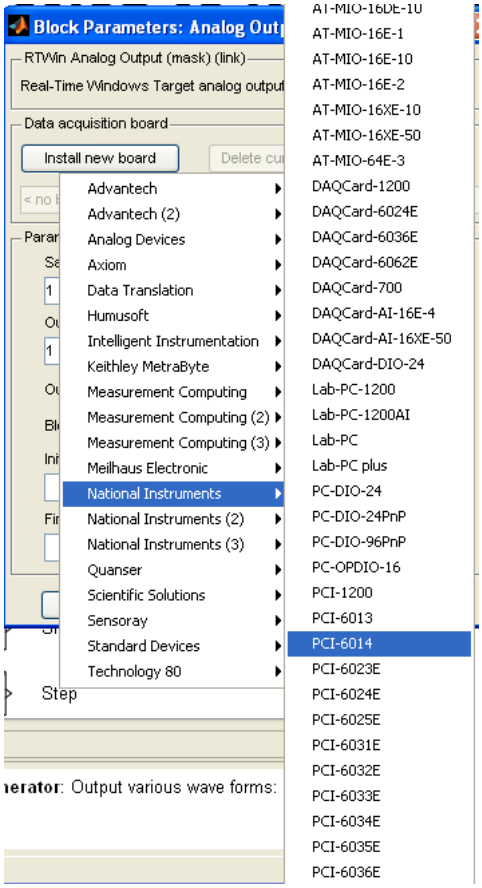


Fig. 2.2 Selecció de la targeta PCI-6014

Dins dels paràmetres de configuració es pot seleccionar quines entrades i sortides digitals s'utilitzaran, comptadors, i el nivell en entrada i sortida analògica Fig. 2.3.

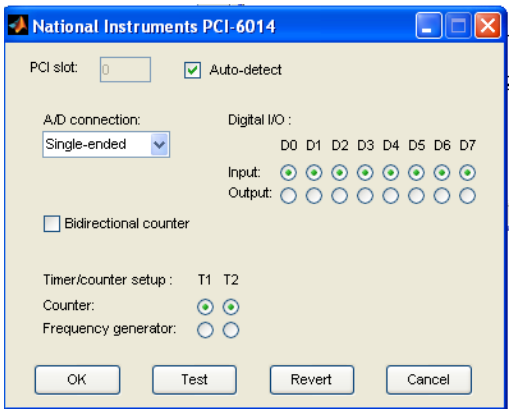


Fig. 2.3 Configuració Nacional Instruments

En aquest moment ja esta apunt per utilitzar les targetes en temps real desde el Simulink.

2.1.4.Utilització a traves de Labview

Labview és una aplicació de simulació i control. Es basa en un entorn de programació visual d'alt nivell. Aquest entorn és molt utilitzat en proves de Laboratori o sistemes SCADA.

Per poder utilitzar el Labview és necessari, descarregar de la pagina web de Nacional Instruments el Software: NI DAQmx. Es pot trobar de forma gratuïta en la direcció: http://joule.ni.com/nidu/cds/view/p/id/2126/lang/es

S'ha de tenir en compte que la llibreria per utilitzar la targeta és compatible nomes amb les versions: Labview 2010, Labview 2009, Labview 8.5.

En el següent exemple es pot veure com capturar un senyal analògic i presentar-la per pantalla com si fos un oscil·loscopi.

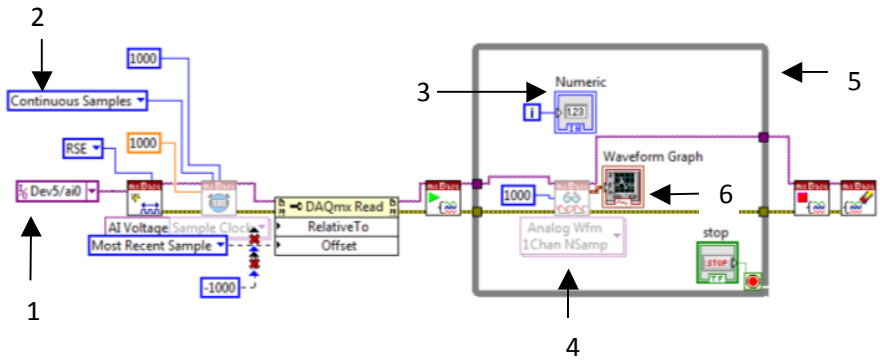


Fig. 2.4 Codi Labview adquisició continua

En la Fig. 2.4 es veu el codi de Labview amb el funcionament de l'adquisició. Hi ha una sèrie d'elements importants en el codi de la Fig. 2.4 que es ressalten amb una indicació.

1- Selecció del port i entrada analògica utilitzada per la captura del senyal.
2- Generador d'interrupcions.
3- Indicador de mostres capturades.
4- Captura del la senyal analògica de l'entrada 1 de la targeta.
5- Bucle que surt segons la condició de STOP ubicat a la interfase gràfica.
6- Sortida per la interfase gràfica.

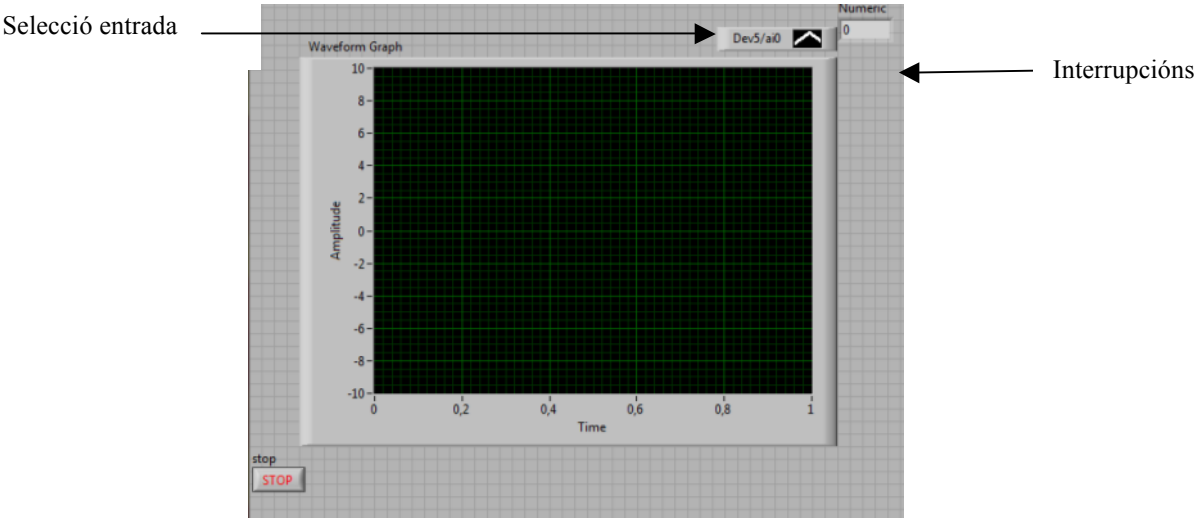


Fig. 2.5 Interfase gràfica Labview adquisició continua.

Com es pot veure en la Fig. 2.5 en la interfase gràfica, hi ha el boto de stop, una gràfica amb els següents eixos: amplitud de -10v a +10v i el temps en segons.

Per altra banda tenim un quadra de text on canviar la entrada i un segon quadra de text on es veuen la quantitat de mostres que s'han capturat fins el moment.

2.1.5. Programació a traves de Visual Basic

La programació a traves de Visual Basic es pot fer de dues maneres, controlant directament el hardware a traves d'instruccions o utilitzant uns controladors ActiveX que proporciona, el mateix fabricant Nacional Instruments. Per poder programa amb Visual Basic és necessari descarregar els drivers de pago des de la web de Nacional Instruments que estan ubicats a: http://joule.ni.com/nidu/cds/view/p/id/2604/lang/es

Aquest software és de demostració durant 30 dies.

S'ha de instal·lar el software amb les següents llibreries marcades de la Fig. 2.6.

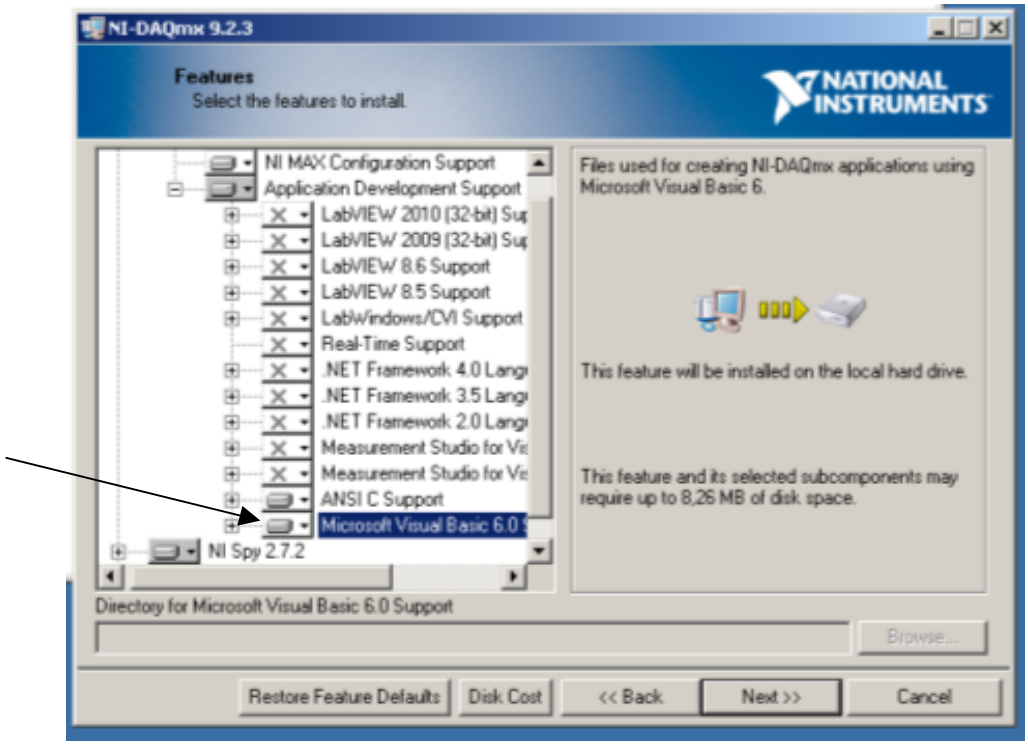


Fig. 2.6 Instal·lació Llibreria de Visual Basic 6.0

- Programació a traves de visual basic utilitzant el hardware directament:

S'utilitza una llibreria que es diu DAQmx on les funcions més utilitzades son:

DAQmxCreateAIVoltageChan: Seleccionar Canal

DAQmxReadAnalogScalarF64: Capturar dades analògiques

DAQmxStopTask: aturar la captura.

DAQmxCreateAOVoltageChan: Seleccionar canal de sortida

DAQmxWriteAnalogF64: Escriure el valor per la sortida analògica

DAQmxCreateCOPulseChanFreq: Configurar el temporitzador de la targeta

DAQmxCfgImplicitTiming: Activar el temporitzador de la targeta

Per més informació es pot trobar en el menú d'inici dins de Programes->Nacional Instruments-> NI-DAQ -> NI-DAQ Help. (Fig. 2.7)

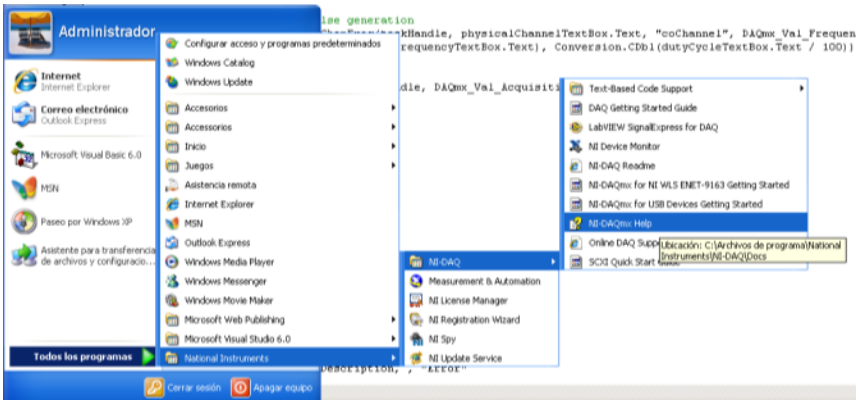


Fig. 2.7 Ajuda de la llibreria Visual Basic 6.0

- Programació a traves de ActiveX:

La utilització de controladors ActiveX ens permet tenir una manera fàcil les dades d'entrada de la targeta i visualitzar-les. Però per altre banda no es té un control del hardware. Com podem veure a la Fig. 2.8 en el Visual Basic s'afegeix un boto amb els controladors ActiveX.

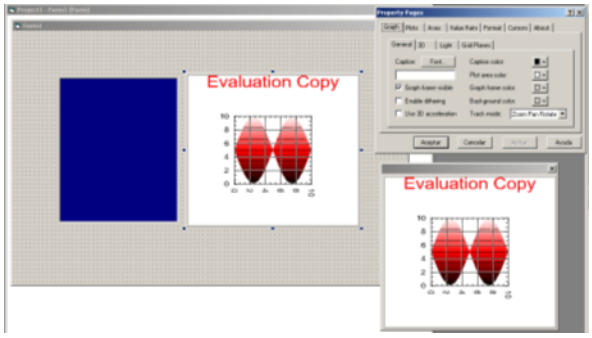


Fig. 2.8 Controlador ActiveX

2.1.6. Programació a traves de Plataforma .NET.

La plataforma .NET és l'evolució del llenguatge Visual Basic. La plataforma permet programar amb diferents llenguatges com: Visual C++, Visual C#, Visual J#, ASP.NET i Visual Basic .NET.

En aquest cas s'explicarà com utilitzar la targeta NI PCI-6014 amb el llenguatge Visual C++ .NET. Primer és necessari tenir instal·lat el Measurement Studio que es pots trobar una versió de prova de 30 dies a: https://lumen.ni.com/nicif/us/evalmstudio/content.xhtml

Per altra banda també és necessari tenir instal·lat els drivers NI-DAQmx 9.2.3 que es pot descarregar de: http://joule.ni.com/nidu/cds/view/p/id/2604/lang/es

Primer de tot s'ha de crear un projecte nou i incloure les llibreries de Nacional Instruments, tal com es mostra a la Fig. 2.9, dins de les referències del projecte.

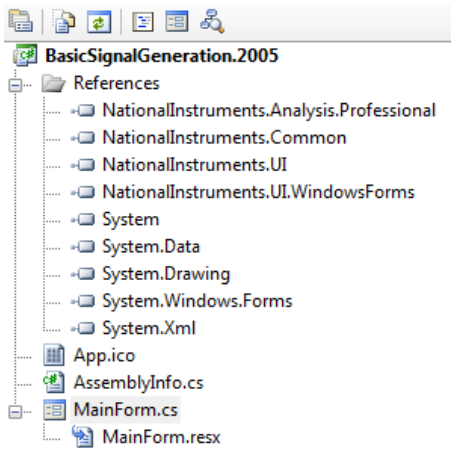


Fig. 2.9 Llibreries Nacional Instruments .NET

S'ha analitzar una sèrie de classes per utilitzar el convertidor Digital/Analògic:

Sortida analògica es pot utilitzar la classe: "**BasicFunctionGenerator"**, aquesta classe té una seria de paràmetres establers com poden ser "sine" es a dir Senoidal on hi ha que passar tot els paràmetres del senyal.

També son utilitzats els paràmetres: "Triangle", "Square", "Swatooth"...

També hi ha classes per llegir el valor de la entrada analògica:"**AnalogWaveform**".

Per altre banda es pot canviar la freqüència de mostreig:

```
dataSocket.Data.Value = data;
dataSocket.Data.Attributes["Frequency"].Value = frequencyValue;
dataSocket.Data.Attributes["Timestamp"].Value = timestampValue;
```

Per més informacions es pot consultar el web: http://www.ni.com/support/esa/ o si es vol consultar les classes de la llibreria i saber com utilitzar-les es pot consultar directament a la llibreria NacionalInstruments. Common. Tal i com es mostra a la Fig. 2.7.

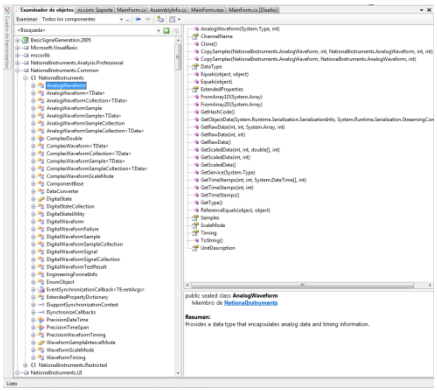


Fig. 2.10 Llista de classes de Nacional Intruments .NET

2.2. Funcionament de les targetes PC-LAB 8112-PG.

La targeta PC-LAB és una targeta d'adquisició de dades a traves del bus ISA. En aquest punt es pretén explicar de forma breu, les possibilitats que tenen aquestes targetes en el moment de controlar un sistema.

2.2.1. Introducció a la targeta de adquisició PC-LAB 8112-PG.

Aquest tipus de targetes s'utilitzen per poder capturar dades analògiques i digitals externes amb un computador, per poder analitza-les i interactuar amb el món exterior.

La targeta PC-LAB és una targeta fabricada per l'empresa Nacional Instruments dotada d'un bus ISA. Aquesta targeta és molt similar a l'anterior però és un model més antic.

A diferencia amb l'altre, les plataformes compatibles son menors però els drivers son gratuïts, es pot tenir el control del hardware directament.

Aquesta targeta es possible utilitzar-la amb les diferents plataformes son les següents: Matlab, Visual Basic 5.0.

2.2.2. Mètodes d'utilització / programació.

Molt genèricament, la utilització de la targeta en les plataformes: Matlab, Visual Basic 5.0.

2.2.3. Utilització a traves de Matlab.

Matlab és una aplicació utilitzada per calcula matrius, com el seu nom indica "MAT" de Matriu i "LAB" de laboratori. Aquest entorn de desenvolupament matemàtic integrat utilitza un llenguatge propi denominat Llenguatge M. Aquesta aplicació té la particularitat que es pot anar instal·lant connectors denominats "Toolbox" que significa caixa d'eines. Dins de les innumerables Toolbox que existeixen, hi ha dues que son les mes significatives que son Simulink i Guide.

- Simulink es un entorn on es pot simular diferents sistemes i interactuar amb el hardware.
- Guide és un entorn de programació on es poden crear interfases de usuari.
- En aquest capítol tractarà d'explicar el funcionament del Simulink amb els toolbox de NI.

Per poder utilitzar Simulink amb la targeta de adquisició de dades és necessari comprar el ToolBox Data Acquisition Toolbox, que es pots trobar a: http://www.mathworks.es/products/daq/tryit.html

En primer lloc s'obre el Matlab i el Simulink, es troba dins de la llibreria del Simulink un apartat anomenat "Real time Windows Target" Fig.2.11. En aquesta llibreria es pot incorporar a una nova fulla de Simulink entrades i sortides analògiques.

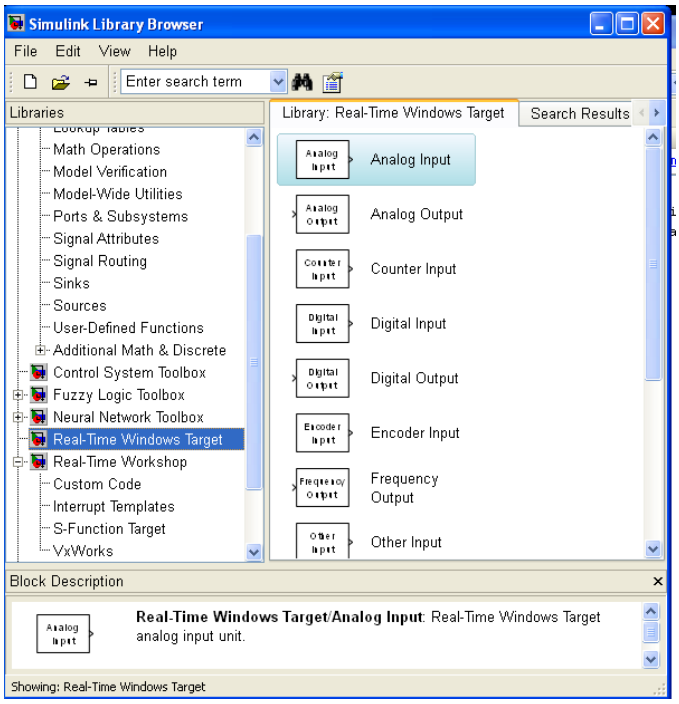


Fig. 2.11 Real time Windows target

Es selecciona la targeta dins del apartat de Nacional Instruments que es troba a la configuració de cada mòduls Fig. 2.12.

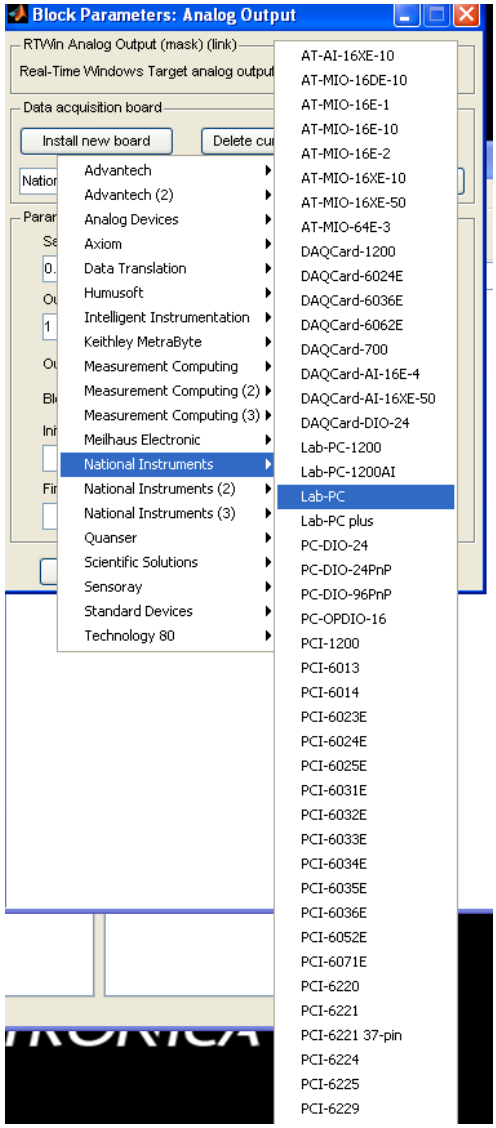


Fig. 2.12 Selecció de la targeta PCI-6014

Dins dels paràmetres de configuració es pot seleccionar quines entrades i sortides digitals s'utilitzaran, comptadors, i el nivell en entrada i sortida analògica Fig. 2.13.

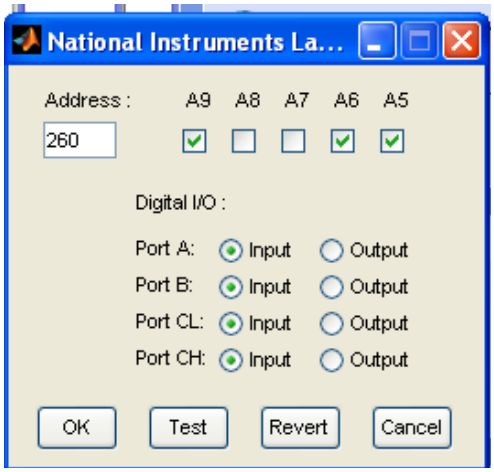


Fig. 2.13 Configuració Nacional Instruments

En aquest moment ja esta apunt per utilitzar les targetes en temps real des de el Simulink.

2.2.4. Programació a traves de Visual Basic

La programació a traves de Visual Basic 5.0 és possible gracies a uns drivers anomenats: LibVB. Primer de tot es guardar la carpeta LibVB a l'arrel del disc del sistema (normalment C). Posteriorment es copia l'arxiu "inpout32.dll" ubicat a la carpeta "C:\LibVB\Inpout32\per la carpert System32", a la carpeta "C:\WINDOWS\system32". Per altre banda es copia els arxius "TVicHW32" ubicat a "C:\LibVB\Tvichw32\per la capeta drivers" i el arxiu "TVicHW32.dll" ubicat a "C:\LibVB\Tvichw32\per la carpeta SYSTEM", el primer a la carpeta "C:\WINDOWS\system32\drivers" i el segon a "C:\WINDOWS\system", respectivament. Ara ja estan instal·lats els drivers de la targeta compatibles per Visual Basic 5.0.

Es crea un projecte nou de Visual Basic i es copien els arxius: Inpout32.OBJ i TVicLib.OBJ just a la carpeta superior del projecte. Dins del projecte s'afegim els dos objectes a dins de mòduls (Fig. 2.14)

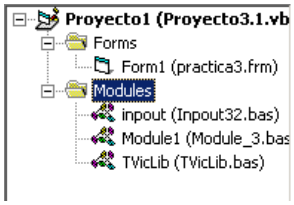


Fig. 2.14 Mòduls de visual basic

En aquest moment ja esta tot configurat per poder programa utilitzant la targeta. Nomes falta obrir els ports de la targeta amb la instrucció:

HW32 = OpenTVicHW32(HW32, "TVICHW32", "TVicDevice0")

En el següent exemple es pot veure com s'utilitzen els temporitzadors, els convertidors A/D i D/A, i les interrupcions per hardware.

- Configuració Interrupcions:

Primer es necessita saber en quina IRQ del PC esta la nostra targeta configurada (Fig. 2.15)

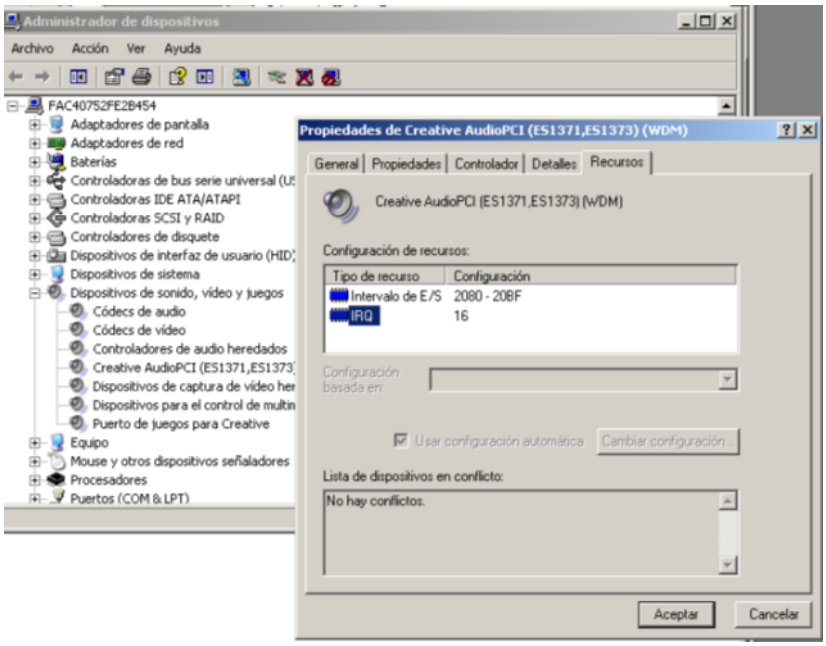


Fig. 2.15 Interrupcions IRQ

Aquesta informació es troba a dins de l'Administrador de dispositius, s'ha de clicar a la tarja que es vol analitzar, dins de propietats -> recursos. En aquest cas la IRQ es 16.

Per tan al activar les interrupcions s'escriu:

```
Call UnmaskIRQ(HW32, 16, AddressOf RutinaServei) 'Activar la IRQ del PC
```

IRQ

Funció d' interrupcions

```
Out &H228, &H0 'Activació de la interrupció de la tarja.
```

Posteriorment es crea un Mòdul nou, amb la funció d'interrupcions (Fig. 2.16)

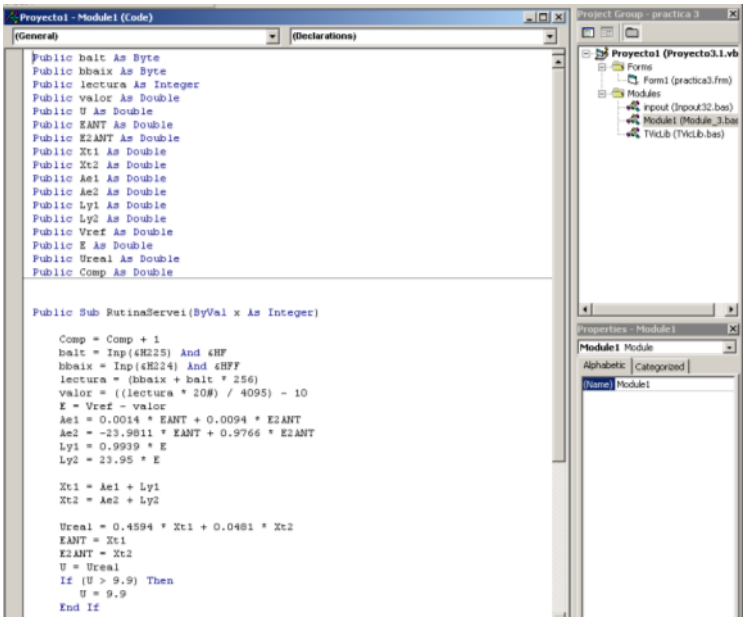


Fig. 2.16 Funció d'interrupcions

L'última instrucció de la funció d'interrupcions te que ser: "Out &H228, &H0" per poder activar de nou les interrupcions ja que cada cop que entra en la funció interrupcions es desactiva.

- **Capturar dades de la entrada analògica**:

Primer de tot s'ha de seleccionar el canal d'entrada que es vol capturar les dades:

```
Out &H22A, &H1
```

On "&H1" es el canal que es vol capturar. En aquest cas esta seleccionat el canal 1, es pot canviar per 2,3,4,5,6,7,8 per utilitzar uns altres canals.

Per altra banda s'ha de seleccionar el guany de l'entrada:

```
Out &H229, &H8 'guany unitari
```

Les instruccions per adquirir les dades de l'entrada analògica són:

```
balt = Inp(&H225) And &HF
bbaix = Inp(&H224) And &HFF
lectura = (bbaix + balt * 256)
valor = ((lectura * 20#) / 4095) - 10
```

La lectura es fa per dues bandes, primer el byte alt i després el byte baix ja que el analògic digital és de 12bits, a traves de les seves direccions de hardware directament.
Posteriorment s'uneixen els dos bytes per tal de tenir una variable a la escala corresponent a vols.

- Escriure un valor en el convertidor Digital / Analògic:

El procediment és el invers:

```
balt = U \ 256
bbaix = U - balt * 256
Call Out(&H224, bbaix)
Call Out(&H225, balt)
```

Separar els dos bytes, la part alta i baixa en dues variables i fer un Out amb l'adreça de hardware corresponent a cada un.

- Utilització dels Temporitzadors "Timer":

La targeta esta dotada de 2 temporitzadors de 16 bits. Que es utilitzat per activa la interrupció de la targeta. En el següent exemple s'està programant el timer de manera que es seleccionà el temps a traves de un camp de text anomenat inter.text. (Fig. 2.17)

```
Private Sub ProgramaTimer()

t1_H = inter.Text \ 256
t1_L = inter.Text - t1_H * 256
t2_H = 7
t2_L = 208


Out &H223, &H74
Out &H221, t1_L
Out &H221, t1_H

Out &H223, &HB4
Out &H222, t2_L
Out &H222, t2_H


End Sub
```

Fig. 2.17 Programar Timer

El primer Timer es amb un temps variable el qual es configura a traves del camp de text "inter.Text" amb les unitats de milisegons. El segon Timer esta amb un valor fixa que dona la unitats de milisegons.

Posteriorment s'escriu a la adreces corresponents els valors dels Timers i s'activen. On el primer timer s'activa per la direcció "&H74" i el segon per "&HB4".

3. Teoria de control per retorn d'estat

3.1. Introducció a les variables d'estat.

Les tècniques de modelatge de sistemes en l' espai d'estat es basen en descriure els sistemes dinàmics a partir de "n" equacions diferencials de primer ordre per al cas de sistemes de temps continu, i "n" equacions en diferència per al cas de sistemes discrets. Aquestes "n" equacions resultants es descriuen per mitjà d' una notació matricial (matriu A,B,C,D), cosa que simplifica la seva representació matemàtica:

La matriu A: hi ha les variables d'estat del sistema.
La matriu B: hi ha les variables d'entrada del sistema.
La matriu C: hi ha les variables de sortida del sistema
La matriu D: és la matriu en la qual l'entrada pot afectar a la sortida (normalment aquesta matriu és 0).
Forma de representació general:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Dx$$

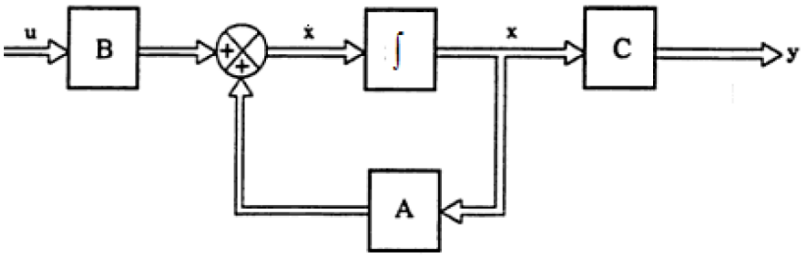


Fig. 3.1 Esquema general d'un sistema per retorn d'estat

La idea d'estat és un concepte bàsic d'aquests sistemes que es podria definir com un instant donat, caracteritzat per un cert conjunt de valors numèrics; i amb aquest conjunt de valors, juntament amb el valor de l'entrada en l'instant actual, es pot determinar el valor que hi haurà a la sortida en l'instant següent.

Arribats aquí, es poden determinar les diferents configuracions per retorn d'estat que hi ha, i es representaran amb el Simulink, juntament amb els diferents observadors (que s'explicaran després) i, finalment, amb un exemple pràctic, amb el qual es farà un control per retorn d'estat, sobre un motor per tal d'obtenir la seva resposta mitjançant un observador.

3.2. Modelització de un sistema.

La modelització d'un sistema, significa descriure un sistema en un conjunt d'equacions diferencials. Si és el sistema continuo o equacions en diferencia si és el sistema discret.

El sistema que s'utilitzarà per modelitzar és el següent:

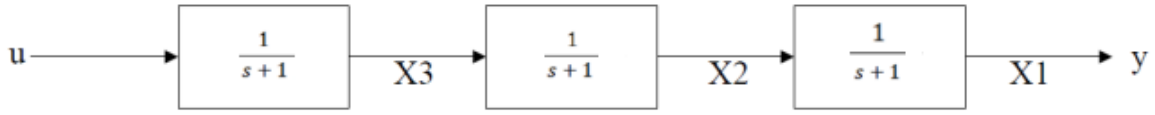


Fig. 3.2 Model LAG+LAG+LAG

Primer s'ha de fer un graf representant el sistema. Es procedeix a fer el graf (Fig. 3.3):

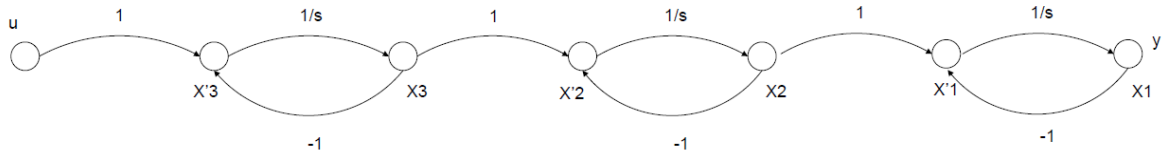


Fig. 3.3 Graf

De la Fig. 4.4 extreuen les següents equacions que formaran les matrius d'estat:

$$\dot{x_1} = -x_1 + x_2 \tag{3.1}$$

$$\dot{x_2} = -x_2 + x_3 \tag{3.2}$$

$$\dot{x_3} = -x_3 + u \tag{3.3}$$

Les matrius d'estat son:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u \tag{3.4}$$

$$y = [1 \quad 0 \quad 0]x + [0]x \tag{3.5}$$

$$A = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{bmatrix} \quad (3.6)$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (3.7)$$

$$C = [1 \quad 0 \quad 0] \quad (3.8)$$

$$D = [0] \quad (3.9)$$

El model d'estat continuo el defineixen les matrius A,B,C,D anteriors. Un cop s'ha obtingut el model continuo, s'extraurà el model discret per exemple per un temps de mostreig de 100ms. Segons la tau del sistema el temps de mostreig es te que fer més gran o mes petit, generalment, s'observa el temps que tarda la resposta en llaç tancat de determinar el temps de mostreig que es treballa, normalment s'escull temps de mostreig 10 vegades més petits que el temps de la resposta. Per fer aquest càlcul s'ha utilitzat la aplicació Matlab amb la funció: c2d(sys,Tm), on "c2d" significa "continus to discret", "sys" és el sistema per convertir i "Tm" és el temps de mostreig.

```
>> a=[-1 1 0;0 -1 1;0 0 -1];
>> b=[0;0;1] ;
>> c=[1 0 0] ;
>> d=[0] ;
>> sysc=ss(a,b,c,d) ;
sysd=c2d(sysc,0.1)
a =          x1        x2        x3
   x1    0.9048   0.09048  0.004524
   x2         0    0.9048   0.09048
   x3         0         0    0.9048

b =           u1
   x1  0.0001547
   x2   0.004679
   x3    0.09516

c =    x1  x2  x3
   y1   1   0   0

d =    u1
   y1   0
Sampling time: 0.1
Discrete-time model.
>> [ad,bd,cd,dd]=ssdata(sysd)
```

Un cop convertit s'obtenen les matrius "ad,bd,cd,dd" que responent el mateix sistema en discret a traves de l' instrucció "ssdata".

Es passa a comprova els resultats amb el Toolbox Simulink del Matlab on es simularà el sistema en llaç tancat. On es veu la resposta a un esglaó unitari.

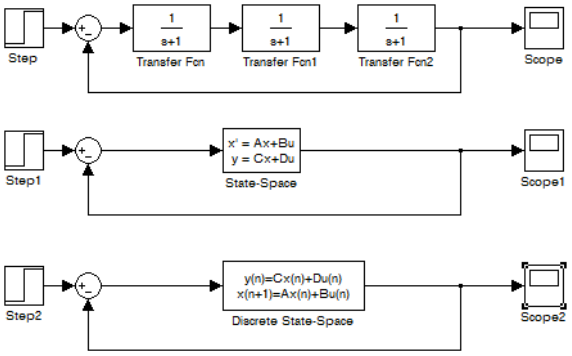
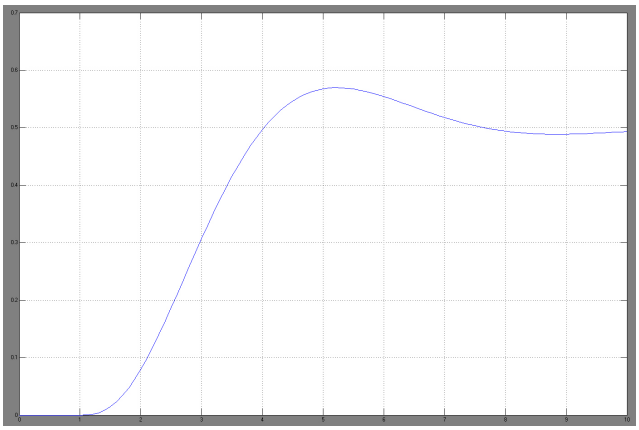


Fig. 3.4 Simulació del sistema en llaç tancat

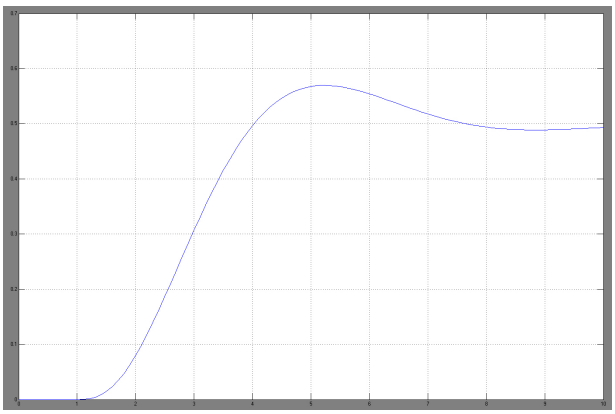
Resposta del sistema en llaç tancat:



Llegenda:

Blau: sortida del sistema

Fig. 3.5 Resposta del sistema original amb llaç tancat

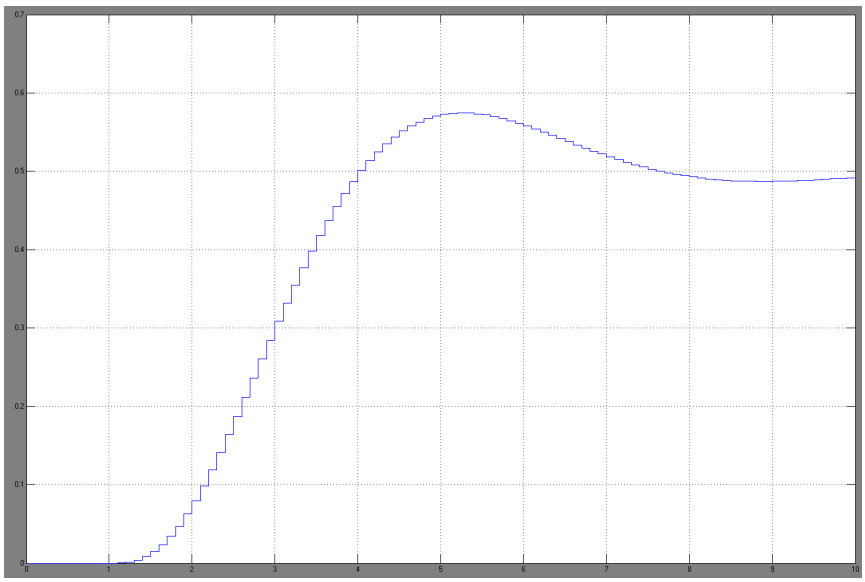


Llegenda:

.Blau: sortida del sistema

Fig. 3.6 Resposta del sistema per retorn d'estat amb llaç tancat

Resposta del sistema per retorn d'estat discret amb llaç tancat (Tm=100ms):



Llegenda:

.Blau: sortida del sistema

Fig. 3.7 Resposta del sistema per retorn d'estat discret amb llaç tancat (Tm=100ms)

Com es pot veure les tres simulacions tenen una resposta similar. En els dos primers casos la resposta és exactament igual, ja que els dos sistemes son continus però expressats de forma diferent. I en el tercer cas té una resposta similar però amb un esglaonat del mostreig de 100ms ja que el sistema és discret. Com es pot veure a la figura el sistema té un error estàtic, ja que per un esglaó unitari s'estabilitza a 0.5. Pertan es tindra que afegir un integrador per augmentar el sistema.

3.3. Disseny de controladors per retorn d'estat.

3.3.1. Càlcul de pols per sistemes continus.

El disseny de un controlador per retorn d'estat consisteix en buscar uns pols que multiplicats per cada una de les variables d'estat X1,X2,X3 es poguí controlar el sistema i reduir el seu temps de resposta.

Per calcular els pols necessaris es possible fer-ho de diferents maneres. Es pot calcular els pols segons les característiques que es volen o a traves d'unes taules estandarditzades segons la resposta que es desitgi com per exemple les taules de ITAE o BESSEL.[1]

Per calcular els pols segons unes característiques desitjades: Mp=10% i Ts=2seg on Mp és el sobre impuls i Ts és el temps d'estabilització. S'utilitzaran les següents formules:

$$\varepsilon = \frac{\left|\ln\left(\frac{Mp}{100}\right)\right|}{\sqrt{\pi^2+\left(\ln\left(\frac{Mp}{100}\right)\right)^2}} \tag{3.10}$$

$$Ts = \frac{4}{-\varepsilon*\omega_n} \tag{3.11}$$

$$\frac{\omega_n}{s^2+2*\varepsilon*\omega_n*s+{\omega_n}^2} \tag{3.12}$$

$$s_{12} = -\varepsilon * \omega_n \pm j * \omega_n * \sqrt{1-\varepsilon^2} \tag{3.13}$$

Per altra banda es pot utilitzar les taules de la Fig. 3.8, on trobem els pols calcificats depenent de el nombre de variables d'estat en el nostre sistema. S'ha de tenir en compte que els valors son per ω_0= 1 rad/s. Si el sistema que es vol controlar és més rapid, és necesari multiplicar els pols, depenent de lo ràpid que es volgui fer el sistema. Els pols de ITAE són respostes amb un sobre impuls (Fig. 3.9) i les respostes de BESSEL són sense sobre impuls (Fig. 3.10).

Polos de respuesta prototipo

(a) Funciones de transferencia ITEA	k	Localización de los polos para $\omega_0 = 1$ rad/s†
	1	$s + 1$
	2	$s + 0.7071 \pm j0.7071$‡
	3	$(s + 0.7081)(s + 0.5210 \pm j1.068)$
	4	$(s + 0.4240 \pm j1.2630)(s + 0.6260 \pm j0.4141)$
	5	$(s + 0.8955)(s + 0.3764 \pm j1.2920)(s + 0.5758 \pm j0.5359)$
	6	$(s + 0.3099 \pm j1.2634)(s + 0.5805 \pm j0.7828)(s + 0.7346 \pm j0.2873)$
(b) Funciones de transferencia de Bessel	k	**Localización de los polos para $\omega_0 = 1$ rad/s†**
	1	$s + 1$
	2	$s + 0.8660 \pm j0.5000$‡
	3	$(s + 0.9420)(s + 0.7455 \pm j0.7112)$
	4	$(s + 0.6573 \pm j0.8302)(s + 0.9047 \pm j0.2711)$
	5	$(s + 0.9264)(s + 0.5906 \pm j0.9072)(s + 0.8516 \pm j0.4427)$
	6	$(s + 0.5385 \pm j0.9617)(s + 0.7998 \pm j0.5622)(s + 0.9093 \pm j0.1856)$

†La localización de los polos para otros valores de ω_0 se puede obtener sustituyendo s/ω_0 por s en todos los casos.
‡Por cuestiones de espacio, los factores $(s + a + jb)(s + a - jb)$ se escribieron como $(s + a \pm jb)$.

Fig. 3.8 Taula pols ITAE i BESSEL

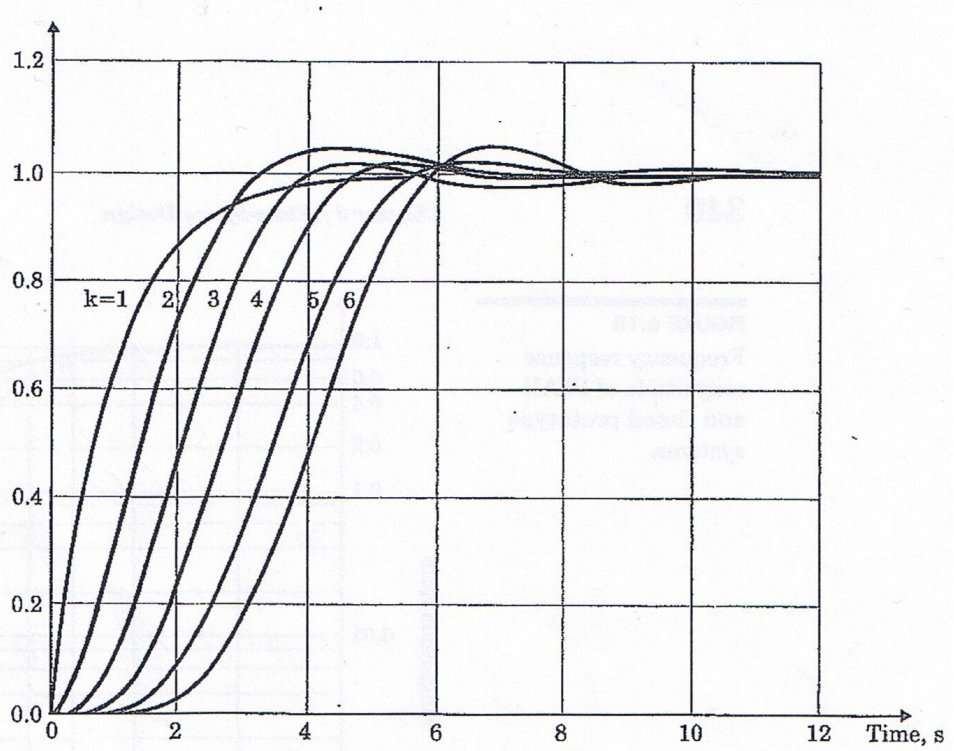


Fig. 3.9 Resposta pols ITAE

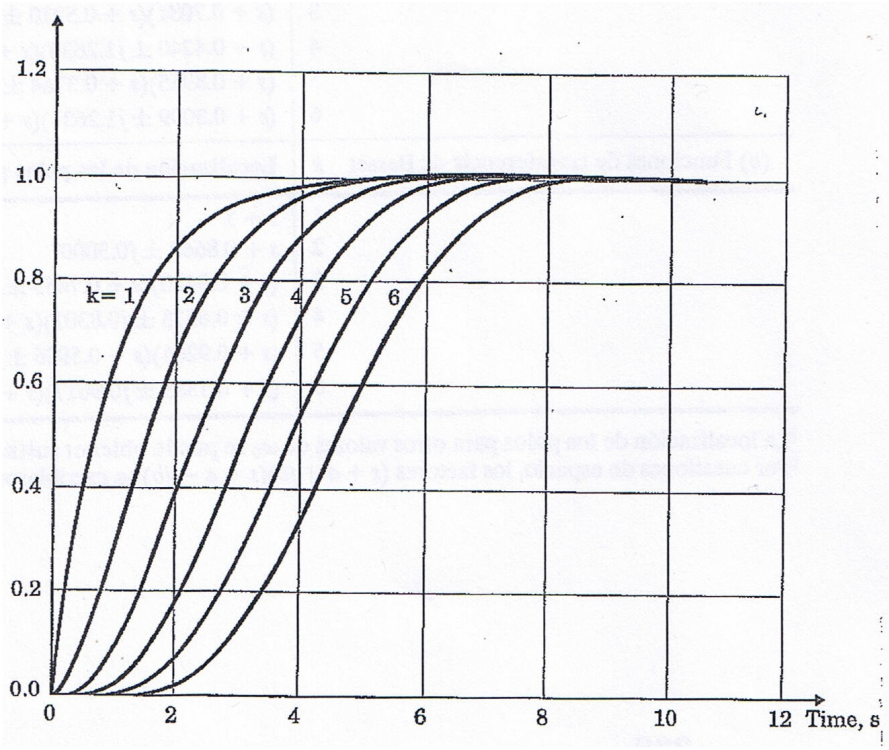


Fig. 3.10 Resposta pols BESSEL

Exemple amb pols:

En aquest cas es calcularan els pols per obtenir una resposta amb un sobre impuls de 10% i un temps de estabilització de 2s. (Mp=10% i Ts=2seg.)

$$\varepsilon = \frac{\left|\ln\left(\frac{Mp}{100}\right)\right|}{\sqrt{\pi^2+\left(\ln\left(\frac{Mp}{100}\right)\right)^2}} = 0.59 \quad (3.11)$$

$$2 = \frac{4}{\varepsilon * \omega_n} \quad (3.12)$$

$$\varepsilon * \omega_n = 2 \quad (3.13)$$

$$\omega_n = 3.383 \quad (3.14)$$

$$s_{12} = -\varepsilon * \omega_n \pm j * \omega_n * \sqrt{1-\varepsilon^2} = -2 \pm j * 2.73 \quad (3.15)$$

S'augmenta el sistema afegint un integrador:

```
>> a1a=[0,c;zeros(3,1),a]
a1a =0     1     0     0
     0    -1     1     0
     0     0    -1     1
     0     0     0    -1
>> b1a=[0;b]
b1a =0
     0
     0
     1
>> c1a=[0,c]
c1a = 0     1     0     0
>> d1a=0
```

S'introdueix els pols:

```
>> Pc=[-2+j*2.73,-2-j*2.73,-20,-20]
```

Un cop calculats el pols, es necessari calcula la matriu K. La matriu K es calcula a traves de la formula Ackerman. Desde el Matlab s'executa la instrucció:

```
Ka=acker(a1a,b1a,Pc)

Ka =  1.0e+003 *

    4.5812    1.5297    0.4865    0.0410
```

Un cop obtinguts els resultats és simula el sistema amb llaç tancat com el controlador.(Fig. 3.11)

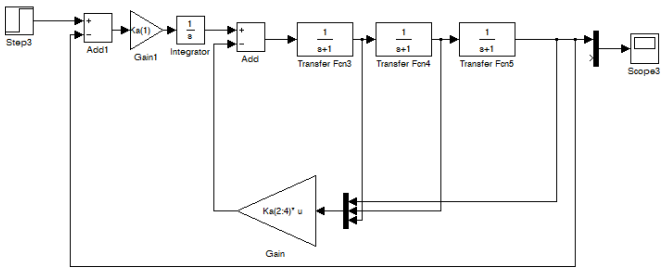
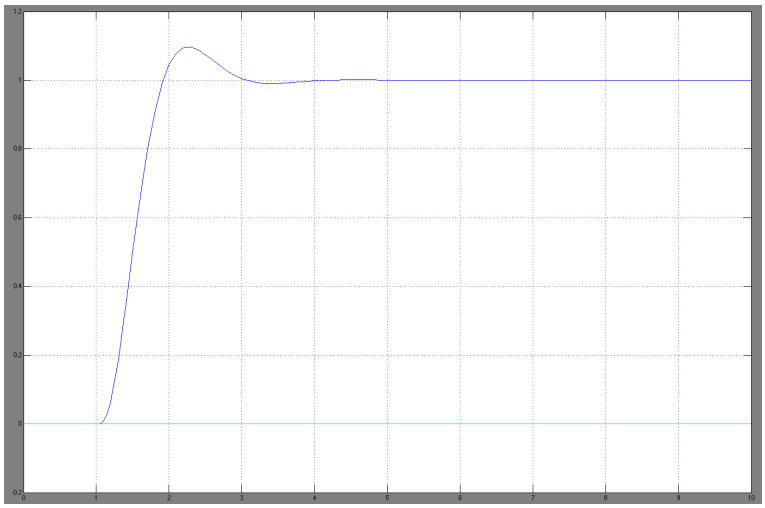


Fig. 3.11 Diagrama control per retorn d'estat



Llegenda:

Blau: sortida del sistema

Fig. 3.12 Resposta del control per retorn d'estat

3.3.2. Càlcul de pols per sistemes discrets.

El càlcul de pols discrets és igual que els pols continus però ara s'han de discretitzat. En primer lloc la formula per discretitzar és:

$$s_{12_{discrets}} = e^{Tm*s_{12}} \tag{3.16}$$

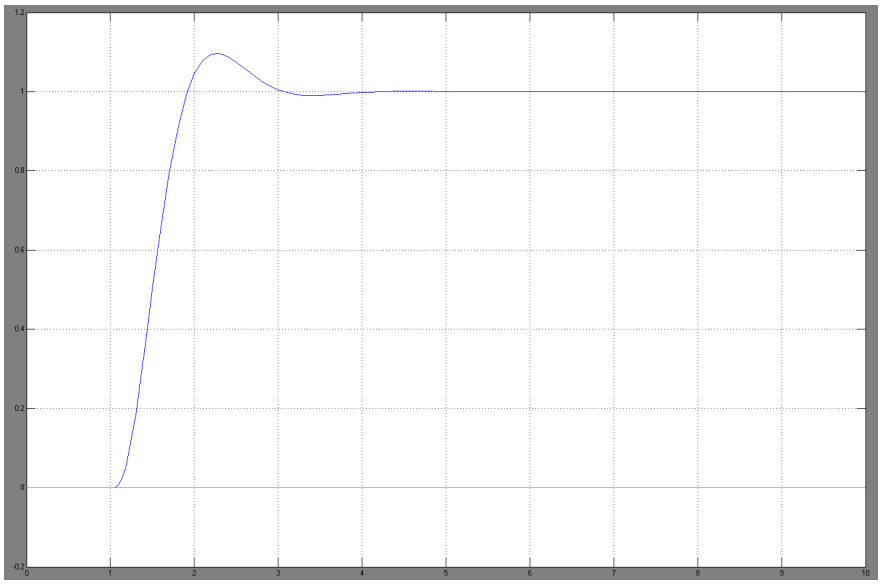
En el exemple anterior calculem els pols a traves del Matlab (Tm=0.05=50ms):

```
>> Pd=[exp(Pc(1,1)*0.05) exp(Pc(1,2)*0.05) exp(Pc(1,3)*0.05)]
Pd =   0.8964 + 0.1231i   0.8964 - 0.1231i   0.3679
```

Ara es crea la matriu K com s'ha fet anteriorment:

```
Kc=acker(ad,bd,Pd)
Kc =   96.0458   30.6158   13.4102
```

Es simula la resposta del controlador:



Llegenda:

Blau: sortida del sistema

Fig. 3.13 Resposta del control per retorn d'estat discret

3.4. Disseny d'observadors.

3.4.1. Introducció als observadors i tipus.

Un observador consisteix en fer una aproximació del valors que tenen les variables d'estat en cada moment, per poder calcular les sortida. Aquesta manera de calcular les variables d'estat és molt comú quant no es pot accedir a les variables d'estat directament i nomes es té accés a la sortida i a l'entrada. La problemàtica que presenten aquests tipus de control es que s'ha de augmentar d'ordre el sistema per dos, és a dir per un sistema d'ordre tres el sistema controlat amb llaç tancat es de ordre sis. Això fa que el control no sigui tan precís com amb un controlador per retorn d'estat sense observador.

El càlcul d'un observador consisteix en afegir uns nous pols de manera que aquests pols no afectin el control del sistema per això es posant molt allunyats dels pols de control. Normalment s'intenta que els pols de l'observador siguin 10 vegades mes petit que els pols del controlador. (Fig. 3.14)

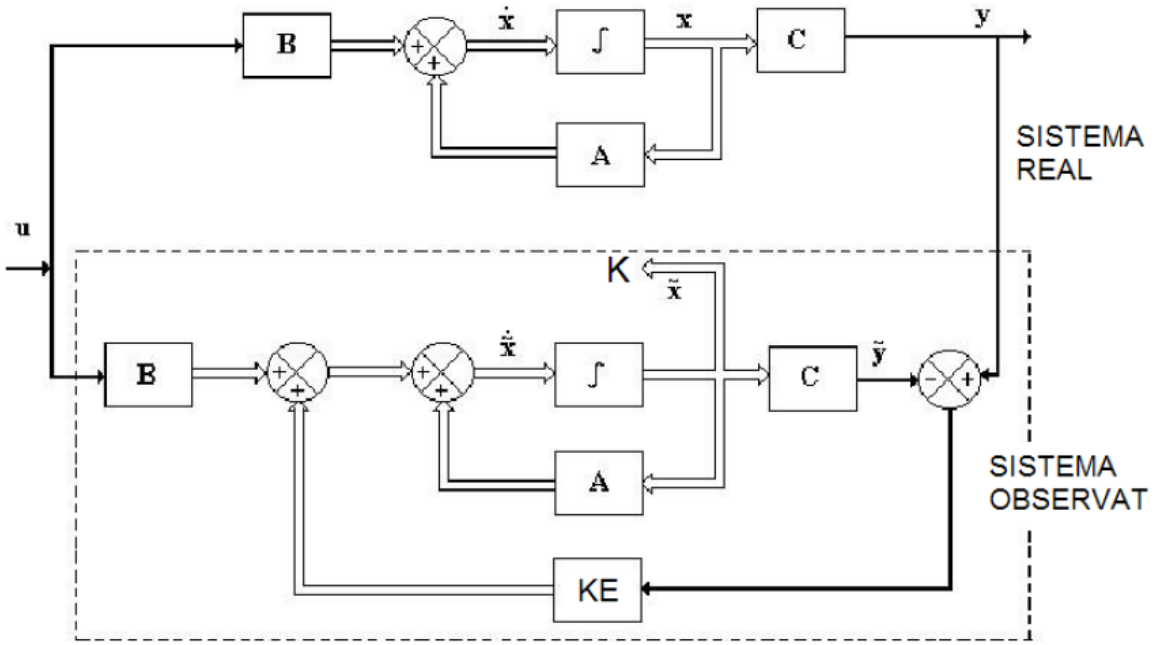


Fig. 3.14 Estructura d'un observador complet

El control amb observador es pot fer de forma continua o de forma discreta exactament igual com s'ha vist en el control per retorn d'estat.

De la forma continua es tracte de controlar a traves a una senyal analògica generada a traves de operacionals. I de forma discreta es pot implementar a traves de un ordinador i convertidors digitals analògics, pero la implementació pot ser de diferents maneres segons el hardware que disposem i el sistema que es volgui controlar. Els observadors discrets els podem implementa de forma predictiu o corrent. Es pot trobar tota la informació en la referencia [1].

Observador Predictiu:

L'observador predictiu, consisteix en calcular la estimació amb l'entrada anterior. Aquest tipus de implementació es la manera estàndard i va bé per ordinadors lents o amb temps de mostreig molt ràpid. Es calcula:

$$\hat{x}(k+1) = A * \hat{x}(k) + B * u(k) + L * \big(y(k) - C * \hat{x}(k)\big) \tag{3.17}$$

$$u(k+1) = -K * \hat{x}(k+1) \tag{3.18}$$

Ho sigui, es corregeix l'estimació amb l'entrada a l'instant k per calcular la sortida a l'instant k+1.

La forma d'implementar-ho seria:

Instant k

Surt $u(k+1)$ calculat en l'anterior passada

Entra $y(k)$

Es calcula i després $u(k+1)$

Instant k+1

Surt $u(k+1)$ calculat en l'anterior passada

Entra $y(k+1)$

Es calcula i després $u(k+2)$

Així indefinidament per k+n mostres que obtingui el convertidor analògic digital. A la pràctica la implementació seria:

```
E = Vref – valor ' Realimentació/ sortida del sistema menys la consigna.
Ae1 = A(1,1) * X1ANT + A(1,2) * X2ANT  ' Multiplicació de la matriu A per
Ae2 = A(2,1) * X1ANT + A(2,1) * X2ANT' les variables d'estat de una
mostra anterior
Ly1 = L(1,1) * E ' Multiplicació dels observador per la entrada del
controlador
Ly2 = L(2,1) * E ' Multiplicació dels observador per la entrada del
controlador
Xt1 = Ae1 + Ly1 ' Aproximació de X1
Xt2 = Ae2 + Ly2 ' Aproximació de X2
Ureal = K(1,1) * Xt1 + K(1,2) * Xt2 ' Calcul del controlador
X1ANT = Xt1 ' Guarda la variable d'estat X1
X2ANT = Xt2  ' Guarda la variable d'estat X2
U = Ureal ' Canvi de variable el valor.
```

Com es pot veure en el codi anterior s'implementa un control amb observador predictiu de un sistema de ordre 2, s'utilitza una sèrie de variables calculades anteriorment com la matriu A, L, K. On La matriu K es el mateix que la matriu C del controlador o la matriu L es el mateix que la matriu B del controlador. Com que en els llenguatges de programació no es pot fer operacions amb matrius s'ha de fer a base de multiplicacions i sumes de un sol element. En les línees 2 i 3 es multiplica les matriu A per la matriu de les variables d'estat aproximades en el instant anterior. Posteriorment es multiplica la matriu L per la entrada del controlador i es calcula les valors de les variables d'estat aproximades sumant els dos matrius obtingudes. Un cop ja s'ha calculat les variables d'estat atraves del

observador, es calcular la senyal de sortida exactament igual, com si fos un control per retorn d'estat.

Observador Corrent:

L'Observador corrent, funciona de manera que es calcula en el instant k el valor que surt. Sempre es fa tot en el mateix instant. Aquest procediment és el més lògic de funcionament però si el ordinador on s'està implementant el temps de càlcul és llarg, la sortida que surt té un retard. Es calcula:

$$\hat{x}_p(k) = A * \hat{x}(k-1) + B * u(k-1) \tag{3.19}$$

$$\hat{x}(k) = \hat{x}_p(k) + L_c * \big(y(k) - C * \hat{x}(k)\big) \tag{3.20}$$

$$u(k) = -K * \hat{x}(k) \tag{3.21}$$

La forma d'implementar-ho seria:

Instant k

Entra y(k) i es calcula amb el valor calculat en la mostra anterior i es calcula u(k) que surt immediatament

Càlcul per la mostra següent

Instant k+1

El mateix

En aquest cas l'entrada, la correcció i el càlcul de la sortida és fa en el mateix període de mostreig. La relació entre les dues matrius L és la següent:

$$L_c = A^{-1} * L \tag{3.22}$$

On L és la matriu de l'estimador predictiu, la que obtenim amb el Matlab.

Es adequat per ordinadors ràpids o amb temps de mostreig lent, així no s'acumula el retard entre l'entrada y(k) i la sortida corresponent u(k)

Totes les matrius A,B,i C són les del model del sistema que es vol controlar i la matriu K és la de retorn d'estat.

A la pràctica la implementació seria:

```
E = Vref – valor ' Realimentació/ sortida del sistema menys la consigna.
X1 = Xp1 + L(1,1) * (E - 1 * Xp1) ' Càlcul de X1 a traves del Observador
i X1 anterior
X2 = Xp2 + L(2,1) * (E - 1 * Xp1) ' Càlcul de X2 a traves del Observador
i X2 anterior
Ureal = K(1,1) * X1 + K(2,1) * X2 ' Càlcul de la sortida
Ae1 = A(1,1) * X1 + A(1,2) * X2 ' Matriu A per les variables d'estat
Ae2 = A(2,1) * X1 + A(2,1) * X2 ' Matriu A per les variables d'estat
Bu1 = L(1,1) * Ureal ' Càlcul de "B" o "L" per la sortida
Bu2 = L(2,1) * Ureal ' Càlcul de "B" o "L" per la sortida
Xp1 = Ae1 + Bu1 ' Càlcul X1 en la mostra anterior
Xp2 = Ae2 + Bu2 ' Càlcul X2 en la mostra anterior
U = Ureal ' Canvi de variable el valor.
```

Com es pot veure en el codi anterior s'implementa un control amb observador corrent de un sistema de ordre 2, s'utilitza una sèrie de variables calculades anteriorment com la matriu A, L, K. On La matriu K es el mateix que la matriu C del controlador o la matriu L es el mateix que la matriu B del controlador. Com que en els llenguatges de programació no es pot fer operacions amb matrius s'ha de fer a base de multiplicacions i sumes de un sol element. En les línees 2 i 3 es calcula les variables d'estat a traves de les variables d'estat anteriors i els observadors.

3.4.2. Càlcul d' observadors continus.

El càlcul d'un observador consisteix en col·locar un pols més allunyats dels pols del controlador. Per exemple:

S'utilitza uns pols de bessel de ordre n=3 per calcular el observador:

```
bessel3=[-0.9420 -0.7455+j*0.7112,-0.7455-j*0.7112]
```

Càlcul de pols de l'observador:

```
Poc=bessel3*6
```

Càlcul de la Matriu L a traves de Ackerman:

```
L=acker(a',c',Po)'
L =11.5980
   62.5835
  140.8194
```

S'utilitza uns pols de bessel de ordre n=4 per calcular el control:

```
bessel4=[0.6573+j*0.8302,-0.6573-j*0.8302,-0.9047+j*0.2711,-0.9047-
j*0.2711]
```

Càlcul de la Matriu K a traves de Ackerman:

```
Kao=acker(a1a,b1a,bessel4)
Kao =    0.2294    0.1256    0.5304   -1.1906
K=Kao(2:4);
Kint=Kao(1);
```

Creació del controlador:

```
aconc=a-b*K-L*c
bconc=L
cconc=K
dconc=0
```

Simulació del control continuo:

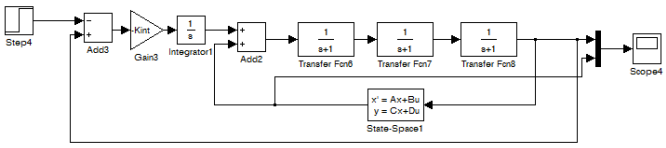
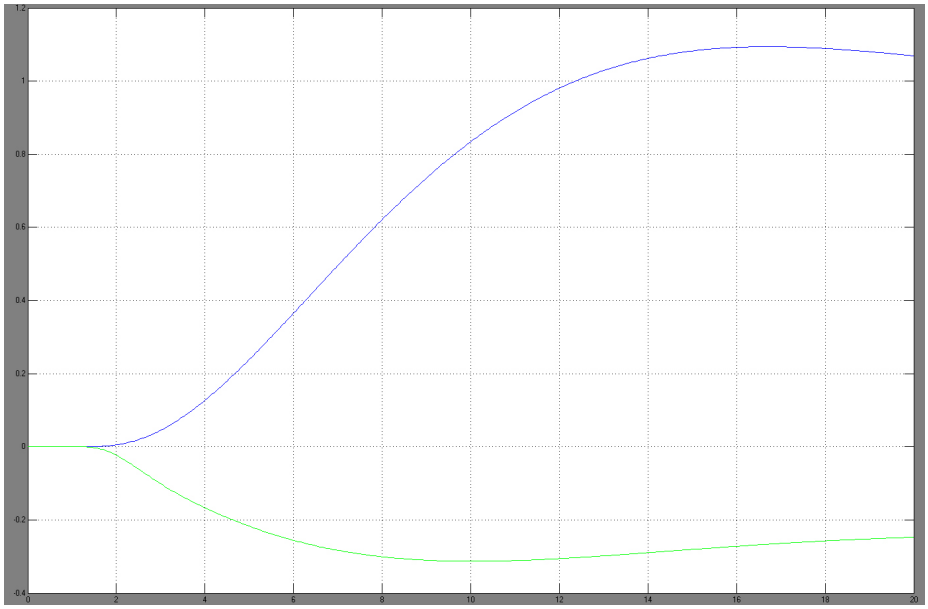


Fig. 3.15 Diagrama del sistema controlat per retorn d'estat amb observador continuo



Llegenda:

Blau: sortida del sistema

Verd: Acció de control

Fig. 3.16 Resposta del sistema controlat per retorn d'estat amb observador continuo

3.4.3. Càlcul d'observadors discret.

El càlcul d'un observador consisteix en col·locar un pols més allunyats dels pols del controlador. Per exemple:

S'utilitza uns pols de bessel de ordre n=3 per calcular el observador:

```
bessel3=[-0.9420 -0.7455+j*0.7112,-0.7455-j*0.7112]
```

Discretitzem els pols per un temps de mostreig de 50ms observador:

```
Pdo=[exp(-0.9420*0.05)     exp((-0.7455+j*0.7112)*0.05)     exp((-0.7455-
j*0.7112)*0.05)]
```

Càlcul de pols de l'observador:

```
Pdoo=Pdo/6
```

Discretitzem el sistema:

```
sysd=c2d(sysc,0.05)
[ad,bd,cd,dd]=ssdata(sysd)
```

Càlcul de la Matriu L a traves de Ackerman:

```
Ld=acker(ad',cd',Pdoo)'
Ld =2.3738
   34.0163
  219.0059
```

S'augmenta el sistema amb un integrador:

```
a1ad=[1,cd;zeros(3,1),ad]
b1ad=[0;bd]
c1ad=[0 cd]
d1ad=0
```

S'utilitza uns pols de bessel de ordre n=4 per calcular el control:

```
bessel4=[0.6573+j*0.8302,-0.6573-j*0.8302,-0.9047+j*0.2711,-0.9047-
j*0.2711]
```

Es discretitzen el pols per un temps de 50ms:

```
Pcd=[exp((0.6573+j*0.8302)*0.05),exp((0.6573-j*0.8302)*0.05),exp((-
0.9047+j*0.2711)*0.05),exp((-0.9047-j*0.2711)*0.05)]
Pcd =1.0325 + 0.0429i    1.0325 - 0.0429i    0.9557 + 0.0130i    0.9557 -
0.0130i
```

Càlcul de la Matriu K a traves de Ackerman:

```
Kad=acker(a1ad,b1ad,Pcd)
Kad =    0.0532    0.7967    1.7268   -2.5595
Kd=Kad(2:4);
Kintd=Kad(1);
```

Creació del controlador:

```
aed=ad-bd*Kd-Ld*cd
bed=Ld
ced=-Kd
ded=0
```

Simulació del control discret:

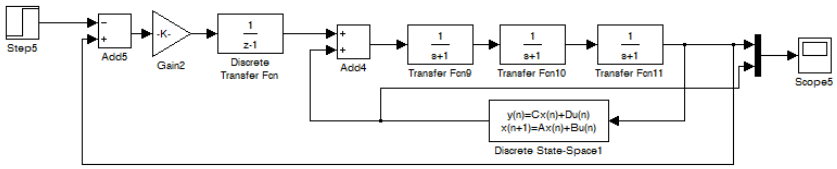
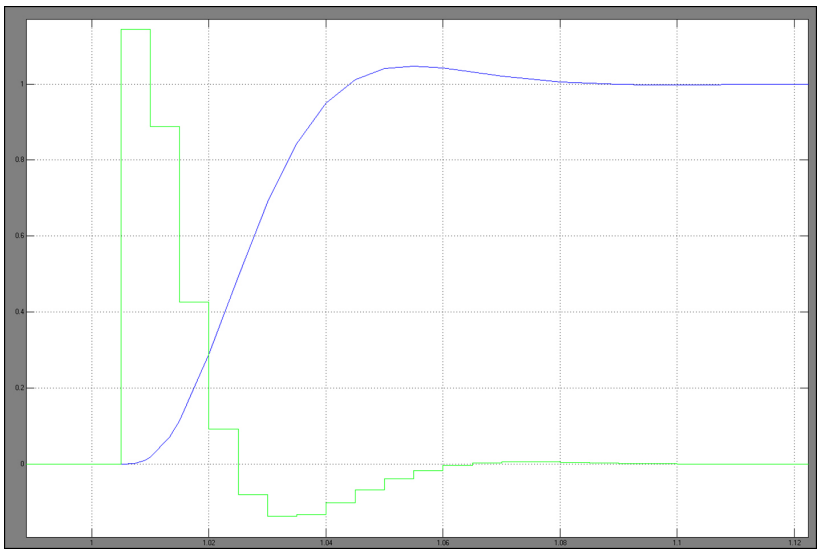


Fig. 3.17 Diagrama del sistema controlat per retorn d'estat amb observador discret



Llegenda:

Blau: sortida del sistema

Verd: Acció de control

Fig. 3.18 Resposta del sistema controlat per retorn d'estat amb observador discret

3.5 Disseny de observadors reduïts.

3.5.1. Introducció a observadors reduïts.

Mostrades ja les diferents tipologies a les quals es pot arribar amb un observador d'estat complert, i les diferents maneres que hi ha per poder-los construir, seguidament es veurà un altre tipus d'observador. L'observador d'ordre reduït.

L'observador d'ordre reduït, consisteix en fer un observador d'ordre n-1. Per exemple un sistema d'ordre 3 com el exemple utilitzat en tot aquest capítol, l'observador seria d'ordre 2, fent que el conjunt del controlador més planta sigui d'ordre 5 en comptes de ordre 6.

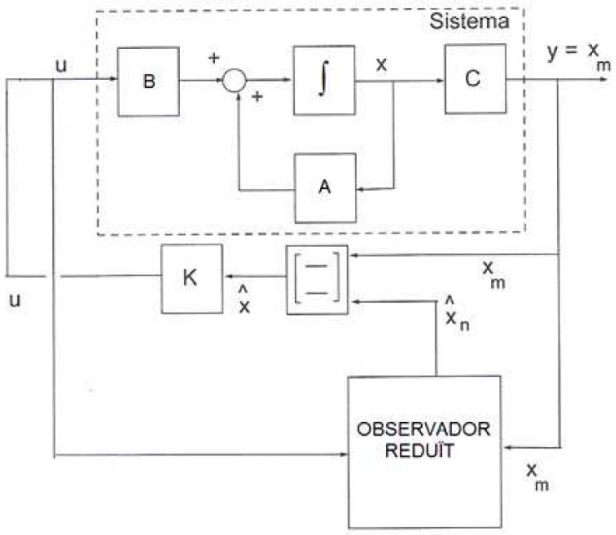


Fig. 3.19 Estructura d'un observador d'odre reduït

El vector X pot ser dividit en 2 vectors:

X_a correspon als estats mesurables, d'ordre (m x 1)

X_b correspon als estats observables, d'ordre (n-m x 1)

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_m \\ X_{m+1} \\ X_{m+2} \\ \vdots \\ X_n \end{bmatrix}_{nx1}$$

$X_1 \ldots X_m \Rightarrow$ Estats coneguts o medibles

$X_{m+1} \ldots X_n \Rightarrow$ Estats no coneguts que requereixen ser observats

Sabent això, s'obté la següent matriu:

$$\begin{bmatrix} \dot{X}_a \\ \dot{X}_b \end{bmatrix} = \begin{bmatrix} Aaa & Aab \\ Aba & Abb \end{bmatrix} \begin{bmatrix} Xa \\ Xb \end{bmatrix} + \begin{bmatrix} Ba \\ Bb \end{bmatrix} u \quad (3.23)$$

$$Y = [Ca \quad Cb] \begin{bmatrix} Xa \\ Xb \end{bmatrix} \quad (3.24)$$

Les dimensions són:

Aaa → m x m
Aab → m x n-m
Aba → n-m x m

Abb → n-m x n-m
Ba → m x 1
Bb → n-m x 1
Ca → 1 x m
Cb → 1 x n-m

Per tant, el sistema queda reduït a la següent expressió:

$$\dot{X}_a = Aaa * Xa + Aab * Xb + Ba * u \quad (3.25)$$
$$\dot{X}_b = Aba * Xa + Abb * Xb + Bb * u \quad (3.26)$$

Així doncs, aquest sistema el que fa és dividir-nos el vector X en la part mesurable del sistema (valors que es coneixen), i els valors que s'han d'observar.

El sistema que descriuen els observadors d'ordre reduït són el següents:

Equació d'estat: $$\dot{X}_b = Aba * Xa + Abb * Xb + Bb * u \quad (3.27)$$

Equació de sortida: $$Aab * Xb = \dot{X}_a - Aaa * Xa + Ba * u \quad (3.28)$$

Ara que se sap això, es poden establir unes equivalències amb l'observador d'estat complert per tal de facilitar el càlcul:

Observador d'ordre complert	Observador d'ordre reduït
$\overline{X}$	$\overline{X}b$
A	Aab
$B \cdot u$	$Aba \cdot Xa + Bb \cdot u$
Y	$\dot{X}a - Aaa \cdot Xa - Ba \cdot u$
$\overline{Y}$	$Aab \cdot \overline{X}b$
C	Aab

Taula 4.1 Conversió d'observador complet a reduït

El vector Kr de l'observador reduït que es busca té un ordre de n-m x 1.

3.5.2. Càlcul de observadors reduïts continus.

En primer lloc es crea amb el Matlab les matrius equivalents a partir del sistema anterior.

```
>> Aaac=a(1,1);
>> Aabc=a(1,2:3);
>> Abac=a(2:3,1);
>> Abbc=a(2:3,2:3);
>> Bac=b(1);
>> Bbc=b(2:3);
```

```
>> Kac=Kc(1);
>> Kbc=Kc(2:3);
>> podrc=Poc(1:2);
```

Un cop creades les matrius equivalents es calcula la matriu de l'observador reduït de la mateix manera que s'ha fet amb l'observador complet però amb les matrius creades:

```
>> Lrc=acker(Abbc',Aabc',podrc)'
Lrc =   -1.8313
         0.8383
```

Es crea el controlador:

```
>> Arc=Abbc-Lrc*Aabc+(Bbc-Lrc*Bac)*(-Kbc);
>> Brc=Arc*Lrc+Abac-Lrc*Aaac+(Bbc-Lrc*Bac)*(-Kac);
>> Crc=-Kbc;
>> Drc=-Kac-Kbc*Lrc;
```

En aquest moment es pot simular el sistema en llaç tancat a traves del simulink per poder comprovar la resposta.(Fig. 4.21)

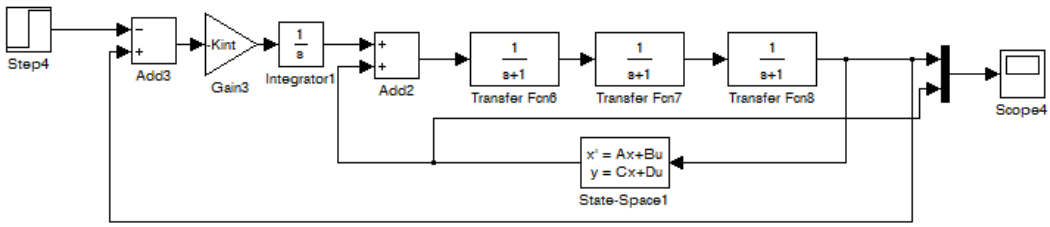
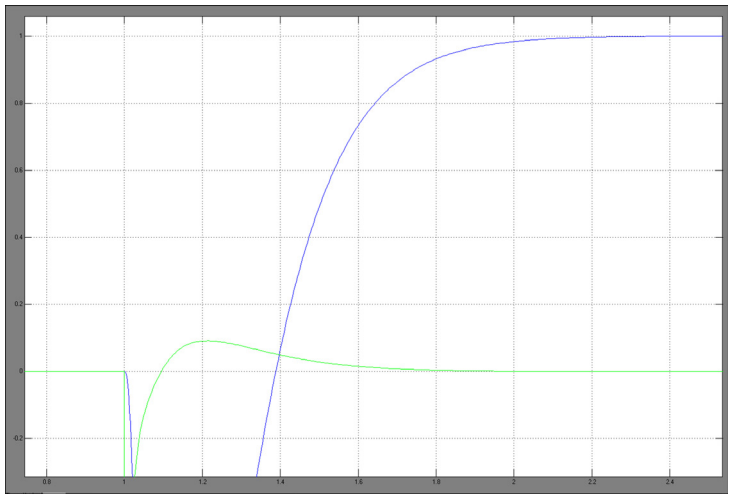


Fig. 3.20 Simulink de l'observador reduït continuo



Llegenda:

Blau: sortida del sistema

Verd: Acció de control

Fig. 3.21 Resposta del sistema amb observador reduït continuo

3.5.3. Càlcul de l'observador reduït discret.

En primer lloc es crea amb el Matlab les matrius equivalents a partir del sistema anterior.

```
>> Aaa=ad(1,1);
>> Aab=ad(1,2:3);
>> Aba=ad(2:3,1);
```

```
>> Abb=ad(2:3,2:3);
>> Ba=bd(1);
>> Bb=bd(2:3);
>> Ka=k(1);
>> Kb=k(2:3);
>> podr=Pod(1:2);
```

Un cop creades les matrius equivalents es calcula la matriu de l'observador reduït de la mateix manera que s'ha fet amb l'observador complet però amb les matrius creades:

```
>> Lr=acker(Abb',Aab',podr)'
```

Es crea el controlador:

```
>> Ar=Abb-Lr*Aab+(Bb-Lr*Ba)*(-Kb);
>> Br=Ar*Lr+Aba-Lr*Aaa+(Bb-Lr*Ba)*(-Ka);
>> Cr=-Kb;
>> Dr=-Ka-Kb*Lr;
```

En aquest moment es pot simular el sistema en llaç tancat a traves del simulink per poder comprovar la resposta.(Fig. 3.22)

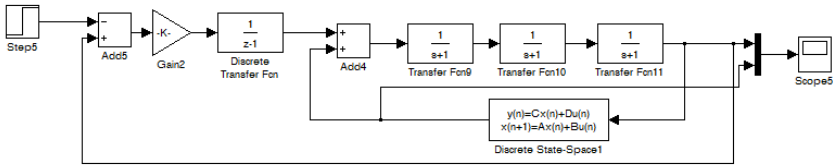
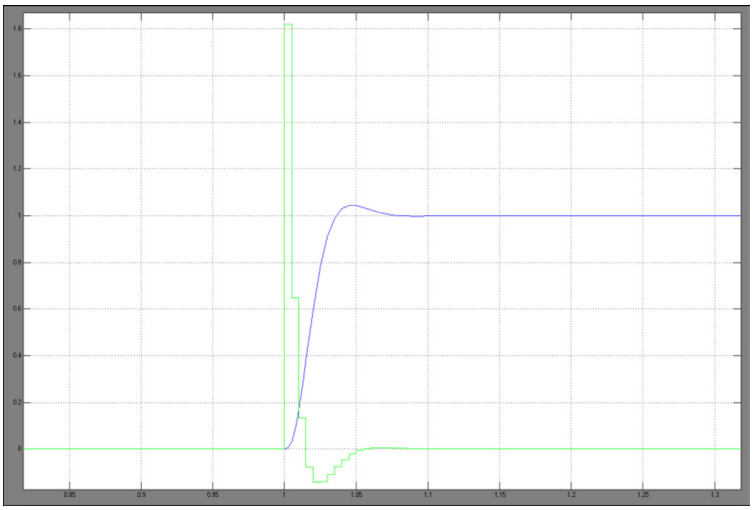


Fig. 3.22 Simulink de l'observador reduït discret



Llegenda:

Blau: sortida del sistema

Verd: Acció de control

Fig. 3.23 Resposta del sistema amb observador reduït discret

4. Control de posició d'un motor.

4.1. Introducció del sistema.

El sistema esta compost d'un motor de continua connectat a una sèrie de reduccions i a un tacòmetre que mesura la posició del motor. El tacòmetre dona 0v a 0º i ±10v a 180º. Tot aquest sistema s'ha connectat a una placa electrònica Feedback i a una font d'alimentació de ±15v i +5v. En la Fig. 5.1 es pot veure la composició del equip.

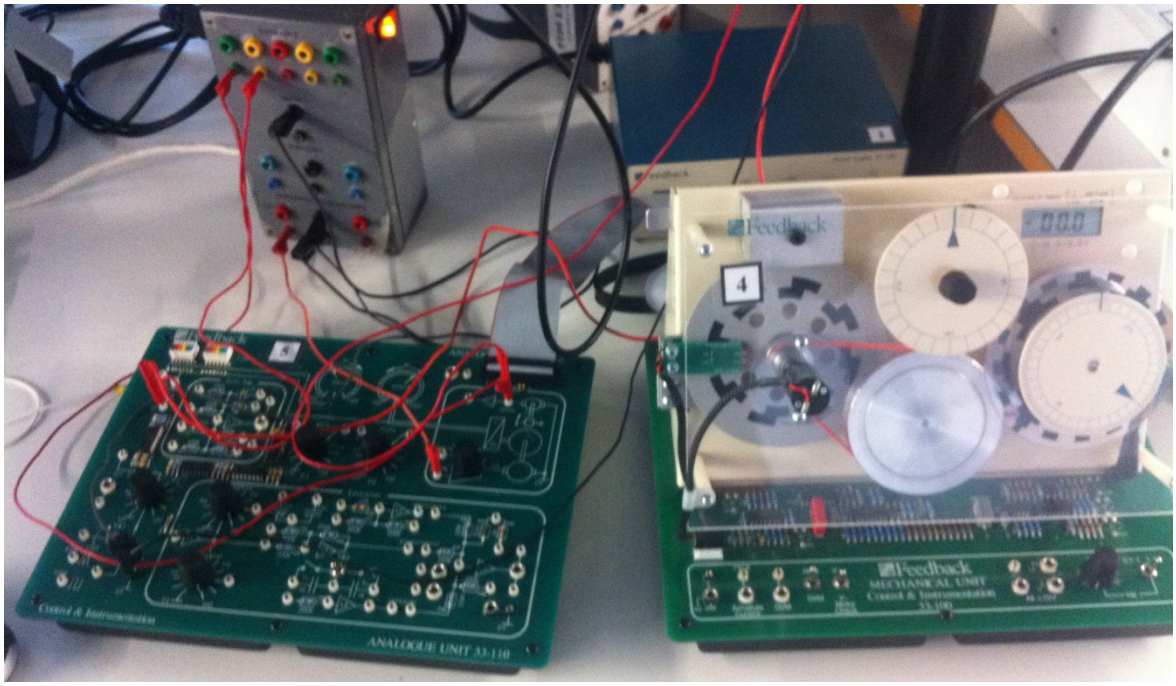


Fig. 4.1 Sistema "Feedback Mechane al Unit"

Aquest sistema permet capturar les dues variables d'estat: Velocitat i posició. Per tant es pot veure el comportament de tots els tipus de controladors.

Després de fer un estudi de les dues targetes de adquisició de dades disponibles en els laboratoris del Tecno Campus Mataró , s'ha decidit utilitzar la targeta PC-LAB per tal de poder interactuar amb el hardware directament a traves del Visual Basic 5.0. Ja que la targeta NI PCI-6014, per treballar amb Visual Basic 6.0 o Visual C++, els drivers són de pagament.

El model matemàtic del sistema esta compost de un LAG i un integrador.

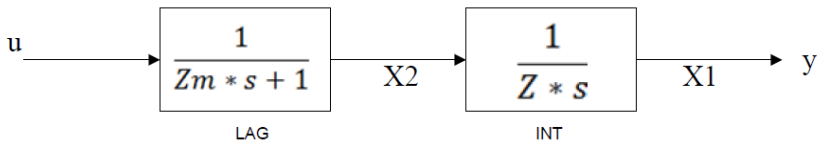


Fig. 4.2 Model Motor LAG+INT

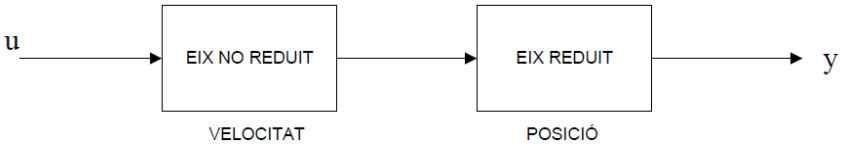


Fig. 4.3 Model motor descriptiu

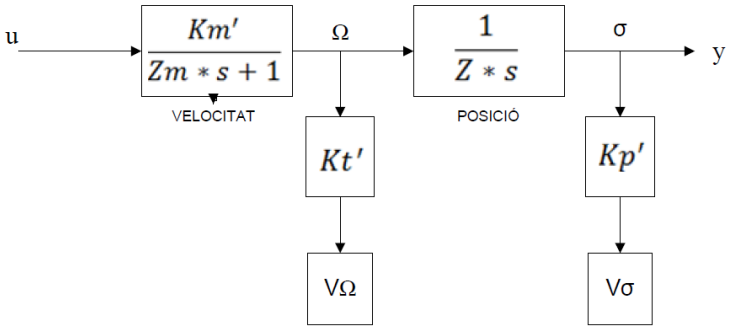


Fig. 4.4 Model de motor amb conversió a Vols les variables d'estat

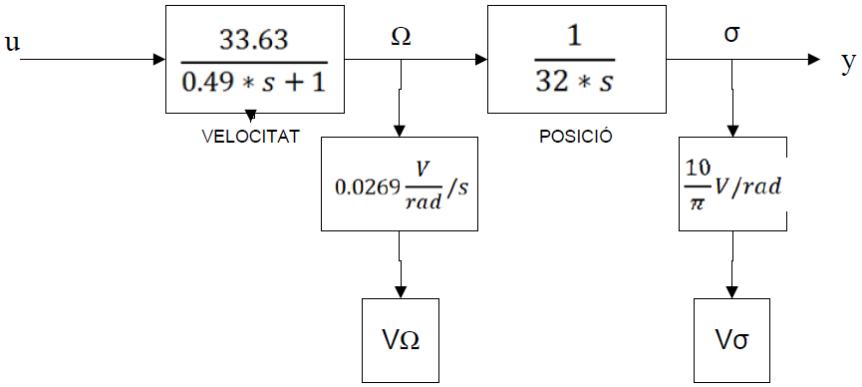


Fig. 4.5 Model motor real

Per més informació del sistema es pot consultar a la direcció: http://www.feedback-group.com/product/servo-fundamentals-trainer-6048

4.2. Càlcul del model.

Partint del model anterior de la Fig. 4.5, es pot realitzar un graf per poder convertir aquest model en les matrius de variable d'estat. Primer s'ha d'arreglar la funció de transferència per poder realitzar el graf:

$$K \longrightarrow \frac{\frac{Km' * Kp'}{32 * Zm} * s^{-1}}{1 + \frac{1}{Zm} * s^{-1}} \tag{5.1}$$

$$\frac{K * s^{-1}}{1 + \frac{1}{Zm} * s^{-1}} \tag{5.2}$$

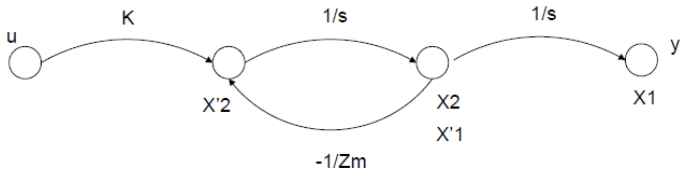


Fig. 4.6 Graf Motor

Càlcul de les matrius d'estat:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{Zm} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -2.0408 \end{bmatrix} \tag{5.3}$$

$$B = \begin{bmatrix} 1 \\ \frac{Km'}{\frac{Kp'}{32 * Z}} \end{bmatrix} = \begin{bmatrix} 1 \\ 6.8270 \end{bmatrix} \tag{5.4}$$

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \tag{5.5}$$

$$D = [0] \tag{5.6}$$

4.3. Simulació del model en llaç tancat.

Un cop obtingudes les matrius d'estat es simula el sistema amb el Matlab i el simulink. En continuo i discret per poder calcular els controladors amb discrets, ja que es controlarà el sistema des de la targeta PC-LAB i la senyal que es pot generar es discreta. En primer lloc s'ha d'analitzar el sistema per poder saber quina tau té el sistema i per tan poder determinar a quina freqüència de mostreig em de treballar. Com s'ha vist a la creació del model, la

constant de temps "TAU" esta per el ordre de 0.49 seg. Per tan s'ha de discretitzar el sistema un valor 10 vegades més petit, per exemple 0.049 seg=50ms o encara més petit. Per aquest sistema s'ha decidit utilitzar un temps de mostreig de 10 ms ja que es més petit que 50ms i es tindrà, més control.

S'introdueix les matrius A,B,C,D en el Matlab:

```
>> a=[0 1;0 (-1/0.49)]
>> b=[1; 6.8270]
>> c=[1 0]
>> d=0
>> sysc=ss(a,b,c,d);
```

Es discretitzà el sistema per un temps de mostreix de 10ms ja que la tau del sistema és de 0.5s.

```
>> sysd=c2d(sysc,0.01);
>> [ad,bd,cd,dd]=ssdata(sysd)
ad =    1.0000    0.0099
             0    0.9798
bd =    0.0103
        0.0676
cd =     1     0
dd =     0
```

Es simula el sistema amb llaç tancat, per comprovar si la conversió ha variable d'estat és correcta:

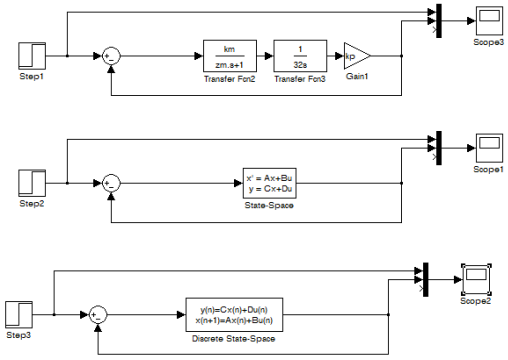
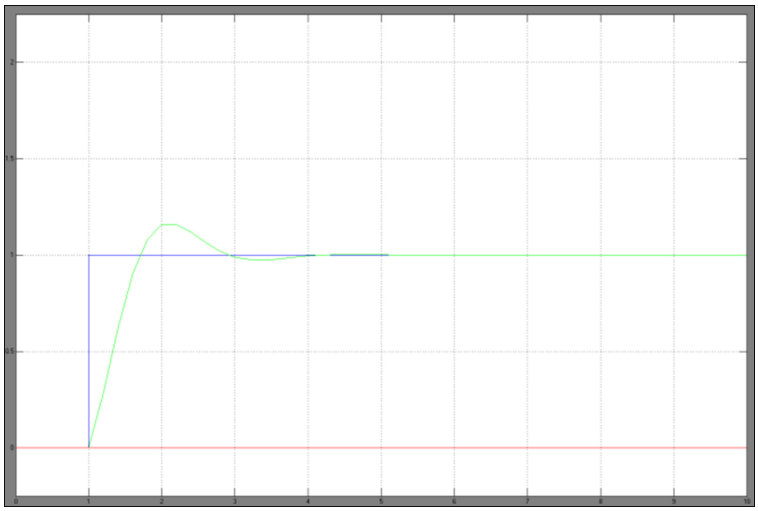


Fig. 4.7 Simulink Motor amb llaç tancat

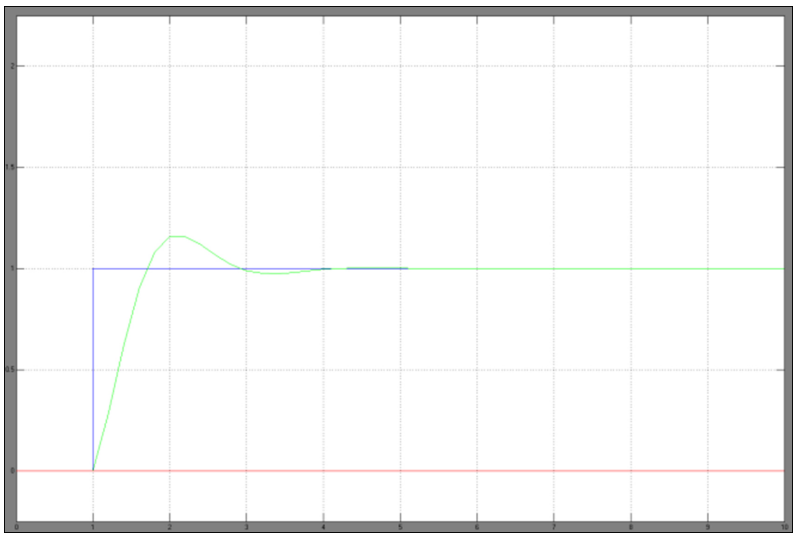


Llegenda:

Blau: Entrada "STEP"

Verd: Sortida del sistema posició

Fig. 4.8 Resposta del sistema de la funció de transferència amb llaç tancat

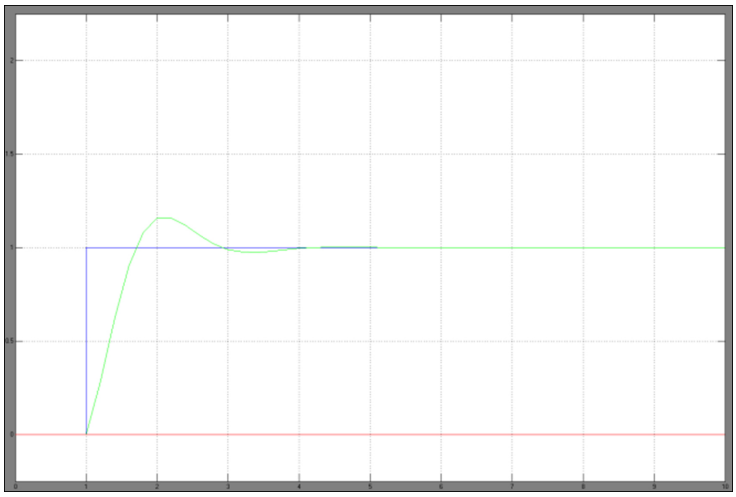


Llegenda:

Blau: Entrada "STEP"

Verd: Sortida del sistema posició

Fig. 4.9 Resposta del sistema amb llaç tancat



Llegenda:

Blau: Entrada "STEP"

Verd: Sortida del sistema posició

Fig. 4.10 Resposta del sistema amb llaç tancat discret

4.4. Disseny del controlador per retorn d'estat.

El control per retorn d'estat tracte de controlar el sistema a traves de la observació de les variables d'estat. En aquest cas el sistema a controlar es de ordre 2 per tan les variables d'estat a observar son X1 i X2. Per realitzar aquest control a traves de la targeta d'adquisició de dades PC-LAB, s'era necessari capturar les dues variables al mateix temps. Això no serà possible ja que la targeta nomes té un convertidor analògic digital. La solució proposada es capturar una variable i canviar el multiplexor de entrada per seleccionar el altre canal a capturar. Com per commutar un canal al altre es necessari un temps per que el multiplexor accioni.

La solució proposada es treballar a un temps de mostreig el doble de ràpid per tal de entrar en la subrutina d'interrupció 2 vegades. La primera captura la senyal de X1 i en la segona captura la senyal de X2 i calcular l'acció de control, es a dir si el sistema esta calculat per un temps de mostreig de 10ms, es treballa 5ms. Això fa que una mostra tingui un retard de 5ms respecte l'altre.

4.4.1. Localització de pols.

El disseny de un controlador per retorn d'estat consisteix en buscar uns pols que multiplicats per cada una de les variables d'estat X1,X2 es poguí controlar el sistema i reduir el seu temps de resposta. Es simularà el sistema amb dos tipus de pols, ITAE.

Càlcul de pols ITAE:

```
P=[0.7071+j*0.7071;0.7071-j*0.7071]
```

4.4.2. Convertir els pols continus a pols discrets.

- Es discretitzà els pols ITAE:

```
pkd=[exp(-(0.7071+j*0.7071)*0.01);exp(-(0.7071-j*0.7071)*0.01)]
pkd = 0.9929 - 0.0070i
      0.9929 + 0.0070i
```

4.4.3. Simulació del sistema en llaç tancat controlat per retorn d'estat.

Càlcul de la matrius K ITAE:

```
k=acker(ad,bd,pkd)
k =    0.4594    0.0481
```

- Simulació del sistema amb pols ITAE:

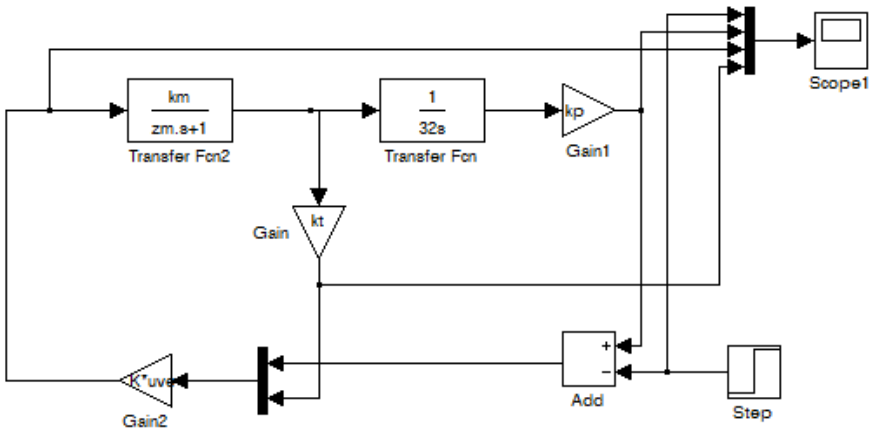
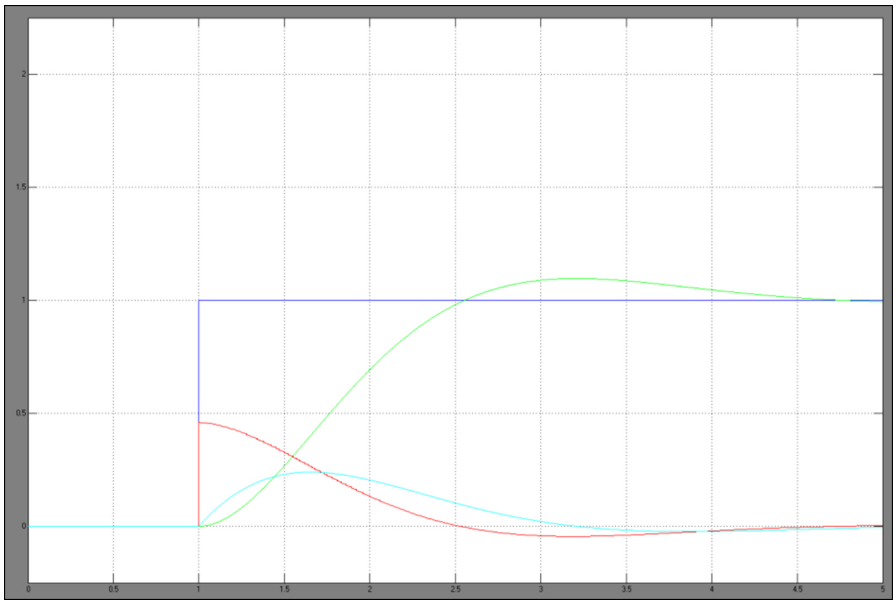


Fig. 4.11 Simulació motor amb controlador per retorn d'estat



Llegenda:

Lila: Entrada del sistema "STEP"

Vermell: Acció de control

Blau: Variable velocitat

Verd: Sortida del sistema posició

Fig. 4.12 Resposta del motor amb controlador per retorn d'estat (ITAE)

4.4.4. Programació per visual basic del programa principal:

En aquest capítol s'explicarà com crear una aplicació per controlar el sistema del motor a traves d'un controlador per retorn d'estat amb la targeta PC-LAB.

En primer lloc s'instal·la els drivers de la targeta PC-LAB tal i com s'ha explicat en el capítol 3.2.3. de aquest mateix document. Un cop estan instal·lats els drivers, es crea un nou projecte de visual basic 5.0.

El programa constarà de dues parts, la part del formulari on es manipulant diferents paràmetres del nostra controlador, i la part de les interrupcions on es realitza el control pas per pas. En primer lloc s'observa en quin port IRQ esta instal·lada la targeta PC-LAB, en aquest cas serà la IRQ 7.

La funció principal de qualsevol aplicació de visual basic és el formulari. Es crea una funció anomenada form_load() que serveix per carregar l'entorn d'usuari, es a dir el interfase gràfic.

```
Private Sub form_load()   ' Creació de la funció
Out &H229, &H8 ' Guany unitari
Out &H22B, &H6 ' Trigger per timer i EOC per interrupció
inter.Text = 5   ' Selecció del temps de mostreig en ms
U = 0 ' Valor desitjat a la sortida
```

```
U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas de escala del
digital / analògic
balt = U \ 256 ' Mascara per agafar el byte baix
bbaix = U - balt * 256 ' Mascara per agafar el byte alt
Call Out(&H224, bbaix) ' Escriu en el port de sortida el valor del byte
baix
Call Out(&H225, balt) ' Escriu en el port de sortida el valor del byte
alt
HW32 = OpenTVicHW32(HW32, "TVICHW32", "TVicDevice0") ' Activació de la
IRQ 7
End Sub ' Final de la funció
```

En la funció anterior s'activen les interrupcions, es posa a 0 v la sortida analògica, es seleccionà el guany del convertidor digital analògic i s'activen les interrupcions quan el convertidor analògic digital acaba de convertir un valor.

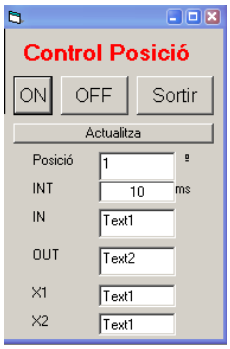


Fig. 4.13 Interfase gràfic control motor per retorn d'estat

En la Fig. 5.14 es veu el entorn gràfic per poder controlar el sistema. Es poden observar diferents botons:

ON : Encendre el controlador, comença a controlar segons la posició introduïda.

OFF: Atura el controlador.

Sortir: Surt de l'aplicació.

Posició: Introduir l'angle que es vol posicionar el motor.

Interrupcions : Es selecciona el temps de mostreig que es vol treballa. (no és aconsellable canviar ja que els càlculs del pols discrets esta fet per aquest temps de mostreig.)

IN: Mostra el valor en Vols de la entrada (Així es pot comprovar que s'estabilitza al valor desitjat.)

OUT: Mostra els valors de la acció de control en Vols.

Explicació de cada boto i funcions en el programa principal:

- Boto ON: Esta lligat a la funció Adquirir_Click

```
Private Sub Adquirir_Click() ' Declaració de la funció
xo = 1 ' Posa el comptador del canal a capturar a 1.
Out &H22A, &H1 ' Selecció el canal 1 de entrada
ref = con.Text ' Captura el valor de consigna del camp de text.
Call ConAngleVols ' Converteix el valor de angle a vols i deixa el valor
a Vref
E = 0 ' Posa a 0 la variable de entrada
U = 0 ' Posa a 0 la variable de sortida
Call UnmaskIRQ(HW32, 7, AddressOf RutinaServei) ' Vincula la interrupció
de IRQ 7 a la rutina d'interrupcions.
Out &H228, &H0 'Activació de la interrupció de la tarja.
Timer1.Enabled = True ' Activa el timer de refresc de pantalla.
Comp = 0 ' Es col·loca a 0 el comptador de les interrupcions
Call ProgramaTimer ' Activar i programa el timer de la tarja
End Sub ' Final de la funció
```

En la funció anterior hi ha una variable anomenada "xo" que es posa a 1, això és perquè al programar un controlador per retorn d'estat, es necessari capturar les dues variables d'estat X1 i X2, però la targeta nomes té un convertidor analògic digital i el multiplexor té un temps de commutació. Per tan la solució que es proposa és treballar amb un temps de mostreig el doble de ràpid i cada cop que s'entra a la funció d'interrupció es captura una de les variables d'estat i en la segona captura pot treure el valor de sortida del control. Això fa que la variable X2 estigui retardada 5ms respecte la variable X1. Aquesta petita diferencia fa que en la resposta no sigui exactament igual que en la simulació del Matlab.

S'ha utilitzat una funció anomenada "ConAngleVols" que consisteix en convertir el valor de referència en el valor en vols corresponent al angle i la funció "ProgramaTimer" que consisteix en escriure els valors de configuració del timer de la targeta. Passem ha analitzar les següents funcions.

- **Funció de conversió d'angle a vols:**

```
Private Sub ConAngleVols
If (ref <= 0) Then
    If (ref >= -180) Then
        Vref = ref / 18
    End If
    If (ref < -180) Then
        Vref = 10 - (-(180 + ref) / 18)
    End If
```

```
End If
If (ref > 0) Then
    If (ref <= 180) Then
        Vref = ref / 18
    End If
    If (ref > 180) Then
        Vref = -(10 + ((180 - ref) / 18))
    End If
End If
End Sub
```

- **Funció programació Timer:**

```
Private Sub ProgramaTimer() ' Declaració funció
t1_H = inter.Text \ 256 ' Captura de la part alta del camp de text
d'interrupcions.
t1_L = inter.Text - t1_H * 256 ' Captura de la part baixa del camp de
text d'interrupcions.
t2_H = 7          ' Programació del segon timer amb un valor prefixat
t2_L = 208      ' de 1 ms com a unitat de temps
Out &H223, &H74 ' Activar el primer timer
Out &H221, t1_L ' Programa byte baix
Out &H221, t1_H ' Programa byte alt
Out &H223, &HB4 ' Activar el segon timer
Out &H222, t2_L ' Programa byte baix
Out &H222, t2_H  ' Programa byte alt
End Sub  ' Final de la funció
```

La targeta esta dotada de 2 temporitzadors de 16 bits, que son utilitzats per activa la interrupció de la targeta. En la funció anterior esta programat el timer de manera què, es selecciona el temps a traves de un camp de text anomenat inter.text.

El primer Timer és amb un temps variable. Es configura a traves del camp de text "inter.Text" amb les unitats de temps de milisegons, aquest depèn del segon Timer que ja esta amb un valor prefixat de milisegons. Posteriorment s'escriu a les adreces corresponents els valors dels Timers i s'activen. On el primer timer s'activa per la direcció "&H74" i el segon per "&HB4".

Un cop es té el controlador en funcionament s'executa regularment una funció de refresc de pantalla que depèn de la funció timer. Aquesta funció s'anomena : "Timer1_Timer", extracte d'un temporitzador per software que estat configurat en 100ms.

- Funció d'actualització de pantalla:

```
Private Sub Timer1_Timer() ' Declaració de la funció
comp_int.Text = Comp ' Actualitzar les vegades que s'ha executat una
rutina d'interrupció.
IND.Text = Format(valor, "###0.00") + " V " ' Mostra per pantalla el
valor de entrada
OUTD.Text = Format(Ureal, "###0.00") + " V " ' Mostra per pantalla el
valor de sortida
```

```
End Sub ' Fi de la funció
```

- Funció d'actualitzar consigna:

```
Private Sub actualitza_Click()' Declaració de la funció
ref = con.Text ' Capura del camp de text el valor de consigna
Call ConAngleVols ' Conversió de angle a vols
End Sub ' Fi de la funció
```

-Funció OFF "Aturar"

```
Private Sub Atura_Click()' Declaració de la funció
Call Out(&H20B, &H0)  ' Desactiva les interrupcions del convertidor
Call Out(&H223, &H74) ' Desactiva el timer 1
Call Out(&H221, &H0) ' Desactiva escriu 0 al timer 1
Call Out(&H223, &HB4) ' Desactiva el timer 2
Call Out(&H222, &H0) ' Desactiva escriu 0 al timer 2
Out &H22A, &H1 ' Selecciona l'entrada 1 del convertidor digital/analògic
Call MaskIRQ(HW32, 7) ' Desactiva les interrupcions IRQ 7
End Sub ' Fi de la funció
```

-Funció Sortir:

```
Private Sub Command1_Click() ' Declaració de la funció
'9 SORTIR tancat el Form
Unload Form1 'descarrega el form i tanca el programa
End Sub ' Fi de la funció
```

En el programa principal s'ha vist totes les funcions que s'utilitzen, per poder fer anar totes aquests funcions és necessari declarar les següents variables:

```
Dim HW32 As Long ' IRQ
Dim t1_H As Byte ' Timer 1 byte alt
Dim t1_L As Byte ' Timer 1 byte baix
Dim t2_H As Byte ' Timer 2 byte alt
Dim t2_L As Byte   ' Timer 2 byte baix
```

4.4.5. Programació per Visual basic de la rutina d'interrupció

La rutina d'interrupció, es la rutina on esta implementada tot el control del sistema. Aquesta funció esta ubicada en un mòdul extern que s'ha creat a dins del projecte anomenat "Modules". Esta formada per una sola funció anomenada Rutina de Servei.

En primer lloc s'utilitza unes variables K1 i K2 on introduïm els valors de la matriu de control calculada anteriorment en el punt 5.4.3. Després s'utilitza el comptador de interrupcions per saber quants cops s'ha executat aquesta funció. A partir de aquí entra la part de control, amb una condició "If else" es determina si es necessari captura el valor de X1 (Posició) o el valor de X2 (Velocitat).

En la condició If es mira quin valor té la variable XO per saber quina variable d'estat tenim capturada. Primer consultem si Xo és igual a 1, si es compleix aquesta condició es

capturarà la variable d'esta X1 (posició). Posteriorment es canvia el valor de Xo per 2 i es canvia també el canal d'entrada per el canal 2.

Si la condició IF detecta que la variable XO és igual a 2 es captura la senyal X2 (velocitat). Posteriorment es canvia el valor de Xo per 1 i es canvia també el canal d'entrada per el canal 1.

Ara, ja hi ha les dues variable d'estat capturades. Es passar a calcular el valor per la sortida. Es resta el valor de la consigna a la entrada de posició i es calcula la sortida, multiplicant els valors de la posició restada amb la consigna per la matriu de control K1 més el valor de velocitat per la matriu de control K2.

Es converteix el valor de la sortida en pas del digital analògic i s'escriu en la adreça del convertidor.

- **Funció d'interrupció:**

```
Public lectura As Double ' Variable de tipus double per capturar el valor
del A/D
Public U As Double ' Variable de tipus double on es guarda la sortida
Public xo As Integer ' Variable de tipus Integer per indicar quin valor
Public K1 As Double ' Variable de tipus Double de la matriu de control
Public K2 As Double ' Variable de tipus Double de la matriu de control
Public X1 As Double ' Variable de tipus Double de la variable d'estat X1
posició
Public X2 As Double ' Variable de tipus Double de la variable d'estat X1
velocitat
Public Vref As Double ' Variable de tipus Double de la consigna
Public Ureal As Double ' Variable de tipus Double de la sortida
Public Comp As Double ' Variable de tipus Double del comptador
d'interrupcions
Public Sub RutinaServei(ByVal x As Integer) ' Declaració de la funció
      K1 = 0.4594 ' Valor de K ITAE
      K2 = 0.0481 ' Valor de K ITAE
    Comp = Comp + 1 ' Compta cops que entra a la funció d'interrupcions
    If (xo = 1) Then ' Condició de la variable a capturar
       xo = 2 ' Canvi del indicador de capturar
       balt = Inp(&H225) And &HF ' Captura valor byte baix del analògic
digital
       bbaix = Inp(&H224) And &HFF ' Captura valor byte alt del analògic
digital
       lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un
double
       X1 = ((lectura * 20#) / 4095) - 10 ' Conversió a vols
       Out &H22A, &H2    ' Seleccionar de entrada canal 2
    Else
        xo = 1 ' Canvi del indicador de capturar
        balt = Inp(&H225) And &HF ' Captura valor byte baix del analògic
digital
        bbaix = Inp(&H224) And &HFF ' Captura valor byte alt del analògic
digital
```

```
        lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un
double
        X2 = ((lectura * 20#) / 4095) - 10 ' Conversió a vols
        Out &H22A, &H1 ' Seleccionar canal 1
         X1r = -Vref + X1 ' Resta consigna menys posició
         Ureal = -K1 * X1r + -K2 * X2 ' Càlcul controlador
         U = Ureal ' Canvi de variable el valor.
         If (U > 9.9) Then ' Limits de 10 v
              U = 9.9
         End If
         If (U < -9.9) Then
              U = -9.9
         End If
         U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas del
convertidor D/A

         balt = U \ 256  ' Separa byte alt
         bbaix = U - balt * 256 ' Separar byte baix
         Call Out(&H224, bbaix) ' Escriu el byte baix al convertidor
analògic digital
         Call Out(&H225, balt) ' Escriu el byte alt al convertidor
analògic digital
         Out &H228, &H0 ' Activar les interrupcions
    End If
End Sub
```

4.4.6. Posta en marxa del sistema.

Per posar en marxa el sistema compilem la aplicació i connectem (Fig. 5.15):

- Entrada 1 de la PC-LAB amb la sortida de posició de la placa electrònica que incorpora el motor.

- Entrada 2 de la PC-LAB amb la sortida de velocitat de la placa electrònica que incorpora el motor.

- Sortida 1 de la PC-LAB amb l'entrada positiva de la placa electrònica que incorpora el motor.

- Connectar la alimentació del motor a ±15v i 5v.

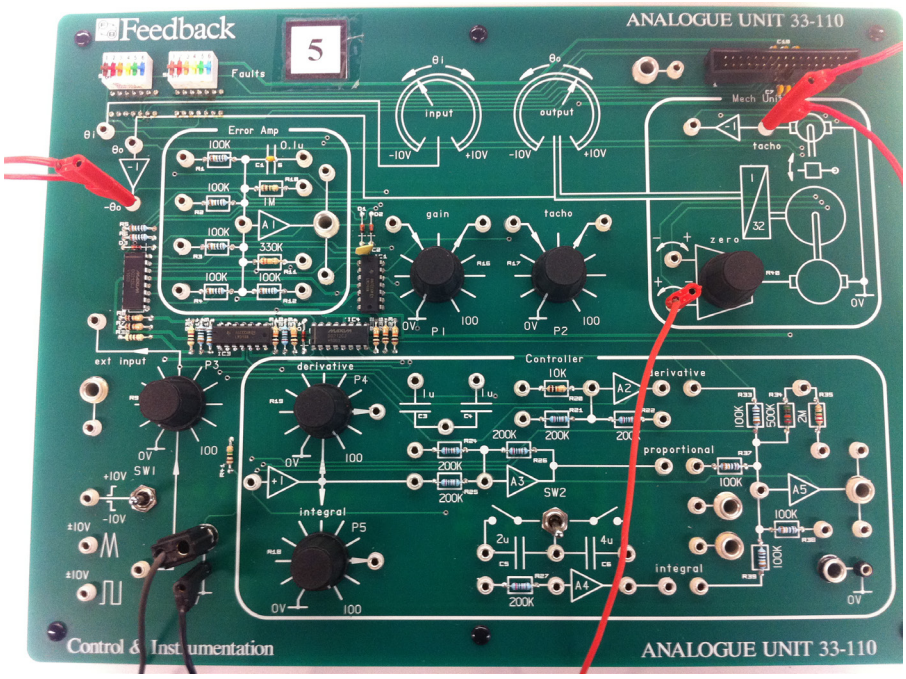
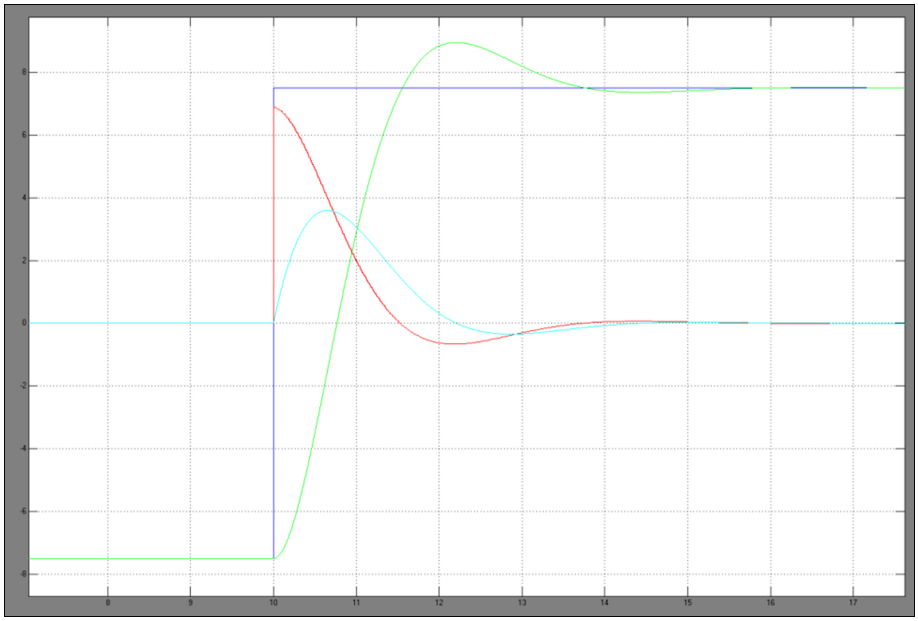


Fig. 4.14 Connexió Motor

4.4.7 Comprovació dels resultats obtinguts.

La comprovació s'ha fet a traves de diferents mesures a diferents posicions comparant amb la corba obtinguda des de el Matlab.

Posició de -135º a 135º (Teòricament 135º són 7.5v ja que 10v són 180º) ITEA:



Llegenda:

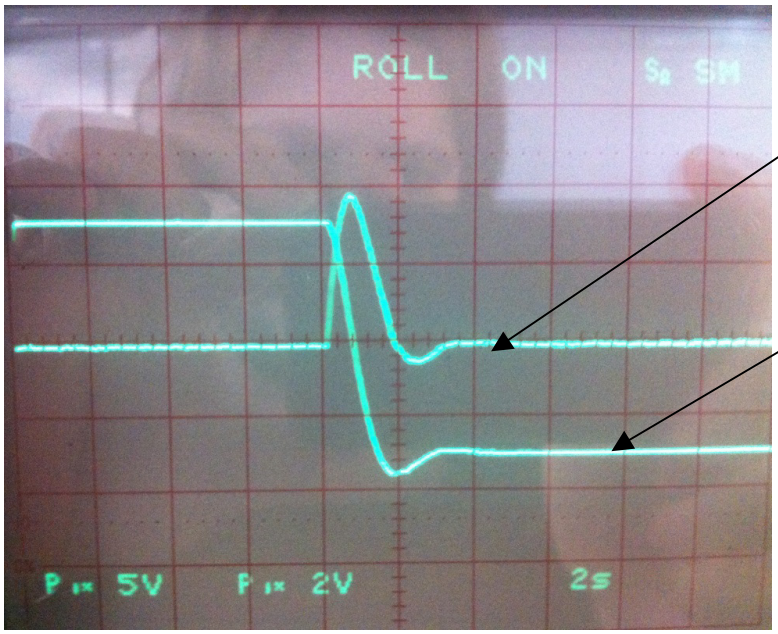
Lila: Entrada del sistema "STEP"

Vermell: Acció de control

Blau: Variable velocitat

Verd: Sortida del sistema

Fig. 4.15 Simulació Motor per retorn d'estat amb pols ITAE de -135º a 135º



Llegenda:

Canal 1: 5 v/div

Senyal de velocitat motor

Canal 2: 2 v/div

Sortida

Temps: 2 s/div

Fig. 4.16 Sistema real per retorn d'estat amb pols ITAE de -135º a 135º

Comparant els dos valors, de les dos gràfiques (4.15 i 4.16), s'estabilitzen a -7.5v i amb un temps de 3.5 segons.

En les figures anterior Fig. 4.15 i Fig.4.16, es pot observar la resposta del control per retorn d'estat del motor. Comparant les senyal adquirides amb l'oscil·loscopi i les senyals simulades amb el Matlab, es veu clarament que són exactament iguals, això significa que el controlador esta funcionant correctament encara que una de les mostres tingui un retard de 5 ms. El perque no es nota aquesta diferencia es perque la diferencia del temps de mostreix i el temps de resposta del sistema hi ha molta diferencia, per tan 5 ms són insignificant comparat amb 3.5 seg.

4.5 Disseny del controlador per retorn d'estat amb observador

4.5.1.Funcionament de l'observador predictiu

El observador predictiu tal com esta explicat en el capítol 4, funció de manera que captura les variable d'estat en un temps de mostreig inferior al temps que surt la sortida això fa que el càlcul del control sigui mes ràpid. Aquests tipus de observadors es possible simular amb el Matlab.

4.5.2.Funcionament de l'observador corrent

A diferencia de l'observador predictiu, l'observador corrent no és possible simular a traves del Matlab. Es un controlador utilitzat per ordinadors amb un temps de càlcul molt ràpid o temps de mostreig.

L'observador corrent, funciona de manera que es calcula en el instant k el valor que surt. Sempre està en mateix instant que és fa tot. Aquest procediment es el més lògic de funcionament, però si l'ordinador on s'està implementant el temps de càlcul és llarg, la sortida que surt té un retard.

4.5.3.Càlcul de pols de l'observador

El càlcul d'un observador consisteix en col·locar un pols més allunyats dels pols del controlador. Aprofitant els pols calculats anteriorment en el apartat 5.4.2

- Pols ITAE

```
pkd =  0.9929 - 0.0070i
       0.9929 + 0.0070i
```

Càlcul del pols de l'observador ITAE:

```
>> po=pkd/2
       po =   0.4930 + 0.0072i
              0.4930 - 0.0072i
```

Càlcul de la Matriu L a traves de Ackerman ITAE:

```
>> L=acker(ad',cd',po)'
       L =    0.9939
             23.9500
```

4.5.4 Creació del controlador:

```
acon=ad-bd*k-L*cd
bcon=L
ccon=k
dcon=0
```

Controlador resultant per un sistema amb observador i pols ITAE:

```
acon =   0.0014    0.0094
          -23.9811    0.9766
bcon =     0.9939
          23.9500
ccon =    0.4594    0.0481
dcon=0
```

4.5.5 Simulació del sistema en llaç tancat controlat per retorn d'estat amb observador.

- Simulació del sistema a traves del Simulink:

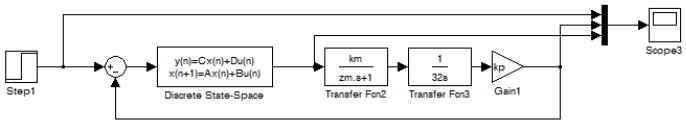
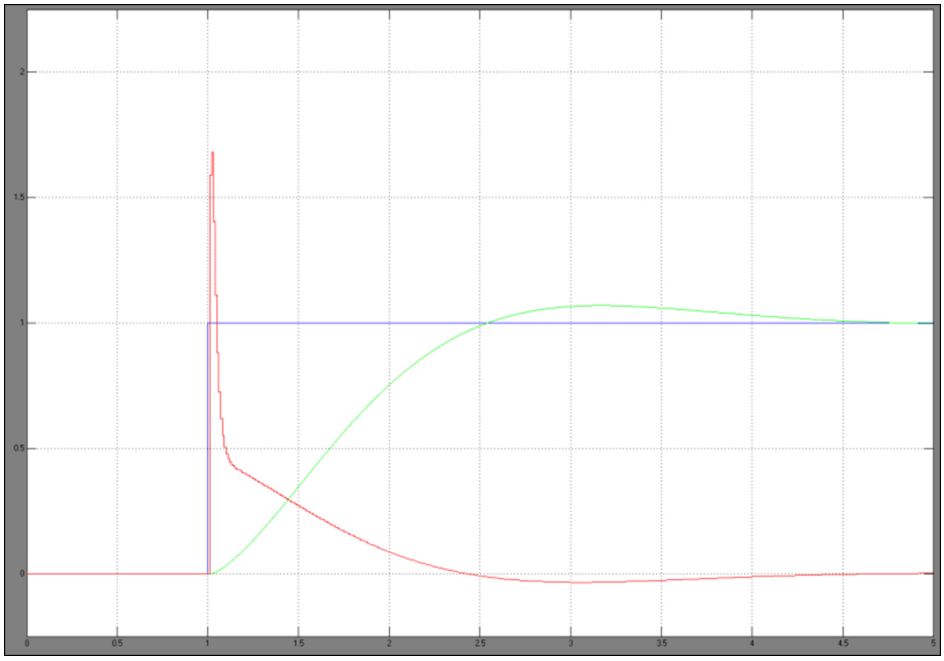


Fig. 4.17 Simulació sistema amb llaç tancat amb observador



Llegenda:

Lila: Entrada del sistema "STEP"

Vermell: Acció de control

Verd: Sortida del sistema

Fig. 4.18 Resposta del motor amb observador (ITAE)

4.5.6. Programació per Visual basic del programa principal:

En aquest capítol s'explicarà com crear una aplicació per controlar el sistema del motor a traves d'un controlador per retorn d'estat amb observador. El funcionament del programa és casi igual que el explicat en el punt 4.4.4. En aquest capítol es mencionant els canvis realitzats respecte el programa del control per retorn d'estat sense observador (Punt 4.4.4).

En les funcions que té el programa principal, no té cap diferencia excepte en les funcions d'adquirir, refresc de pantalla i l'entorn gràfic. Es comença observant el entorn gràfic, on s'ha posat, diferents camps per poder observar tot el que fa el controlador.

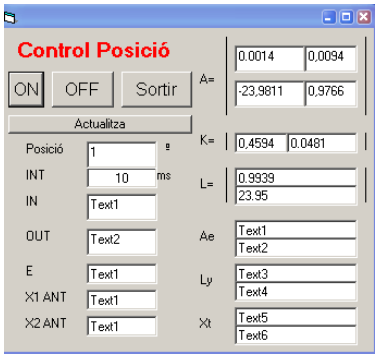


Fig. 4.19 Entorn gràfic control motor amb observador

En la Fig. 4.19 es veu el entorn gràfic per poder controlar el sistema. Es poden observar diferents botons:

ON : Encendre el controlador, comença a controlar segons la posició introduïda.

OFF: Atura controlador.

Sortir: Surt de l'aplicació.

Posició: Introduir el angle que es vol posicionar el motor.

Interrupcions : Es selecciona el temps de mostreig en que es treballa. (no es aconsellable canviar ja que els càlculs del pols discrets estan fets per aquest temps de mostreig.)

IN: Mostra el valor en Vols de la entrada (Així es pot comprovar que s'estabilitza al valor desitjat.)

OUT: Mostra els valors de la acció de control en Vols.

Matriu A: S'observa els valors calculats anteriorment.

Matriu K: S'obseva el valors calculats de la matriu K o que és el mateix de la matriu C del controlador.

Matriu L: S'obseva el valors calculats de la matriu L o que és el mateix de la matriu B del controlador.

Matriu Ae: És la matriu A multiplicada per les sortides anterior a la mostra actual.

Matriu Ly: És la matriu L multiplicada per l'entrada menys la consigna.

Matriu Xt: Variable d'estat X1 i X2, aproximació de l'observador.

Valor E: L'entrada menys la consigna.

Valor X1 ANT: Valor de la sortida en vols d'una mostra anterior.

Valor X2 ANT: Valor de X2 en vols de un mostra anterior.

- Boto ON: Esta lligat a la funció Adquirir_Click

En aquesta funció s'ha eliminat la línea de codi de "xo" ja que no és necessària perquè en un sistema amb observador nomes es captura la realimentació que és la variable d'estat X1 (posició).

4.5.7. Programació per Visual basic de la rutina d'interrupció del observador predictiu

La rutina d'interrupció, és la rutina on esta implementada tot el control del sistema. Aquesta funció esta ubicada en un mòdul extern que hem creat a dins del projecte anomenat "Modules". Esta formada per una sola funció anomenada Rutina de Servei.

En primer lloc s'analitzarà el control per un observador predictiu, aquests tipus de observadors són els més utilitzats, ja que van be per ordinadors lents o sistemes molts ràpids. En aquest cas en la mateix execució es calcula i surt el valor de control.

Es converteix el valor de la sortida en pas del digital analògic i s'escriu en la adreça del convertidor.

- **Funció d'interrupció:**

```
Public lectura As Double ' Variable de tipus double per capturar el valor del A/D
Public valor As Double ' Variable de tipus double, valor de l'entrada en Vols
Public U As Double ' Variable de tipus double on es guarda la sortida
Public EANT As Double ' Variable de tipus double la variable de X1 anterior
Public E2ANT As Double ' Variable de tipus double la variable de X2 anterior
Public Xt1 As Double ' Variable de tipus double on es guarda l'aproximació de X1 (posició)
Public Xt2 As Double ' Variable de tipus double on es guarda l'aproximació de X2 (velocitat)
```

```
Public Ae1 As Double ' Variable de tipus double, de la multiplicació de A
per sortides anteriors
Public Ae2 As Double ' Variable de tipus double, de la multiplicació de A
per sortides anteriors
Public Ly1 As Double ' Variable de tipus double on és la matriu L
multiplicada per l'entrada menys la consigna.
Public Ly2 As Double ' Variable de tipus double on és la matriu L
multiplicada per l'entrada menys la consigna.
Public Vref As Double ' Variable de tipus Double de la consigna
Public E As Double ' Variable de tipus double de la entrada menys la
consigna
Public Ureal As Double ' Variable de tipus Double de la sortida
Public Comp As Double ' Variable de tipus Double del comptador
d'interrupcions
Public Sub RutinaServei(ByVal x As Integer) ' Declaració de la funció
    Comp = Comp + 1 ' Compta cops que entra a la funció d'interrupcions
    balt = Inp(&H225) And &HF ' Captura valor byte baix de l'analògic
digital
    bbaix = Inp(&H224) And &HFF ' Captura valor byte alt de l'analògic
digital
    lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un double
    valor = ((lectura * 20#) / 4095) - 10 ' Conversió a vols
    E = Vref - valor ' Realimentació/ sortida del sistema menys la
consigna.
   Ae1 = 0.0014 * EANT + 0.0094 * E2AN' Multiplicació de la matriu a per
    Ae2 = -23.9811 * EANT + 0.9766 * E2ANT 'les variables d'estat d'una
mostra anterior
    Ly1 = 0.9939 * E ' Multiplicació de l'observador per l'entrada del
controlador
    Ly2 = 23.95 * E ' Multiplicació de l'observador per l'entrada del
controlador
    Xt1 = Ae1 + Ly1 ' Aproximació de X1
    Xt2 = Ae2 + Ly2 ' Aproximació de X2
    Ureal = 0.4594 * Xt1 + 0.0481 * Xt2 ' Càlcul del controlador
    EANT = Xt1 ' Guarda la variable d'estat X1
    E2ANT = Xt2  ' Guarda la variable d'estat X2
    U = Ureal ' Canvi de variable el valor.
   If (U > 9.9) Then ' Límits de ±10 v
             U = 9.9
   End If
   If (U < -9.9) Then
             U = -9.9
   End If
   U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas del convertidor
D/A
   balt = U \ 256  ' Separa byte alt
   bbaix = U - balt * 256 ' Separar byte baix
   Call Out(&H224, bbaix) ' Escriu el byte baix al convertidor analògic
digital
   Call Out(&H225, balt) ' Escriu el byte alt al convertidor analògic
digital
   Out &H228, &H0 ' Activar les interrupcions
End Sub
```

4.5.8. Programació per Visual basic de la rutina d'interrupcions amb observador corrent.

Tal i com s'ha descrit a la part de teoria de l'observador corrent. Primer es calculant les dues variables d'estat del nostre sistema, i es calcula la senyal de sortida.

Posteriorment fem el càlcul amb els valors de la sortida de aquest instant de temps per poder calcular com varien les variables d'estat en el proper instant de temps.

- **Funció d'interrupció:**

```
Public lectura As Double ' Variable de tipus double per capturar el valor del A/D
Public valor As Double ' Variable de tipus double, valor de l'entrada en Vols
Public U As Double ' Variable de tipus double on es guarda la sortida
Public Xp1 As Double ' Variable de tipus double on es guarda l'aproximació de X1 (posició) anterior
Public Xp2 As Double ' Variable de tipus double on es guarda l'aproximació de X2 (velocitat) anterior
Public X1 As Double ' Variable de tipus double on es guarda l'aproximació de X1 (posició)
Public X2 As Double ' Variable de tipus double on es guarda l'aproximació de X2 (velocitat)
Public Ae1 As Double ' Variable de tipus double, de la multiplicació de A per sortides anteriors
Public Ae2 As Double ' Variable de tipus double, de la multiplicació de A per sortides anteriors
Public Bu1 As Double ' Variable de tipus double on és la matriu L multiplicada per la sortida.
Public Bu2 As Double ' Variable de tipus double on és la matriu L multiplicada per la sortida.
Public Vref As Double ' Variable de tipus Double de la consigna
Public E As Double ' Variable de tipus double de l'entrada menys la consigna
Public Ureal As Double ' Variable de tipus Double de la sortida
Public Comp As Double ' Variable de tipus Double del comptador d'interrupcions
Public Sub RutinaServei(ByVal x As Integer) 'Declaració de la funció
    Comp = Comp + 1 ' Compta cops que entra a la funció d'interrupcions
    balt = Inp(&H225) And &HF ' Captura valor byte baix de l'analògic digital
    bbaix = Inp(&H224) And &HFF ' Captura valor byte alt de l'analògic digital
    lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un double
    valor = ((lectura * 20#) / 4095) - 10 ' Conversió a vols
    E = Vref - valor ' Realimentació/ sortida del sistema menys la consigna.
   X1 = Xp1 + 0.9939 * (E - 1 * Xp1) ' Càlcul de X1 (posició) a traves del Observador i X1 anterior
   X2 = Xp2 + 23.95 * (E - 1 * Xp1) ' Càlcul de X2 (velocitat) a traves de l'observador i X2 anterior
    Ureal = 0.4594 * X1 + 0.0481 * X2 ' Càlcul de la sortida
    Ae1 = 0.0014 * X1 + 0.0094 * X2 ' Matriu A per les variables d'estat
```

```
    Ae2 = -23.9811 * X1 + 0.9766 * X2 ' Matriu A per les variables
d'estat
    Bu1 = 0.9939 * Ureal ' Càlcul de "B" o "L" per la sortida
    Bu2 = 23.95 * Ureal ' Càlcul de "B" o "L" per la sortida
    Xp1 = Ae1 + Bu1 ' Càlcul X1 en la mostra anterior
    Xp2 = Ae2 + Bu2 ' Càlcul X2 en la mostra anterior
    U = Ureal ' Canvi de variable el valor.
    If (U > 9.9) Then ' Limits de ±10 v
              U = 9.9
   End If
   If (U < -9.9) Then
              U = -9.9
   End If
   U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas del convertidor
D/A
   balt = U \ 256  ' Separa byte alt
   bbaix = U - balt * 256 ' Separar byte baix
   Call Out(&H224, bbaix) ' Escriu el byte baix al convertidor A/D
   Call Out(&H225, balt) ' Escriu el byte alt al convertidor A/D
   Out &H228, &H0 ' Activar les interrupcions
End Sub
```

4.5.9. Posta en marxa del sistema.

Per posar en marxa el sistema es compilà l'aplicació i es connectà (Fig. 4.20):

- Entrada 1 de la PC-LAB amb la sortida de posició de la placa electrònica que incorpora el motor.

- Sortida 1 de la PC-LAB amb l'entrada positiva de la placa electrònica que incorpora el motor.

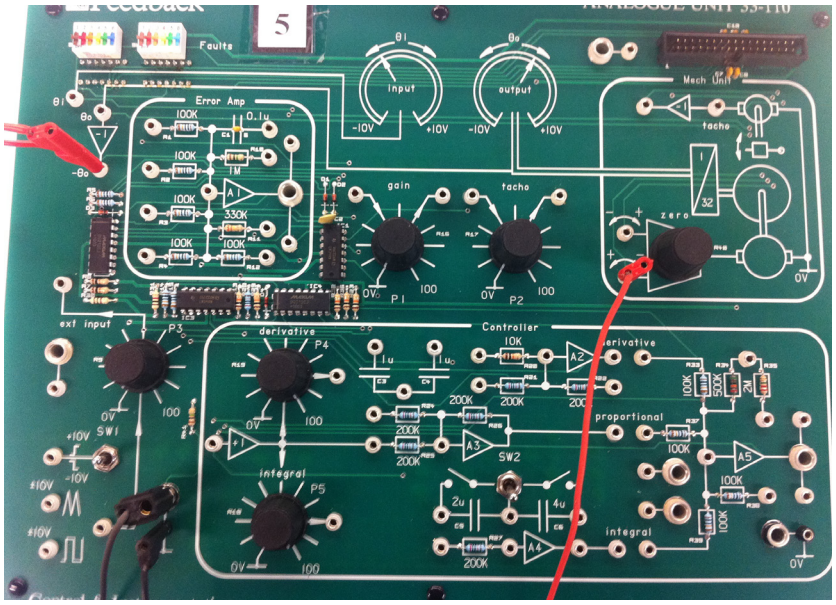
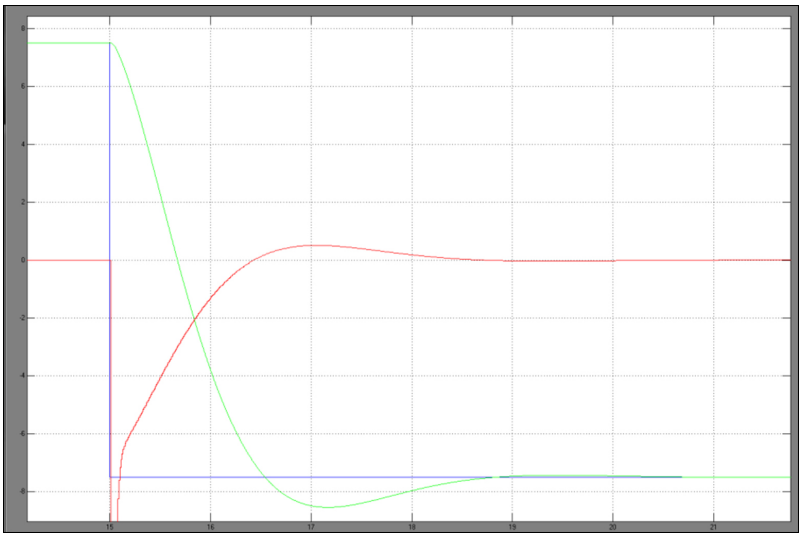


Fig. 4.20 Connexió Motor

4.5.10. Comprovació dels resultats obtinguts amb observador predictiu.

La comprovació s'ha fet a traves de diferents mesures a diferents posicions comparant amb la corba obtinguda des del Matlab.

Posició de -135º a 135º (Teòricament 135º són 7.5v ja que 10v són 180º) ITEA:



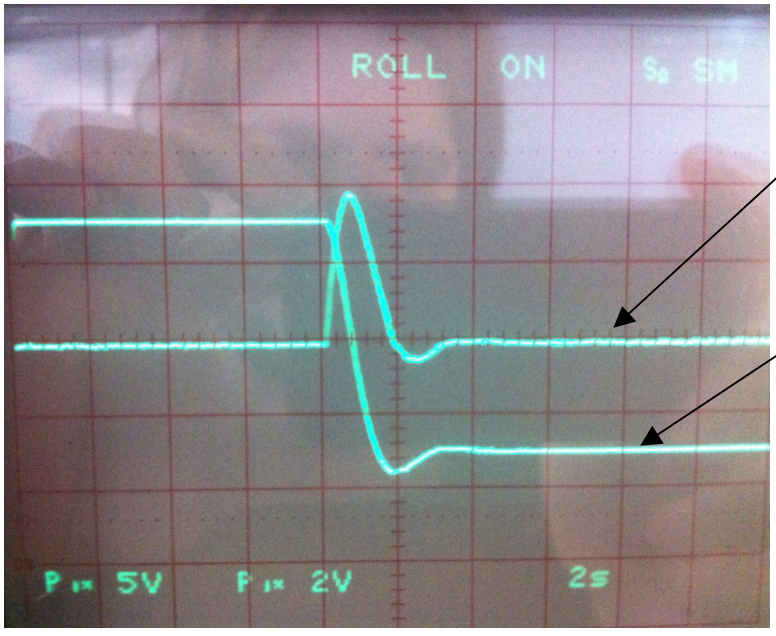
Llegenda:

Lila: Entrada del sistema "STEP"

Vermell: Acció de control

Verd: Sortida del sistema posició

Fig. 4.21 Simulació Motor per retorn d'estat amb pols ITAE de -135º a 135º



Llegenda:

Canal 1: 5 v/div

Senyal de velocitat motor

Canal 2: 2 v/div

Sortida

Temps: 2 s/div

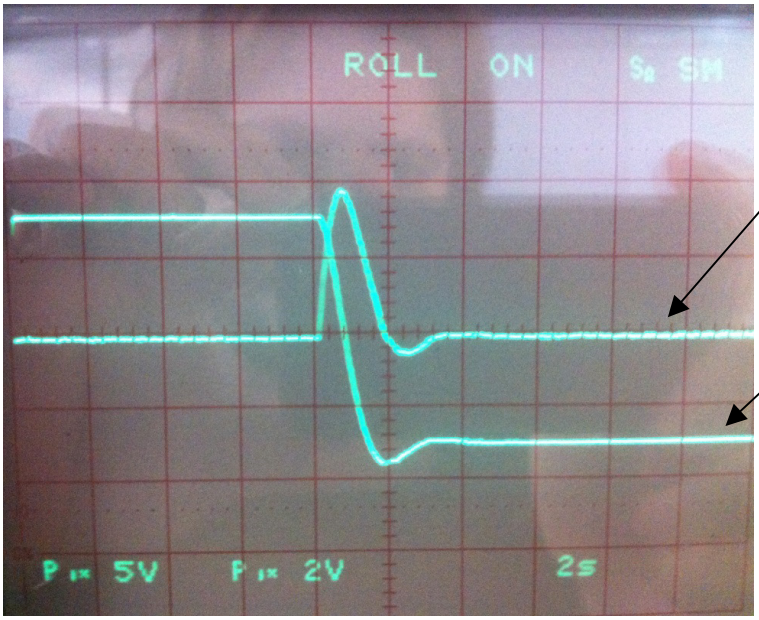
Fig. 4.22 Sistema real per retorn d'estat amb pols ITAE de -135º a 135º

Comparant els dos valors, els dos s'estabilitzen a -7.5v i amb un temps de 3.5 segons.

4.5.11. Comprovacions del resultats obtinguts amb observador corrent.

En aquest cas no es pot comprovar el seu funcionament amb el Matlab ja que no es pot crear un observador corrent amb el Matlab. Però si que podem comprovar el comportament del observador predictiu amb el observador corrent, ja que ha de tenir la mateix resposta.

Posició de -135º a 135º (Teòricament 135º són 7.5v ja que 10v són 180º) ITEA:



Llegenda:

Canal 1: 5 v/div

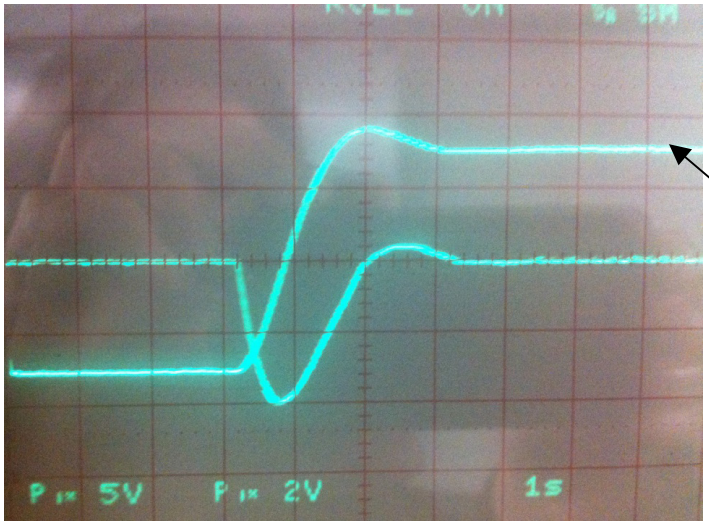
Senyal de velocitat motor

Canal 2: 2 v/div

Sortida

Temps: 2 s/div

Fig. 4.23 Sistema real amb observador predictiu amb pols ITAE de -135° a 135°



Llegenda:

Canal 1: 5 v/div

Senyal de velocitat motor

Canal 2: 2 v/div

Sortida

Temps: 1 s/div

Fig. 4.24 Sistema real amb observador corrent amb pols ITAE de -135° a 135°

Comparant els dos valors, (Figures 5-31 i 5.32) s'estabilitzen a -7.5v i amb un temps de 3.5 segons. Com es pot veure en els figures anteriors no hi ha pràcticament diferencia entre el observador predictiu i el corrent, ja que el sistema es molt lent, el ordinador li dona temps de calcular les variables d'estat.

4.6. Disseny del controlador per retorn d'estat amb observador reduït.

4.6.1. Creació de les Matrius:

En primer lloc, s'ha de crear les matrius reduïdes per poder crear el observador. De manera que s'aprofitarà les matrius del sistema que ja estan introduïdes en el Matlab per crear les noves:

```
>> Aaa=ad(1,1)
Aaa =     1
>> Aab=ad(1,2)
Aab =    0.0099
>> Aba=ad(2,1)
Aba =     0
>> Abb=ad(1,2)
Abb =    0.0099
>> Abb=ad(2,2)
Abb =    0.9798
>> Ba=bd(1,1)
Ba =    0.0103
>> Bb=bd(2,1)
Bb =    0.0676
```

4.6.2 Càlcul pols observador reduït:

Els pols del controlador son els mateixos, per tant la K del sistema és la mateixa, però és separa la matriu K en Ka i Kb per posteriorment poder calcular el controlador.

Pols ITAE:

```
>> Ka=k(1,1)
Ka =    0.4594
Kb=k(1,2)
Kb =    0.0481
```

Pols del observador:

```
>> Po=0.0990049833749
>> Lr=acker(Abb',Aab',Po)'
Lr =   -9.0019
```

4.6.3 Creació del controlador.

S'aplica la teoria del observador reduït:

```
>> Ar=Abb-Lr*Aab+(Bb-Lr*Ba)*(-Kb);
>> Br=Ar*Lr+Aba-Lr*Aaa+(Bb-Lr*Ba)*(-Ka)
>> Cr=-Kb
>> Dr=-Ka-Kb*Lr
```

4.6.4 Simulació del sistema en llaç tancat controlat amb un observador reduït.

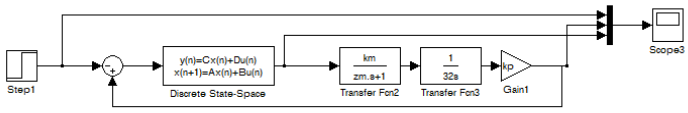
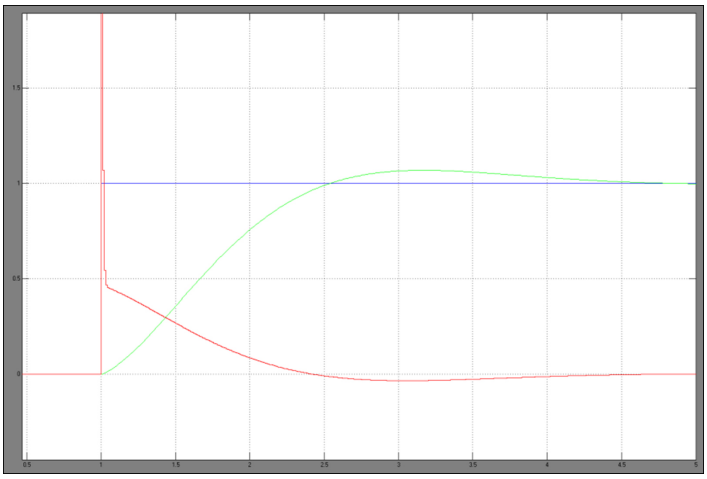


Fig. 4.25 Simulink del sistema amb control per retorn d'estat amb observador reduït

Resposta del sistema:



Llegenda:

Lila: Entrada del sistema "STEP"

Vermell: Acció de control

Verd: Sortida del sistema

Fig. 4.26 Resposta de la simulació amb pols ITAE.

4.6.5. Programació per Visual basic del programa principal.

La programació del programa principal té que ser el mateix que la programació del sistema per un observador predictiu.

En aquest cas el control del observador reduït s'ha implementat igual que el observador predictiu ja que es la manera més comuna de implementar un observador. Però també es podria implementa un observador reduït de la forma corrent. La diferencia de un observador reduït a un observador complet es nomes el ordre del controlador i la forma de implementar es indiferent.

4.6.6. Programació per Visual basic de la rutina d'interrupcions.

Com s'ha comentat en el apartat anterior 5.7.4, la implementació del observador reduït es de igual que el observador complet predictiu però de ordre n-1 es a dir que en el cas de un sistema de ordre 2 el controlador es de ordre 1.

- **Funció d'interrupcions:**

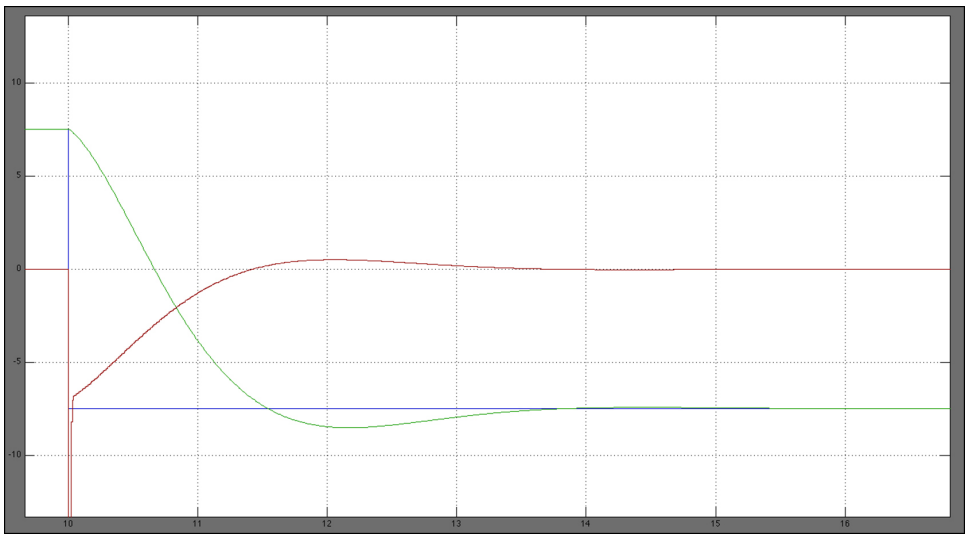
```
Public lectura As Double ' Variable de tipus double per capturar el valor
del A/D
Public valor As Double ' Variable de tipus double, valor de la entrada en
Vols
Public U As Double ' Variable de tipus double on es guarda la sortida
Public EANT As Double ' Variable de tipus double la variable de X1
anterior
Public E2ANT As Double    ' Variable de tipus double la variable de X2
anterior
Public Xt1 As Double ' Variable de tipus double on es guarda
l'aproximació de X1
Public Ar As Double ' Variable de tipus double, Matriu A del controlador
Public  Br As Double ' Variable de tipus double, Matriu B del controlador
Public Cr As Double ' Variable de tipus double, Matriu C del controlador
Public  Dr As Double ' Variable de tipus double, Matriu D del controlador
Public Ae As Double ' Variable de tipus double, de la multiplicació de A
per sortides anteriors
Public By As Double ' Variable de tipus double on es la matriu B
multiplicada per la entrada menys la consigna.
Public Vref As Double ' Variable de tipus Double de la consigna
Public E As Double ' Variable de tipus double de la entrada menys la
consigna
Public Ureal As Double ' Variable de tipus Double de la sortida
Public Comp As Double ' Variable de tipus Double del comptador
d'interrupcions
Public Sub RutinaServei(ByVal x As Integer) ' Declaració de la funció
      Ar = 0.14
      Br = -76.134
      Cr = -0.0481
      Dr = -4.7367
    Comp = Comp + 1 ' Compta cops que entra a la funció d'interrupcions
    balt = Inp(&H225) And &HF ' Captura valor byte baix del analògic
digital
    bbaix = Inp(&H224) And &HFF ' Captura valor byte alt del analògic
digital
    lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un double
    valor = ((lectura * 20#) / 4095) - 10 ' Conversió a vols
    E = Vref - valor ' Realimentació/ sortida del sistema menys la
consigna.
    Ae = Ar * EANT  ' Matriu A per la variable d'estat X1 anterior
    By = Br * E     ' Matriu B per entrada del controlador
    Xt1 = Ae + By   ' Aproximació de X1
    Ureal = Cr * Xt1 + Dr * E  ' Càlcul del controlador
    EANT = Xt1 ' Guarda la variable d'estat X1
    U = Ureal ' Canvi de variable el valor.
   If (U > 9.9) Then ' Límits de ±10 v
             U = 9.9
   End If
   If (U < -9.9) Then
```

```
            U = -9.9
   End If
   U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas del convertidor
D/A
   balt = U \ 256  ' Separa byte alt
   bbaix = U - balt * 256 ' Separar byte baix
   Call Out(&H224, bbaix) ' Escriu el byte baix al convertidor analògic
digital
   Call Out(&H225, balt) ' Escriu el byte alt al convertidor analògic
digital
   Out &H228, &H0 ' Activar les interrupcions
End Sub
```

4.6.7. Comprovacions dels resultats obtinguts.

La comprovació s'ha fet a traves de diferents mesures a diferents posicions comparant amb la corba obtinguda des del Matlab amb la corba capturada per el oscil·loscopi.

Posició de -135º a 135º (Teòricament 135º són 7.5v ja que 10v són 180º) ITEA:



Llegenda:

Lila: Entrada del sistema "STEP"

Vermell: Acció de control

Verd: Sortida del sistema

Fig. 4.27 Simulació Motor per retorn d'estat amb pols ITAE de -135º a 135º

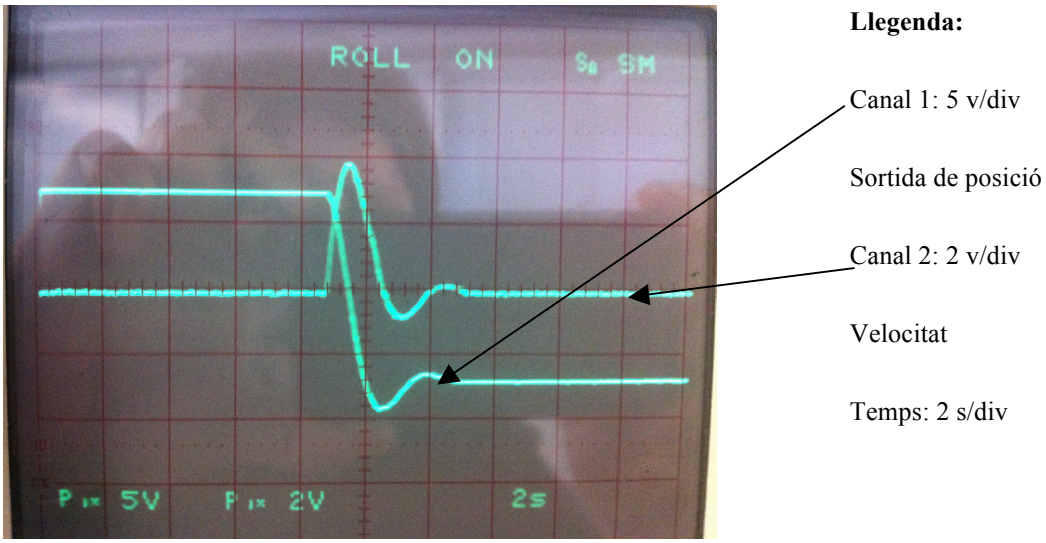


Fig. 4.28 Sistema real per retorn d'estat amb pols ITAE de -135º a 135º

Comparant els dos valors, els dos s'estabilitzen a -7.5v i amb un temps de 3.5 segons.

En el observador reduït es pot observar que en les simulacions del Matlab hi ha un impuls en la senyal de control que en el sistema real no esta. Això es perquè en el sistema real no li dona temps de fer aquest sobre impuls.

5. Control de Feedback (2LAGS + INTEGRADOR).

5.1. Introducció del sistema.

El sistema esta compost de una sèrie de operacionals que simulant lag o integradors. Em aquest cas s'ha utilitzat dos lag i un integrador amb una constant de temps de 10ms cada un. Així serà un sistema bastant ràpid i podrem comprovar les funcions de les targetes PC-LAB treballant en temps real.

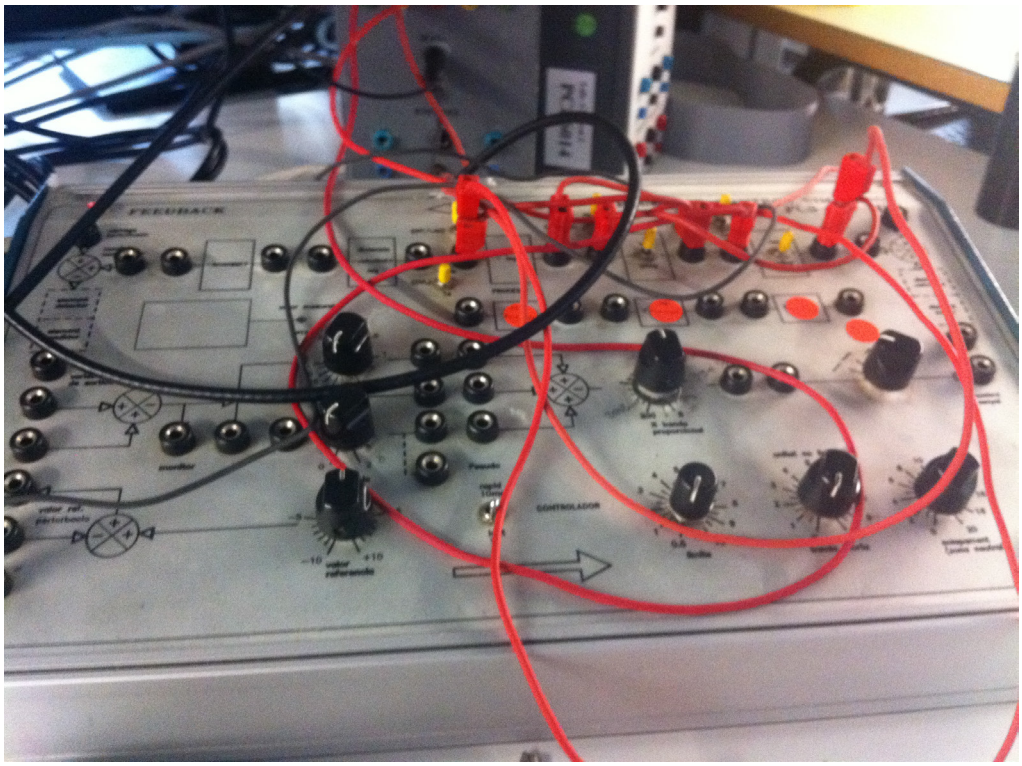


Fig. 5.1 Sistema Feedback

Aquest sistema permet capturar les tres variables d'estat, ja que els dos lags i el integrador estan separats. Per tan es pot veure el comportament de tots els tipus de controladors.

Després de fer un estudi de les dues targetes de adquisició de dades disponibles en els laboratoris del Tecno Campus Mataró , s'ha decidit utilitzar la targeta PC-LAB per tal de poder interactuar amb el hardware directament a traves del Visual Basic 5.0. Ja que la targeta NI PCI-6014, per treballar amb Visual Basic 6.0 o Visual C++, els drivers són de pagament.

El model matemàtic del sistema esta compost de un LAG i un integrador.

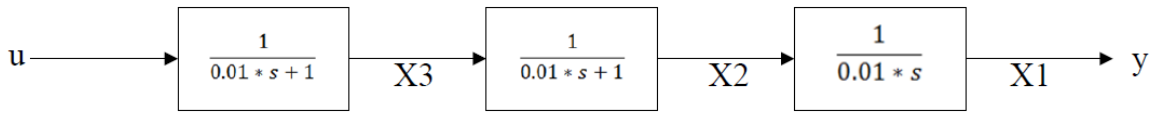


Fig. 5.2 Model LAG+LAG+INT amb tau de 10ms

Per més informació del sistema es pot consultar a la direcció: http://www.feedback-group.com/

5.2. Càlcul del model.

Partint del model anterior de la Fig. 5.2, es pot realitzar un graf per poder convertir aquest model en les matrius de variable d'estat. Primer s'ha t'arregla la funció de transferència per poder realitzar el graf.

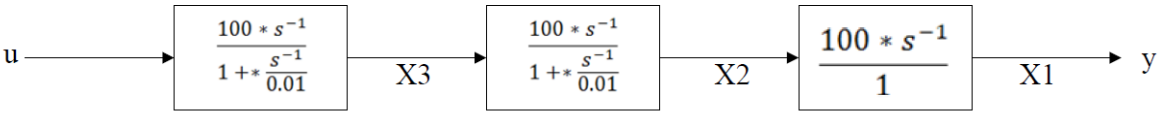


Fig. 5.3 Model LAG+LAG+INT preparat

Es procedeix a fer el graf (Fig. 5.4):

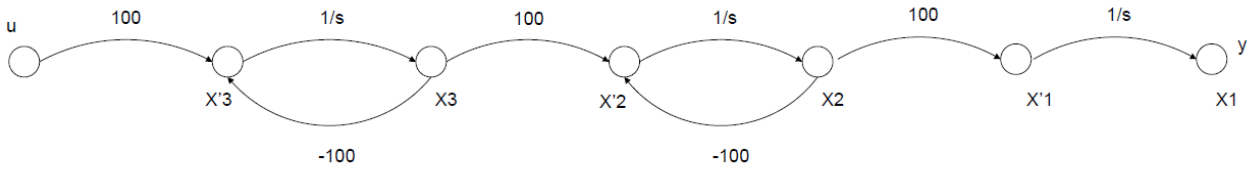


Fig. 5.4 Graf del sistema

De la Fig. 6.4 extreuen les següents equacions que formaran les matrius d'estat:

$$\dot{x_1} = 100 * x_2 \quad (5.1)$$

$$\dot{x_2} = -100 * x_2 + 100 * x_3 \quad (5.2)$$

$$\dot{x_3} = -100 * x_3 + 100 * u \quad (5.3)$$

Les matrius d'estat son:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} = \begin{bmatrix}0 & 100 & 0\\ 0 & -100 & 100\\ 0 & 0 & -100\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix} + \begin{bmatrix}0\\ 0\\ 100\end{bmatrix}u \tag{5.4}$$

$$y = [1 \quad 0 \quad 0]x + [0]x \tag{5.5}$$

$$A = \begin{bmatrix}0 & 100 & 0\\ 0 & -100 & 100\\ 0 & 0 & -100\end{bmatrix} \tag{5.6}$$

$$B = \begin{bmatrix}0\\ 0\\ 100\end{bmatrix} \tag{5.7}$$

$$C = [1 \quad 0 \quad 0] \tag{5.8}$$

$$D = [0] \tag{5.9}$$

Aquest sistema té una resposta molt rapida, ja que cada un dels LAG o integradors tenen una tau de 10ms, això fa que el sistema en conjunt estigui sobre una tau de 100ms. Per comprovar la tau del sistema es pot fer de diferents maneres. A traves de la formula o podem simular el sistema en llaç tancat i observar quan temps tarda en arribar el 63%.

Per determinar el temps de mostreig que s'ha de treballar es calcula a partir de la tau. Si la tau es de 100 ms s'escullira un valor que estigui per sota de 10 vegades menys el temps de la tau, es a dir si 10ms o menys. En aquest cas, s'ha triat treballar en un temps de mostreig de 5ms.

5.3. Simulació del model en llaç tancat.

Un cop obtingudes les matrius d'estat simularem el sistema amb el Matlab i el simulink. En continuo i discret per poder calcular els controladors amb discrets, ja que controlarem el sistema des de la targeta PC-LAB i la senyal que es pot generar es discreta. S'introdueix les matrius A,B,C,D en el Matlab:

```
>> a=[0 100 0;0 -100 100;0 0 -100];
>> b=[0;0;100] ;
>> c=[1 0 0] ;
>> d=[0] ;
>> sysc=ss(a,b,c,d) ;
sysd=c2d(sysc,0.005)
 a =       x1      x2      x3
   x1       1  0.3935  0.0902
   x2       0  0.6065  0.3033
   x3       0       0  0.6065
 b =       u1
```

```
   x1  0.01633
   x2   0.0902
   x3   0.3935
 c =   x1  x2  x3
   y1   1   0   0
 d =   u1
   y1   0
 Sampling time: 0.005
Discrete-time model.
>> [ad,bd,cd,dd]=ssdata(sysd)
ad =    1.0000    0.3935    0.0902
         0    0.6065    0.3033
         0         0    0.6065
bd =    0.0163
    0.0902
    0.3935
cd =     1     0     0
dd =     0
```

Es simula el sistema amb llaç tancat, per comprovar si la conversió ha variable d'estat esta ben fet:

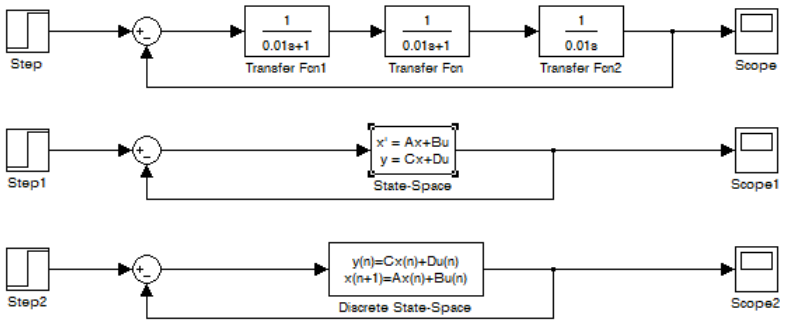
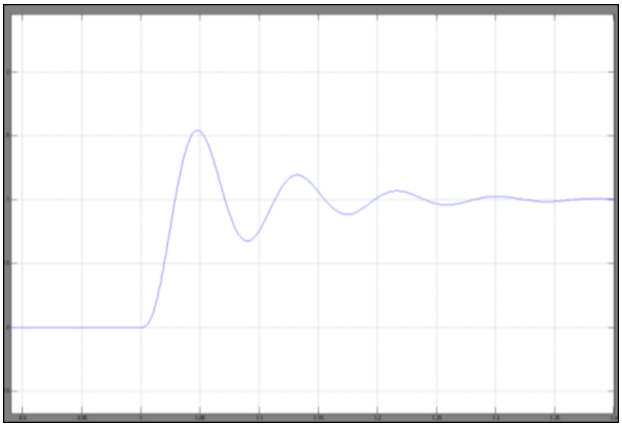


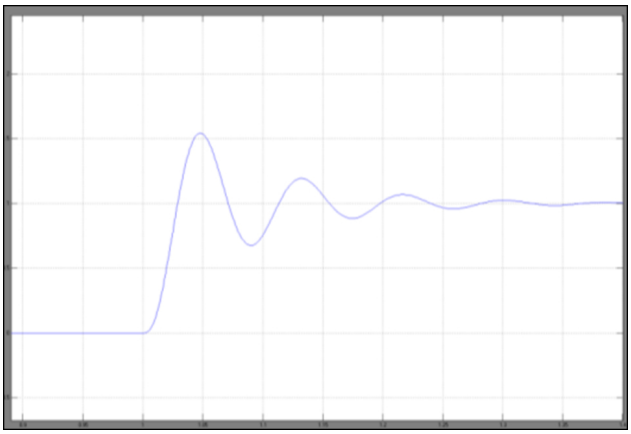
Fig. 5.5 Simulink del sistema amb llaç tancat



Llegenda:

.Blau: sortida del sistema

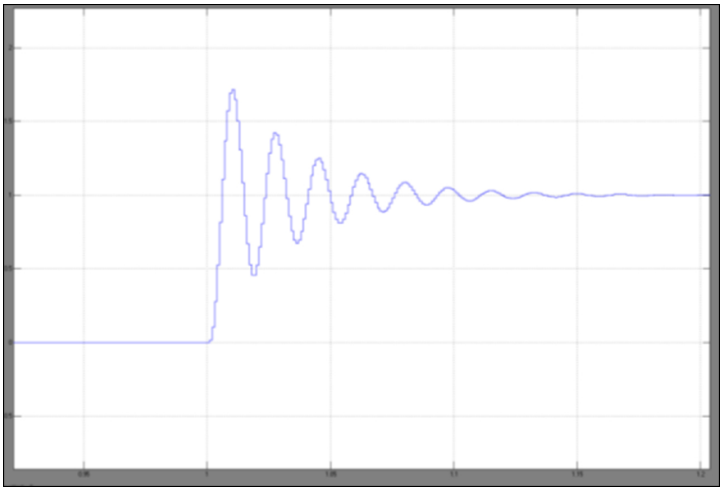
Fig. 5.6 Resposta del sistema original amb llaç tancat



Llegenda:

.Blau: sortida del sistema

Fig. 5.7 Resposta del sistema per variable d'estat amb llaç tancat



Llegenda:

Blau: sortida del sistema

Fig. 5.8 Resposta del sistema per variable d'estat discret amb llaç tancat (Tm=50ms)

5.4. Disseny del controlador per retorn d'estat.

El control per retorn d'estat tracte de controlar el sistema a traves de la observació de les variables d'estat. En aquest cas el sistema a controlar es de ordre 3 per tan les variables d'estat a observar son X1, X2 i X3. Per realitzar aquest control a traves de la targeta d'adquisició de dades PC-LAB, s'era necessari capturar les tres variables al mateix temps. Això no serà possible ja que la targeta nomes té un convertidor analògic digital. La solució proposada es capturar una variable i canviar el multiplexor de entrada per seleccionar el altre canal a capturar. Com per commutar un canal al altre es necessari un temps per que el multiplexor accioni.

La solució proposada es treballar tres cops més ràpid que el temps de mostreig per tal de entrar en la subrutina d'interrupció 3 vegades. La primera captura la senyal de X1 i en la segona captura la senyal de X2 i la tercera capturar la X3 i calcular l'acció de control, es a dir si el sistema esta calculat per un temps de mostreig de 5ms, es treballa 1.6667 ms. Això fa que una mostra tingui un retard de 1.6667 ms i l'altre un retard de 3.3334 ms respecte la primera.

5.4.1. Localització de pols.

El disseny de un controlador per retorn d'estat consisteix en buscar uns pols que multiplicats per cada una de les variables d'estat X1,X2,X3 es poguí controlar el sistema i reduir el seu temps de resposta. Ja que el sistema té una constant de temps uns 100ms multiplicarem per 100 els pols. Es simularà el sistema amb uns pols de BESSEL.

- Càlcul de pols BESSEL:

Pols de Bessel : 0.942 0.7455+j*0.7112 0.7455-j*0.7112

Pols Bessel continus per 100:

```
Pc=[-0.942*100 (-0.7455+j*0.7112)*100 (-0.7455-j*0.7112)*100]
Pc = -94.2000            -74.5500 +71.1200i           -74.5500 -71.1200i
```

5.4.2. Convertir els pols continuos a pols discrets.

- Discretitzem els pols BESSEL:

```
Pd=[exp(Pc(1,1)*0.005) exp(Pc(1,2)*0.005) exp(Pc(1,3)*0.005)]
Pd =   0.6244             0.6457 + 0.2398i   0.6457 - 0.2398i
```

5.4.3. Simulació del sistema en llaç tancat controlat per retorn d'estat.

Calcul de la matrius K BESSEL:

```
k=acker(ad,bd,Pd)
k =    0.8881    0.9105    0.5097
```

- Simulació del sistema amb pols BESSEL:

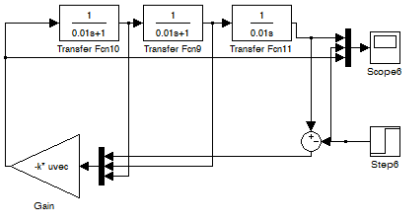
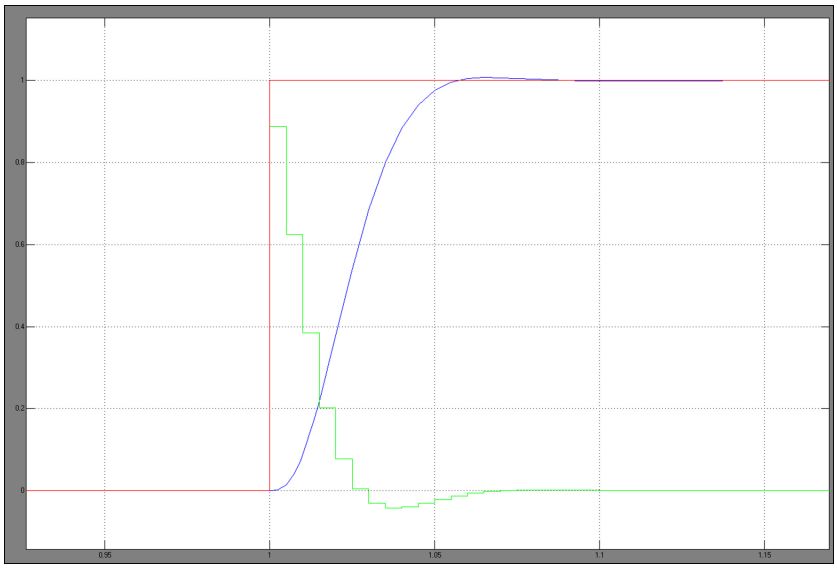


Fig. 5.9 Simulació feedback amb controlador per retorn d'estat



Llegenda:

Verd: Acció de controls

Lila: Sortida del sistema

Vermell:Entrada del sistema "STEP"

Fig. 5.10 Resposta del feedback amb controlador per retorn d'estat (BESSEL)

5.4.4. Programació per visual basic del programa principal:

En aquest capítol s'explicarà com crear una aplicació per controlar el sistema del motor a traves de un controlador per retorn d'estat amb la targeta PC-LAB. En primer lloc s'instal·la els drivers de la targeta PC-LAB tal i com s'ha explicat en el capítol 3.2.3. de aquest mateix document. Un cop estan instal·lats els drivers, es crea un nou projecte de visual basic 5.0.

El programa constarà de dues parts, la part del formulari on es manipula diferents paràmetres del nostra controlador, i la part de les interrupcions on es realitza el control pas per pas. En primer lloc s'observa en quin port IRQ esta instal·lada la targeta PC-LAB, en aquest cas serà la IRQ 7.

La funció principal de qualsevol aplicació de visual basic és el formulari. Es crea una funció anomenada form_load() que serveix per carregar el entorn de usuari, es a dir el interfase gràfic.

```
Private Sub form_load()   ' Creació de la funció
Out &H229, &H8 ' Guany unitari
Out &H22B, &H6 ' Trigger per timer i EOC per interrupció
inter.Text = 1.666667   ' Selecció del temps de mosteig en ms
U = 0 ' Valor desitjat a la sortida
U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas de escala del
digital / analògic
balt = U \ 256 ' Mascara per agafar el byte baix
bbaix = U - balt * 256 ' Mascara per agafar el byte alt
Call Out(&H224, bbaix) ' Escriu en el port de sortida el valor del byte
baix
Call Out(&H225, balt) ' Escriu en el port de sortida el valor del byte
alt
HW32 = OpenTVicHW32(HW32, "TVICHW32", "TVicDevice0") ' Activació de la
IRQ 7
End Sub ' Final de la funció
```

En la funció anterior s'activen les interrupcions, es posa a 0 v la sortida analògica, es selecció el guany del convertidor digital analògic i s'activa les interrupcions quan el convertidor analògic digital acaba de convertir un valor.

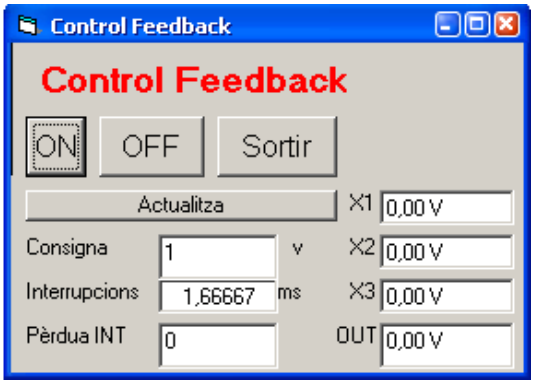


Fig. 5.11 Interfase gràfic control motor per retorn d'estat

En la Fig. 6.11 es veu el entorn gràfic per poder controlar el sistema. En aquesta aplicació s'han implementat altres dades comparat amb el control del motor, ja que el sistema és més ràpid, s'ha implementat un control de interrupció per poder comprovar que el conjunt del ordinador i la targeta treballant en temps real i no es perden dades en el control. Es poden observar diferents botons:

ON : Encendre el controlador, comença a controlar segons la posició introduïda.

OFF: El controlador para de controlar.

Sortir: Surt de l'aplicació.

Posició: introduir el angle que es vol posicionar el motor.

Interrupcions : Es selecciona el temps de mostreig que es treballa. (no es aconsellable canviar ja que els càlculs del pols discrets esta fet per aquest temps de mostreig.)

OUT: Mostra els valors de la acció de control en Vols.

X1,2,3: Mostra els valors en vols del valor que tenen en cada moment.

Pèrdua INT: Mostra si en algun moment es perden interrupcions

Explicació de cada boto i funcions en el programa principal:

- Boto ON: Esta lligat a la funció Adquirir_Click

```
Private Sub Adquirir_Click() ' Declaració de la funció
xo = 1 ' Posa el comptador del canal a capturar a 1.
Out &H22A, &H1 ' Selecció el canal 1 de entrada
Vref = con.Text' Captura el valor de consigna del camp de text.
E = 0 ' Posa a 0 la variable de entrada
U = 0 ' Posa a 0 la variable de sortida
Call UnmaskIRQ(HW32, 7, AddressOf RutinaServei) ' Vincula la interrupció
de IRQ 7 a la rutina d'interrupcions.
Out &H228, &H0 'Activació de la interrupció de la tarja.
Timer1.Enabled = True ' Activa el timer de refresc de pantalla.
Comp = 0 ' Es col·loca a 0 el comptador de les interrupcions
Call ProgramaTimer ' Activar i programa el timer de la tarja
End Sub ' Final de la funció
```

En la funció anterior hi ha una variable anomenada "xo" que es posa a 1, això es perquè al programar un controlador per retorn d'estat, es necessari capturar les tres variables d'estat X1, X2 i X3, però la targeta nomes té un convertidor analògic digital i el multiplexor té un temps de commutació. Per tan la solució proposada es treballar amb una temps de mostreig tres cops més ràpid i cada cop que s'entra a la funció d'interrupció es captura una de les variables d'estat i en la tercera captura poder treure el valor de sortida del control. Això fa que la variable X2 i X3 estigui retardada 1.66667 ms la variable X2 i la variable X3 3.33334 ms respecte la variable X1. Aquesta petita diferencia fa que en la resposta no sigui exactament igual que en la simulació del Matlab.

S'ha utilitzat una funció anomenada "ProgramaTimer" que consisteix en escriure els valors de configuració del timer de la targeta. Passem analitzar les següents funcions.

- **Funció programació Timer:**

```
Private Sub ProgramaTimer() ' Declaració funció
t1_H = inter.Text \ 256 ' Captura de la part alta del camp de text
d'interrupcions.
t1_L = inter.Text - t1_H * 256 ' Captura de la part baixa del camp de
text d'interrupcions.
t2_H = 7          ' Programació del segon timer amb un valor prefixat
t2_L = 208      ' de 1 ms com a unitat de temps
Out &H223, &H74 ' Activar el primer timer
Out &H221, t1_L ' Programa byte baix
Out &H221, t1_H ' Programa byte alt
Out &H223, &HB4 ' Activar el segon timer
Out &H222, t2_L ' Programa byte baix
Out &H222, t2_H  ' Programa byte alt
End Sub  ' Final de la funció
```

La targeta esta dotada de 2 temporitzadors de 16 bits. Que es utilitzat per activa la interrupció de la targeta. En la funció anterior esta programat el timer de manera que es selecciona el temps a traves de un camp de text anomenat inter.text.

El primer Timer es amb un temps variable on es configura a traves del camp de text "inter.Text" amb les unitats de milisegons. El segon Timer esta amb un valor fixa que dona la unitats de milisegons. Posteriorment s'escriu a la adreces corresponents els valors dels Timers i s'activen. On el primer timer s'activa per la direcció "&H74" i el segon per "&HB4".En cop s'ha esta executant el controlador hi ha una funció que s'executa regularment per refrescar els valors mostrats en pantalla. Aquesta funció depèn de un timer anomenat: "Timer1_Timer".

- Funció de actualització de pantalla:

```
Private Sub Timer1_Timer() ' Declaració de la funció
comp_int.Text = Comp ' Actualitzar les vegades que s'ha executat una
rutina d'interrupció.
X1T.Text = Format(X1, "###0.00") + " V " ' Mostra la variable X1
X2T.Text = Format(X2, "###0.00") + " V " ' Mostra la variable X2
X3T.Text = Format(X3, "###0.00") + " V " ' Mostra la variable X3
OUTD.Text = Format(Ureal, "###0.00") + " V " ' Mostra per pantalla el
valor de sortida
End Sub ' Fi de la funció
```

- Funció de actualitzar consigna:

```
Private Sub actualitza_Click()' Declaració de la funció
ref = con.Text ' Capura del camp de text el valor de consigna
Call ConAngleVols ' Conversió de angle a vols
End Sub ' Fi de la funció
```

-Funció OFF "Aturar"

```
Private Sub Atura_Click()' Declaració de la funció
Call Out(&H20B, &H0)   ' Desactiva les interrupcions del convertidor
Call Out(&H223, &H74)  ' Desactiva el timer 1
Call Out(&H221, &H0)  ' Desactiva escriu 0 al timer 1
Call Out(&H223, &HB4)  ' Desactiva el timer 2
Call Out(&H222, &H0)  ' Desactiva escriu 0 al timer 2
Out &H22A, &H1 ' Selecciona l'entrada 1 del convertidor digital/analògic
Call MaskIRQ(HW32, 7)  ' Desactiva les interrupcions IRQ 7
End Sub ' Fi de la funció
```

-Funció Sortir:

```
Private Sub Command1_Click() ' Declaració de la funció
'9 SORTIR tancat el Form
Unload Form1 'descarrega el form i tanca el programa
End Sub ' Fi de la funció
```

Ens programa principal s'ha vist totes les funcions que s'utilitzen, per poder fer anar totes aquests funcions es necessari declarar les següents variables:

```
Dim HW32 As Long ' IRQ
Dim t1_H As Byte   ' Timer 1 byte alt
Dim t1_L As Byte   ' Timer 1 byte baix
Dim t2_H As Byte ' Timer 2 byte alt
Dim t2_L As Byte   ' Timer 2 byte baix
```

5.4.5. Programació per Visual basic de la rutina d'interrupció

La rutina d'interrupció, es la rutina on esta implementada tot el control del sistema. Aquesta funció esta ubicada en un mòdul extern que em creat a dins del projecte anomenat "Modules". Esta formada per una sola funció anomenada Rutina de Servei.

En primer lloc s'implementa el control de interrupcions que determina si executen correctament totes les interrupcions, es tracte de activar de nou les interrupcions i sumar 1 a una variable anomenada "Comp", que final de la funció d'interrupció es restarà 1. Això fa que si en alguna interrupció el comptador varia de 0 vol dir que s'ha interrompi't la rutina per una altre interrupció abans que acabi la primera. Posteriorment s'utilitza unes variables K1, K2 i K3 on introduïm els valors de la matriu de control calculada anteriorment en el punt 6.4.3. Després s'utilitza el comptador de interrupcions per saber quants cops s'ha executat aquesta funció. A partir de aquí entra la part de control, amb una sèrie de condicions "If else" que determinant quina de les variables d'estat és necessari capturar.

En el punt de tenir que captura la variable X1, primer canviar el indicador de la pròxima variable d'estat (xo) que s'ha de capturar. Captura el valor de la entrada analògica i es

canviar el canal al per el canal 2 on resideix la senyal X2. Posteriorment fa exactament el mateix amb X3, i en el moment que té X3 calcula la sortida.

Es converteix el valor de la sortida en pas del digital analògic i s'escriu en la adreça del convertidor.

- **Funció d'interrupció:**

```
Public lectura As Double ' Variable de tipus Double per capturar el valor
del A/D
Public U As Double ' Variable de tipus Double on es guarda la sortida
Public xo As Integer ' Variable de tipus Integer per indicar quin valor
Public K1 As Double ' Variable de tipus Double de la matriu de control
Public K2 As Double ' Variable de tipus Double de la matriu de control
Public K3 As Double ' Variable de tipus Double de la matriu de control
Public X1 As Double ' Variable de tipus Double de la variable d'estat X1
Public X2 As Double ' Variable de tipus Double de la variable d'estat X2
Public X3 As Double ' Variable de tipus Double de la variable d'estat X3
Public Vref As Double ' Variable de tipus Double de la consigna
Public Ureal As Double ' Variable de tipus Double de la sortida
Public Comp As Double ' Variable de tipus Double del comptador
d'interrupcions
Public Sub RutinaServei(ByVal x As Integer) ' Declaració de la funció
    Out &H228, &H0 ' Activa les interrupcions
    Comp = Comp + 1 ' Suma 1 a la variable Comp
    K1 = -0.8881 ' Valor de K BESSEL
    K2 = -0.9105 ' Valor de K BESSEL
    K3 = -0.5097 ' Valor de K BESSEL
    If (xo = 1) Then ' Condició de la variable a capturar
       xo = 2 ' Canvi del indicador de capturar
       balt = Inp(&H225) And &HF ' Captura valor byte baix del analògic
digital
       bbaix = Inp(&H224) And &HFF ' Captura valor byte alt del analògic
digital
       lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un
double
       X1 = ((lectura * 20#) / 4095) - 10 ' Conversió a vols de X1
       Out &H22A, &H2    ' Seleccionar de entrada canal 2
    ElseIf ( xo = 2 ) Then ' Entra si xo es 2
        xo = 3 ' Canvi del indicador de capturar
        balt = Inp(&H225) And &HF ' Captura valor byte baix del analògic
digital
        bbaix = Inp(&H224) And &HFF ' Captura valor byte alt del analògic
digital
       lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un
double
       X2 = ((lectura * 20#) / 4095) - 10 ' Conversió a vols de X2
       Out &H22A, &H3 ' Seleccionar canal 3
    ElseIf (xo = 3) Then
        xo = 1 ' Canvi del indicador de capturar
        balt = Inp(&H225) And &HF ' Captura valor byte baix del analògic
digital
        bbaix = Inp(&H224) And &HFF ' Captura valor byte alt del analògic
digital
       lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un
double
       X2 = ((lectura * 20#) / 4095) - 10 ' Conversió a vols de X3
```

```
       Out &H22A, &H3 ' Seleccionar canal 3
        ' Càlcul del control:
        X1r = -Vref - X1 ' Resta consigna menys posició
        Ureal = K1 * X1r + K2 * X2 + K3 * -X3 ' Càlcul controlador
        U = Ureal ' Canvi de variable el valor.
        If (U > 9.9) Then ' Limits de 10 v
             U = 9.9
        End If
        If (U < -9.9) Then
             U = -9.9
        End If
        U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas del
convertidor D/A

        balt = U \ 256  ' Separa byte alt
        bbaix = U - balt * 256 ' Separar byte baix
        Call Out(&H224, bbaix) ' Escriu el byte baix al convertidor
analògic digital
        Call Out(&H225, balt) ' Escriu el byte alt al convertidor
analògic digital
        Comp = Comp - 1 ' Resta 1 a la variable Comp
    End If
End Sub
```

En la funció anterior s'ha calculat el control amb alguns signes canviats això es perquè el sistema a cada LAG o integrador té un canvi de signe no que s'havia considerat anteriorment. Per tan la variable X3 després del primer LAG es negada, la X2 sortida del segon LAG es positiva ja que esta negada dues vegades, i per acabar la sortida que es el mateix que la variable X1 torna esta negada.

5.4.6. Posta en marxa del sistema.

Per posar en marxa el sistema compilem la aplicació i connectem (Fig. 5.12):

Primer és necessari connectar a la xarxa de 230v el Feedback. Ja que el sistema té els LAG's i els integradors per separat, es necessari cablejar el sistema tal i com es mostra el següent esquema (Fig. 5.12)

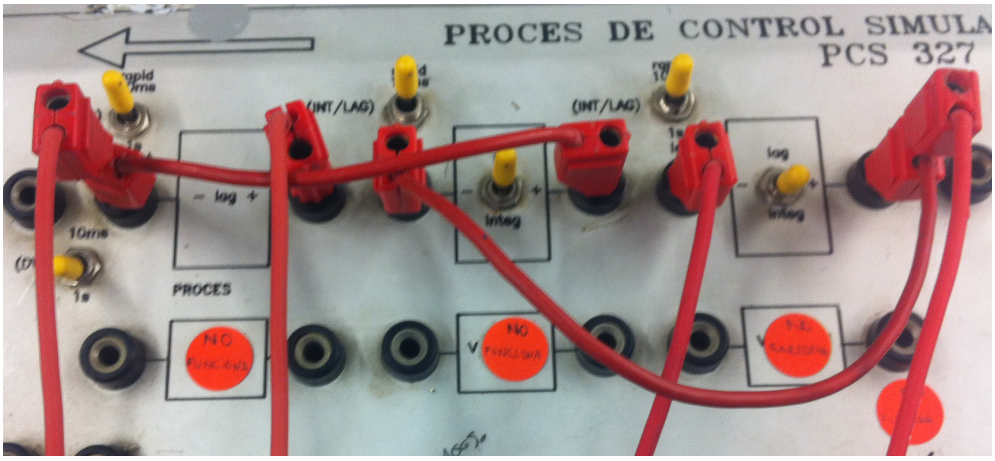
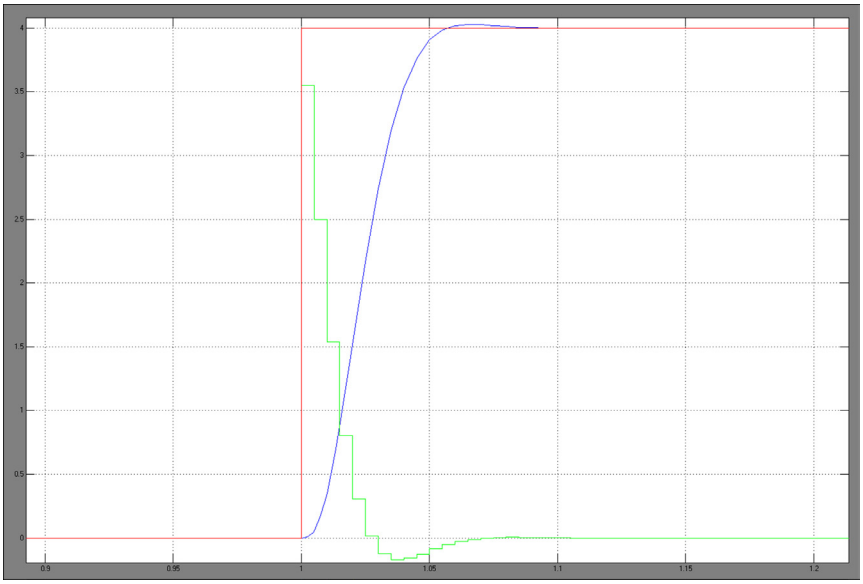


Fig. 5.12 Connexió Feedback

5.4.7 Comprovació dels resultats obtinguts.

La comprovació s'ha fet a traves de diferents mesures a diferents posicions comparant amb la corba obtinguda desde el Matlab.

STEP de 0v a 4v BESSEL:



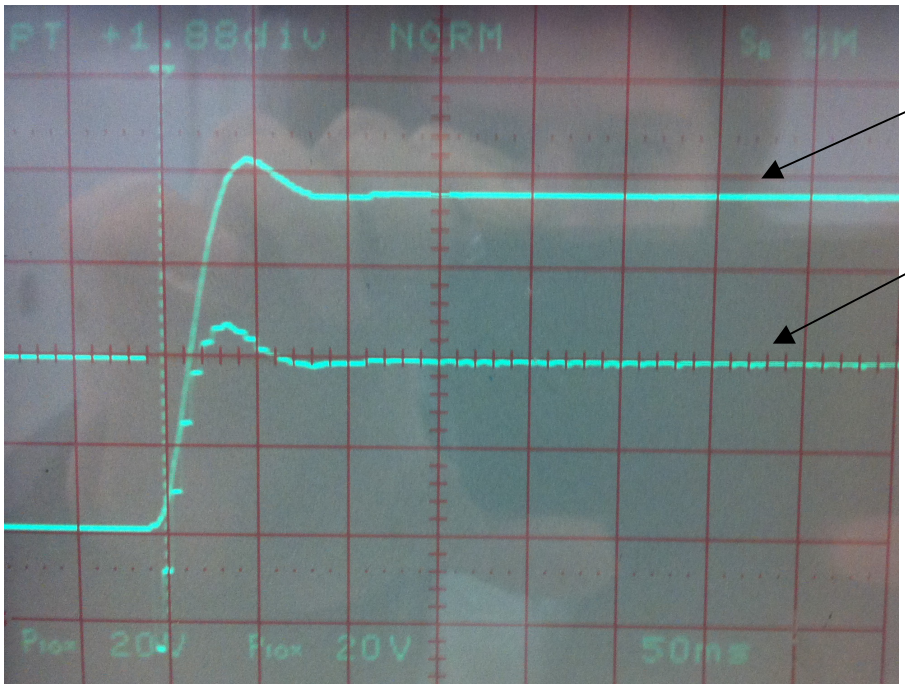
Llegenda:

Lila: Sortida del sistema

Vermell: Entrada del sistema "STEP"

Verd: Acció de control

Fig. 5.13 Simulació Feedback per retorn d'estat amb pols BESSEL de 0v a 4v



Llegenda:

Canal 1 : 2 v/div

Senyal de sortida

Canal 2 : 2 v/div

Acció de control

Temps: 20 ms/div

Fig. 5.14 Sistema real per retorn d'estat amb pols BESSEL de 0v a 4v

Comparant els dos valors, els dos s'estabilitzen a 4v i amb un temps de 60 ms.

En la simulació del control per retorn d'estat es veu un petit sobre impuls degut a que les variables d'estat X2 i X3 tenen un retard respecte la variable X1.

5.5 Disseny del controlador per retorn d'estat amb observador

5.5.1.Funcionament de l'observador predictiu

El observador predictiu tal com esta explicat en el capítol 3, funció de manera que captura les variable d'estat en un temps de mostreig inferior al temps que surt la sortida això fa que el càlcul del control sigui mes ràpid. Aquests tipus de observadors es possible simular amb el Matlab.

5.5.2.Funcionament del observador corrent

A diferencia del observador predictiu, el observador corrent no es possible simular a traves del Matlab. Es un controlador utilitzat per ordinadors amb un temps de càlcul molt ràpid o temps de mostreig.

El Observador corrent, funciona de manera que es calcula en el instant k el valor que surt. Sempre esta en mateix instant es fa tot. Aquest procediment es el més lògic de funcionament però si el ordinador on s'està implementant el temps de càlcul es llarg, la sortida que surt té un retard.

5.5.3.Càlcul de pols del observador

El càlcul de un observador consisteix en col·locar un pols més allunyats dels pols del controlador. Aprofitant els pols calculat anteriorment en el apartat 5.4.2

- Pols BESSEL:

```
Pd =   0.6244             0.6457 + 0.2398i   0.6457 - 0.2398i
```

Càlcul del pols del Observador BESSEL:

```
>> Pod=Pd/1.5
Pod =   0.4163             0.4305 + 0.1599i   0.4305 - 0.1599i
```

Càlcul de la Matriu L a traves de Ackerman BESSEL:

```
>> L=acker(ad',cd',Pod)'
L =     0.9358
        0.2933
        0.0902
```

5.5.4 Creació del controlador:

```
acon=ad-bd*k-L*cd
bcon=L
ccon=k
dcon=0
```

Controlador resultant per un sistema amb observador i pols BESSEL:

```
acon =    0.0497    0.3786    0.0819
         -0.3734    0.5244    0.2573
         -0.4396   -0.3583    0.4060
bcon =    0.9358
          0.2933
          0.0902
ccon =    0.8881    0.9105    0.5097
dcon =     0
```

5.5.5 Simulació del sistema en llaç tancat controlat per retorn d'estat amb observador.

- Simulació del sistema a traves del Simulink:

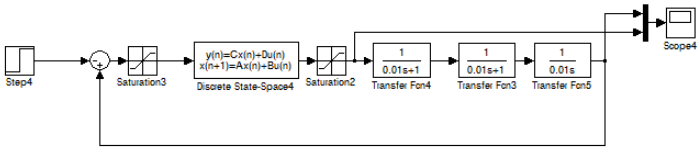
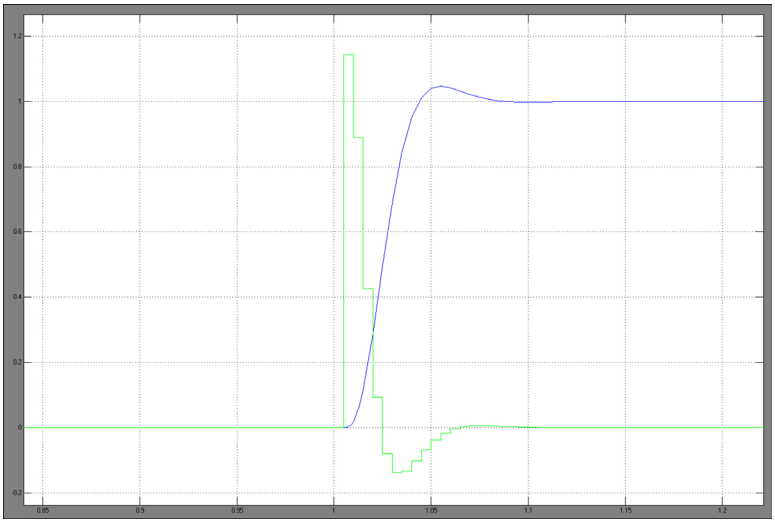


Fig. 5.15 Simulació sistema amb llaç tancat amb observador



Llegenda:

Lila: Sortida del sistema.

Groc: Acció de control.

Fig. 5.16 Resposta del motor amb observador (BESSEL)

5.5.6. Programació per visual basic del programa principal:

En aquest capítol s'explicarà com crear una aplicació per controlar el sistema del feedback a traves de un controlador per retorn d'estat amb observador. El funcionament del programa es casi igual que el explicat en el punt 5.4.4. En aquest capítol és mencionant els canvis realitzats respecte el programa del control per retorn d'estat sense observador (Punt 5.4.4).

En les funcions que té el programa principal no té cap diferencia excepte en la funció de "form_load". Que la principal diferencia es que anteriorment el temps de mostreig ara de 1.66667 ms ja que teníem que capturar les tres variables d'estat, però en aquest moment nomes necessitem capturar la sortida del sistema per poder fer la realimentació. Per tant el temps de mostreig serà de 5ms, és a dir que la variable "inter.Text" es igual a 5.

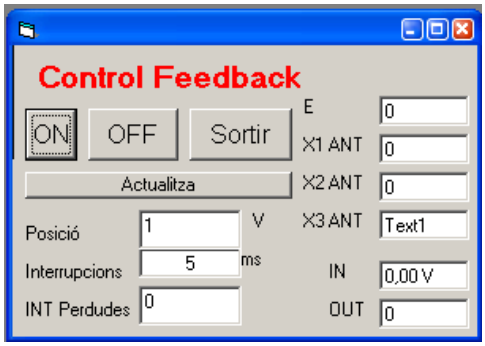


Fig. 5.17 Entorn gràfic control Feedback amb observador

En la Fig. 5.17 es veu el entorn gràfic per poder controlar el sistema. Es poden observar diferents botons:

ON : Encendre el controlador, comença a controlar segons la posició introduïda.

OFF: El controlador para de controlar.

Sortir: Surt de l'aplicació.

Posició: introduir el angle que es vol posicionar el motor.

Interrupcions : Es selecciona el temps de mostreig que es treballa. (no es aconsellable canviar ja que els càlculs del pols discrets esta fet per aquest temps de mostreig.)

IN: Mostra el valor en Vols de la entrada (Així es pot comprovar que s'estabilitza al valor desitjat.)

OUT: Mostra els valors de la acció de control en Vols.

Valor E: La entrada menys la consigna.

Valor EANT: Valor de la sortida en vols de un mostra anterior.

Valor E2ANT: Valor de la sortida en vols de dues mostres anteriors.

- Boto ON: Esta lligat a la funció Adquirir_Click

En aquesta funció s'ha eliminat la línea de codi de "xo" ja que no es necessària perquè en un sistema amb observador nomes es captura la realimentació que és la variable d'estat X1 (posició). Per altre banda es necessari unes variables que son EANT, E2ANT i E3ANT que

serveixen per saber quin valor tenien les variables d'estat en mostres anteriors. Gracies aquestes variables podem observar les variables d'estat de la següent mostra.

```
Private Sub Adquirir_Click() ' Declaració de la funció
Out &H22A, &H1 ' Selecció el canal 1 de entrada
ref = con.Text ' Captura el valor de consigna del camp de text.
Call ConAngleVols ' Converteix el valor de angle a vols i deixa el valor
a Vref
E = 0 'Posa a 0 la variable de entrada
U = 0 'Posa a 0 la variable de sortida
EANT = 0 'Posar a 0 la variable de X1 anterior
E2ANT = 0 'Posar a 0 la variable de X2 anterior
Call UnmaskIRQ(HW32, 7, AddressOf RutinaServei) ' Vincula la interrupció
de IRQ 7 a la rutina d'interrupcions.
Out &H228, &H0 'Activació de la interrupció de la tarja.
Timer1.Enabled = True ' Activa el timer de refresc de pantalla.
Comp = 0 ' Es col·loca a 0 el comptador de les interrupcions
Call ProgramaTimer ' Activar i programa el timer de la tarja
End Sub ' Final de la funció
```

5.5.7. Programació per visual basic de la rutina d'interrupció

La rutina d'interrupció, es la rutina on esta implementada tot el control del sistema. Aquesta funció esta ubicada en un mòdul extern que em creat a dins del projecte anomenat "Modules". Esta formada per una sola funció anomenada Rutina de Servei.

En primer lloc s'analitzarà el control per un observador predictiu, aquests tipus de observadors són els més utilitzats, ja que van be per ordinadors lents o sistemes molts ràpids. En aquest cas en la mateix execució es calcula i surt el valor de control.

Es converteix el valor de la sortida en pas del digital analògic i s'escriu en la adreça del convertidor.

- **Funció d'interrupció:**

```
Public lectura As Double ' Variable de tipus double per capturar el valor
del A/D
Public valor As Double ' Variable de tipus double, valor de la entrada en
Vols
Public U As Double ' Variable de tipus double on es guarda la sortida
Public EANT As Double 'Variable de tipus double la variable de X1
anterior
Public E2ANT As Double   'Variable de tipus double la variable de X2
anterior
Public E3ANT As Double   'Variable de tipus double la variable de X2
anterior
Public Xt1 As Double 'Variable de tipus double on es guarda l'aproximació
de X1
Public Xt2 As Double 'Variable de tipus double on es guarda l'aproximació
de X2
```

```
Public Xt3 As Double 'Variable de tipus double on es guarda l'aproximació
de X3
Public Ae1 As Double 'Variable de tipus double, de la multiplicació de A
per sortides anteriors
Public Ae2 As Double 'Variable de tipus double, de la multiplicació de A
per sortides anteriors
Public Ae3 As Double 'Variable de tipus double, de la multiplicació de A
per sortides anteriors
Public Ly1 As Double 'Variable de tipus double on es la matriu L
multiplicada per la entrada menys la consigna.
Public Ly2 As Double 'Variable de tipus double on es la matriu L
multiplicada per la entrada menys la consigna.
Public Ly3 As Double 'Variable de tipus double on es la matriu L
multiplicada per la entrada menys la consigna.
Public Vref As Double 'Variable de tipus Double de la consigna
Public E As Double 'Variable de tipus double de la entrada menys la
consigna
Public Ureal As Double 'Variable de tipus Double de la sortida
Public Comp As Double 'Variable de tipus Double del comptador
d'interrupcions
Public Sub RutinaServei (ByVal x As Integer) 'Declaració de la funció
    Comp = Comp + 1 'Compta cops que entra a la funció d'interrupcions
    balt = Inp(&H225) And &HF 'Captura valor byte baix del analògic
digital
    bbaix = Inp(&H224) And &HFF 'Captura valor byte alt del analògic
digital
    lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un double
    valor = ((lectura * 20#) / 4095) - 10 ' Conversió a vols
    E = Vref - valor ' Realimentació/ sortida del sistema menys la
consigna.
    Ae1 = 0.0497 * EANT + 0.3786 * E2ANT + 0.0819 * E3ANT ' Multiplicació
de la
    Ae2 = -0.3734 * EANT + 0.5244 * E2ANT + 0.2573 * E3ANT'matriu a per
les
    Ae3 = -0.4396 * EANT + -0.3583 * E2ANT + 0.406 * E3ANT'variables
d'estat de una mostra anterior
    Ly1 = 0.9358 * E ' Multiplicació dels observador per la entrada del
controlador
    Ly2 = 0.2933 * E ' Multiplicació dels observador per la entrada del
controlador
    Ly3 = 0.0902 * E ' Multiplicació dels observador per la entrada del
controlador
    Xt1 = Ae1 + Ly1 ' Aproximació de X1
    Xt2 = Ae2 + Ly2 ' Aproximació de X2
    Xt3 = Ae3 + Ly3 ' Aproximació de X3
    Ureal = 0.8881 * Xt1 + 0.9105 * Xt2 + 0.5097 * Xt3 ' Calcul del
controlador
    EANT = Xt1 ' Guarda la variable d'estat X1
    E2ANT = Xt2 ' Guarda la variable d'estat X2
    E3ANT = Xt3  ' Guarda la variable d'estat X3
    U = Ureal ' Canvi de variable el valor.
   If (U > 9.9) Then ' Limits de ±10 v
             U = 9.9
   End If
   If (U < -9.9) Then
             U = -9.9
   End If
   U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas del convertidor
D/A
   balt = U \ 256  ' Separa byte alt
```

```
   bbaix = U - balt * 256 ' Separar byte baix
   Call Out(&H224, bbaix) ' Escriu el byte baix al convertidor analògic
digital
   Call Out(&H225, balt) ' Escriu el byte alt al convertidor analògic
digital
   Out &H228, &H0 ' Activar les interrupcions
End Sub
```

5.5.8. Programació per visual basic de la rutina d'interrupcions amb observador corrent.

Tal i com s'ha descrit a la part de teoria del observador corrent. Primer es calcula les tres variables d'estat del nostre sistema, i es calcula la senyal de sortida.

Posteriorment fem el càlcul amb els valors de la sortida de aquest instant de temps poder calcular com varien les variables d'estat en el proper instant de temps.

- **Funció d'interrupció:**

```
Public lectura As Double ' Variable de tipus double per capturar el valor
del A/D
Public valor As Double ' Variable de tipus double, valor de la entrada en
Vols
Public U As Double ' Variable de tipus double on es guarda la sortida
Public Xp1 As Double ' Variable de tipus double on es guarda
l'aproximació de X1
Public Xp2 As Double ' Variable de tipus double on es guarda
l'aproximació de X2
Public Xp3 As Double ' Variable de tipus double on es guarda
l'aproximació de X3
Public X1 As Double ' Variable de tipus double on es guarda l'aproximació
de X1
Public X2 As Double ' Variable de tipus double on es guarda l'aproximació
de X2
Public X3 As Double ' Variable de tipus double on es guarda l'aproximació
de X2
Public Ae1 As Double ' Variable de tipus double, de la multiplicació de A
per sortides anteriors
Public Ae2 As Double ' Variable de tipus double, de la multiplicació de A
per sortides anteriors
Public Bu1 As Double ' Variable de tipus double on es la matriu L
multiplicada per la sortida.
Public Bu2 As Double ' Variable de tipus double on es la matriu L
multiplicada per la sortida.
Public Vref As Double ' Variable de tipus Double de la consigna
Public E As Double ' Variable de tipus double de la entrada menys la
consigna
Public Ureal As Double ' Variable de tipus Double de la sortida
Public Comp As Double ' Variable de tipus Double del comptador
d'interrupcions
Public Sub RutinaServei(ByVal x As Integer) ' Declaració de la funció
    Comp = Comp + 1 ' Compta cops que entra a la funció d'interrupcions
    balt = Inp(&H225) And &HF ' Captura valor byte baix del analògic
digital
```

```
    bbaix = Inp(&H224) And &HFF ' Captura valor byte alt del analògic
digital
    lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un double
    valor = ((lectura * 20#) / 4095) - 10 ' Conversió a vols
    E = Vref - valor ' Realimentació/ sortida del sistema menys la
consigna.
   X1 = Xp1 + 0.9358 * (E - 1 * Xp1) ' Càlcul de X1 a traves del
Observador i X1 anterior
   X2 = Xp2 + 0.2933 * (E - 1 * Xp1) ' Càlcul de X2 a traves del
Observador i X2 anterior
   X3 = Xp3 + 0.0902 * (E - 1 * Xp3) ' Càlcul de X3 a traves del
Observador i X1 anterior

    Ureal = 0.8881 * X1 + 0.9105 * X2 + 0.5097 * X3 ' Càlcul de la
sortida
    Ae1 = 1 * X1 + 0.3935 * X2 + 0.0902 * X3 ' Matriu A per les variables
d'estat
    Ae2 = 0 * X1 + 0.6065 * X2 + 0.3033 * X3 ' Matriu A per les variables
d'estat
    Ae3 = 0 * X1 + 0 * X2 + 0.6065 * X3 ' Matriu A per les variables
d'estat
    Bu1 = 0.0163* Ureal ' Càlcul de "B" o "L" per la sortida
    Bu2 = 0.0902 * Ureal ' Càlcul de "B" o "L" per la sortida
    Bu3 = 0.3935 * Ureal ' Càlcul de "B" o "L" per la sortida
    Xp1 = Ae1 + Bu1 ' Càlcul X1 en la mostra anterior
    Xp2 = Ae2 + Bu2 ' Càlcul X2 en la mostra anterior
    Xp3 = Ae3 + Bu3 ' Càlcul X3 en la mostra anterior
    U = Ureal ' Canvi de variable el valor.
   If (U > 9.9) Then ' Limits de ±10 v
             U = 9.9
   End If
   If (U < -9.9) Then
             U = -9.9
   End If
   U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas del convertidor
D/A
   balt = U \ 256  ' Separa byte alt
   bbaix = U - balt * 256 ' Separar byte baix
   Call Out(&H224, bbaix) ' Escriu el byte baix al convertidor analògic
digital
   Call Out(&H225, balt) ' Escriu el byte alt al convertidor analògic
digital
   Out &H228, &H0 ' Activar les interrupcions
End Sub
```

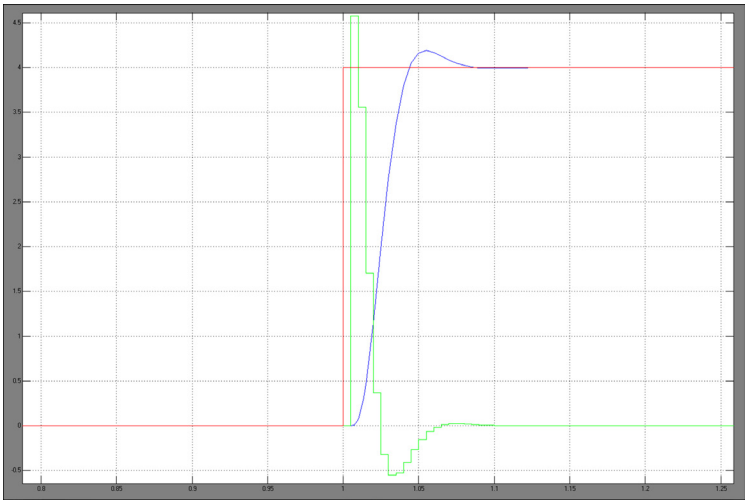
5.5.9. Posta en marxa del sistema.

Per posar en marxa el sistema compilem la aplicació i connectem (Fig. 5.18). Primer és necessari connectar a la xarxa de 230v el Feedback. Ja que el sistema té els LAG's i els integradors per separat, es necessari cablejar el sistema tal i com es mostra el següent esquema (Fig. 5.18)

Fig. 5.18 Connexió Feedback

5.5.10 Comprovació dels resultats obtinguts observador predictiu.

STEP de 0v a 4v BESSEL:



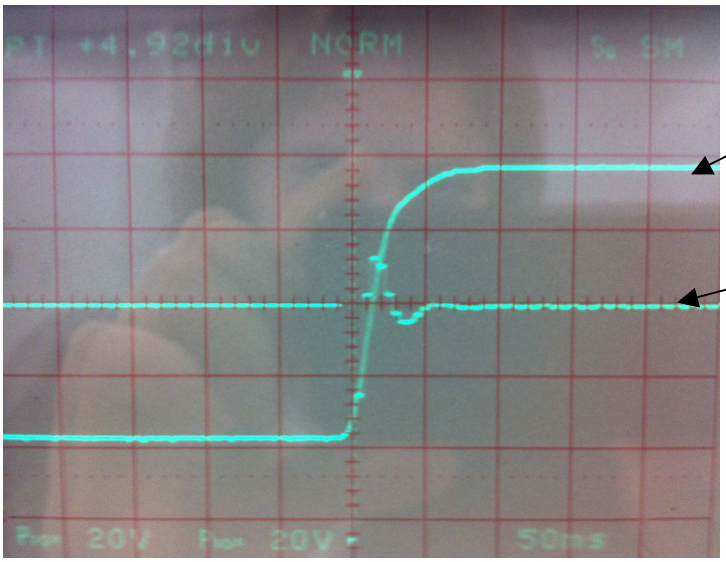
Llegenda:

Lila: Sortida del sistema

Vermell: Entrada del sistema "STEP"

Verd: entrada de control del sistema

Fig. 5.19 Simulació Feedback per retorn d'estat amb pols BESSEL de 0v a 4v



Llegenda:

Canal1 : 2 v/div

Sortida del sistema

Canal2 : 2 v/div

Acció de control

Temps: 50 ms/div

Fig. 5.20 Sistema real per retorn d'estat amb pols BESSEL de 0v a 4v

Comparant els dos valors, els dos s'estabilitzen a 4v i amb un temps de 60 ms.

En la Fig.5.20. es veu el funcionament perfecte del sistema, en comparació amb els resultats del retorn d'estat com no hi ha un retard en les mostres de variable d'estat, ja que amb un observador no es necessari capturar les variables d'estat.

5.5.11. Comprovacións del resultats obtinguts del observador corrent.

En aquest cas no es pot comprovar el seu funcionament amb el matlab ja que no es pot crear un observador corrent amb el matlab. Però si que podem comprar el comportament del observador predictiu amb el observador corrent, ja que ha de tenir la mateix resposta.

Simulació del sistema amb retorn d'estat amb observador corrent de 0v a 4v BESSEL:

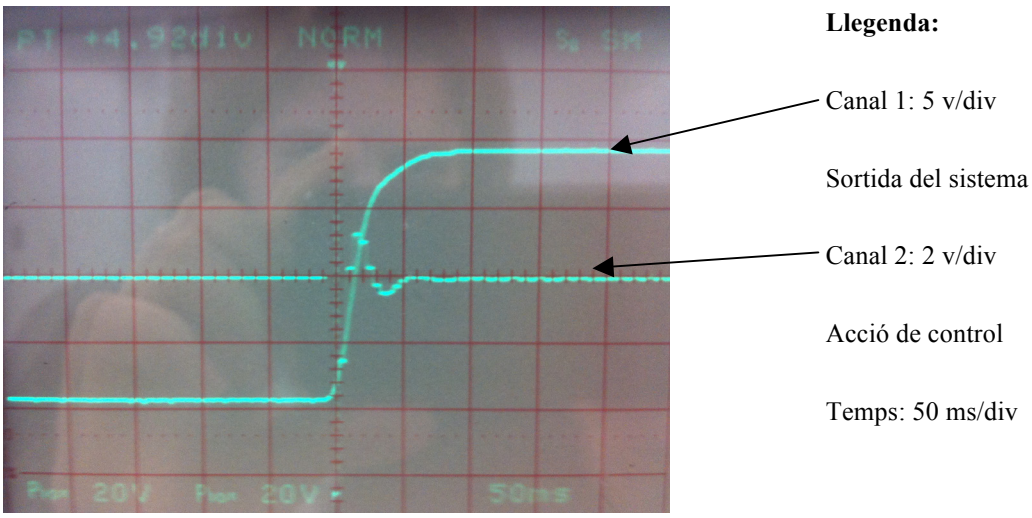


Fig. 5.21 Resposta real amb observador predictiu amb pols BESSEL de 0v a 4v

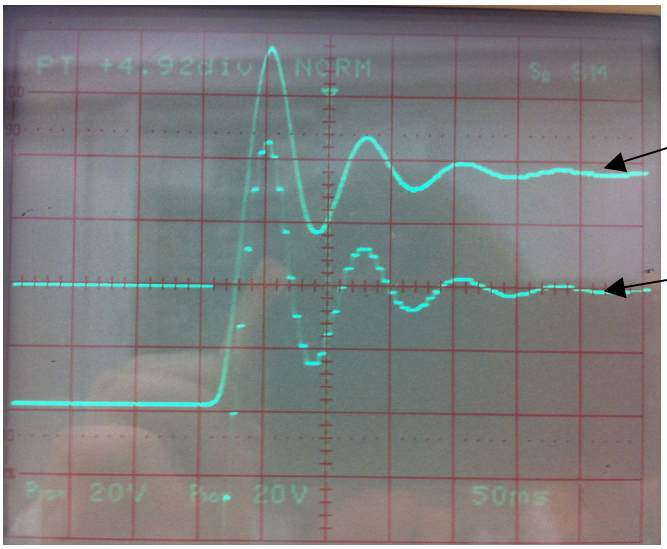


Fig. 5.22 Sistema real amb observador corrent amb pols BESSEL de 0v a 4v

Comparant els dos valors, els dos s'estabilitzen a 4v.

En les gràfiques anteriors es pot observar la gran diferencia entre un observador predictiu i un observador corrent. Aquestes diferencies no es veuen en el sistema del capítol 5, ja que el sistema del capítol 5 té un temps de resposta lent i el ordinador li dona temps de calcular sense massa retard. En canvi per un sistema ràpid com el sistema del capítol 6 es pot veure en la Fig. 6.22 que el sistema es torna més inestable ja que el ordinador té menor temps per calcular les variables d'estat i treure el valor per la sortida.

5.6. Disseny del controlador per retorn d'estat amb observador reduït.

5.6.1. Creació de les Matrius:

En primer lloc, s'ha de crear les matrius reduïdes per poder crear el observador. De manera que aprofitarem les matrius del sistema que ja estan introduïdes el matlab per crear les noves:

```
>> Aaa=ad(1,1);
>> Aab=ad(1,2:3);
>> Aba=ad(2:3,1);
>> Abb=ad(2:3,2:3);
>> Ba=bd(1);
>> Bb=bd(2:3);
```

5.6.2 Càlcul pols observador reduït:

Els pols del controlador seran els mateixos, per tant la K del sistema es la mateixa, però separarem la matriu K en Ka i Kb per posteriorment poder calcular el controlador.

Pols BESSEL:

```
>> Ka=k(1);
>> Kb=k(2:3);
```

Pols del observador:

```
>> podr=Pod(1:2)
podr =   0.4163              0.4305 + 0.1599i
>> Lr=acker(Abb',Aab',podr)'
```

5.6.3 Creació del controlador.

S'aplica la teoria del observador reduït:

```
>> Ar=Abb-Lr*Aab+(Bb-Lr*Ba)*(-Kb);
>> Br=Ar*Lr+Aba-Lr*Aaa+(Bb-Lr*Ba)*(-Ka)
>> Cr=-Kb
>> Dr=-Ka-Kb*Lr
```

5.6.4 Simulació del sistema en llaç tancat controlat amb un observador reduït.

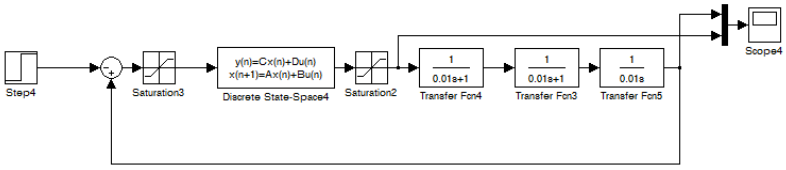
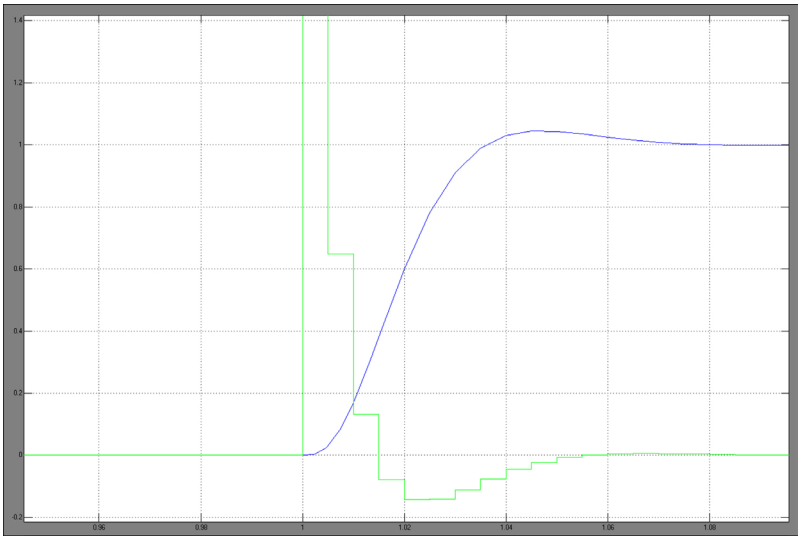


Fig. 5.23 Simulació del sistema amb control per retorn d'estat amb observador reduït

Resposta del sistema:



Llegenda:

Lila: Sortida del sistema

Verd: entrada de control del sistema

Fig. 5.24 Resposta de la simulació amb pols BESSEL.

5.6.5. Programació per visual basic del programa principal.

La programació del programa principal ve a ser el mateix que la programació del sistema per un observador predictiu.

En aquest cas el control del observador reduït s'ha implementat igual que el observador predictiu ja que es la manera més comuna de implementar un observador. Però també es podria implementa un observador reduït de la forma corrent. La diferencia de un observador reduït a un observador complet es nomes el ordre del controlador i la forma de implementar es indiferent.

5.6.6. Programació per visual basic de la rutina d'interrupcions.

Com s'ha comentat en el apartat anterior 6.7.4, la implementació del observador reduït es de igual que el observador complet predictiu però de ordre n-1 es a dir que en el cas de un sistema de ordre 2 el controlador es de ordre 1.

- **Funció d'interrupcions:**

```
Public lectura As Double ' Variable de tipus double per capturar el valor
del A/D
Public valor As Double ' Variable de tipus double, valor de la entrada en
Vols
Public U As Double ' Variable de tipus double on es guarda la sortida
```

```
Public EANT As Double ' Variable de tipus double la variable de X1 anterior
Public E2ANT As Double    ' Variable de tipus double la variable de X2 anterior
Public Xt1 As Double ' Variable de tipus double on es guarda l'aproximació de X1
Public Xt2 As Double ' Variable de tipus double on es guarda l'aproximació de X2
Public Ae1 As Double ' Variable de tipus double, de la multiplicació de A per sortides anteriors
Public Ae2 As Double ' Variable de tipus double, de la multiplicació de A per sortides anteriors
Public Ly1 As Double ' Variable de tipus double on es la matriu L multiplicada per la entrada menys la consigna.
Public Ly2 As Double ' Variable de tipus double on es la matriu L multiplicada per la entrada menys la consigna.
Public Vref As Double ' Variable de tipus Double de la consigna
Public E As Double ' Variable de tipus double de la entrada menys la consigna
Public Ureal As Double ' Variable de tipus Double de la sortida
Public Comp As Double ' Variable de tipus Double del comptador d'interrupcions
Public Sub RutinaServei(ByVal x As Integer) ' Declaració de la funció
    Out &H228, &H0 ' Activar interrupcións
    Comp = Comp + 1 ' Suma 1 al comptador d'errors
    balt = Inp(&H225) And &HF ' Captura valor byte baix del analògic digital
    bbaix = Inp(&H224) And &HFF ' Captura valor byte alt del analògic digital
    lectura = (bbaix + balt * 256) ' Ajuntar byte alt i baix en un double
    valor = ((lectura * 20#) / 4095) - 10 ' Conversió a vols
    E = Vref - valor ' Realimentació/ sortida del sistema menys la consigna.
    Ae1 = 0.1963 * EANT + 0.1863 * E2ANT ' Multiplicació de la matriu a per
    Ae2 = -0.4645 * EANT + 0.383 * E2ANT  'les variables d'estat de una mostra anterior
    Ly1 = -0.7118 * E ' Multiplicació dels observador per la entrada del controlador
    Ly2 = -0.9211 * E ' Multiplicació dels observador per la entrada del controlador
    Xt1 = Ae1 + Ly1  ' Aproximació de X1
    Xt2 = Ae2 + Ly2 ' Aproximació de X2
    Ureal = -0.9105 * Xt1 - 0.5097 * Xt2 - 1.8202 * E ' Calcul del controlador
    EANT = Xt1 ' Guarda la variable d'estat X1
    E2ANT = Xt2  ' Guarda la variable d'estat X2
    U = Ureal ' Canvi de variable el valor.
   If (U > 9.9) Then ' Límits de ±10 v
             U = 9.9
   End If
   If (U < -9.9) Then
             U = -9.9
   End If
   U = ((U + 10) * 4096) / 20 ' Conversió de vols a pas del convertidor D/A
   balt = U \ 256  ' Separa byte alt
   bbaix = U - balt * 256 ' Separar byte baix
   Call Out(&H224, bbaix) ' Escriu el byte baix al convertidor analògic digital
```

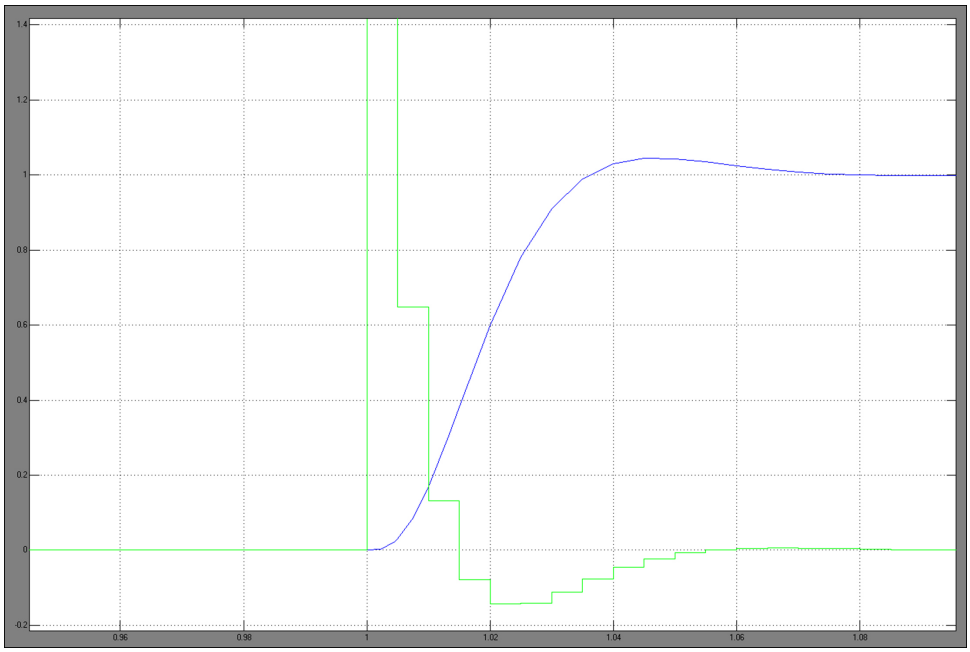
```
   Call Out(&H225, balt) ' Escriu el byte alt al convertidor analògic
digital
    Comp = Comp - 1 ' Resta el comptador d'errors
End Sub
```

5.6.7 Posta en marxa del sistema.

La posta en marxa del sistema es exactament igual que la connexió que es fa per el controlador de retorn d'estat amb observador predictiu. (Punt 6.5.6)

5.6.8. Comprovacions dels resultats obtinguts.

STEP de 0v a 4v BESSEL:

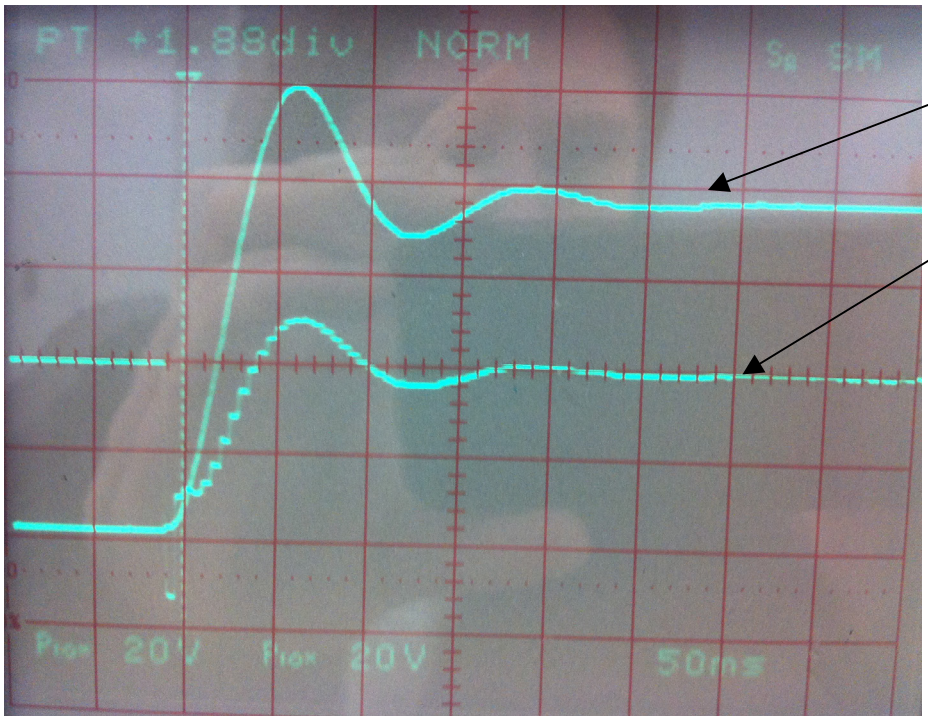


Llegenda:

Lila: sortida del sistema

Verd: Acció de control

Fig. 5.25 Simulació Feedback per observador reduït amb pols BESSEL de 0v a 4v



Llegenda:

Canal 1 : 2 v/div

Sortida del sistema

Canal 2 : 2 v/div

Acció de control

Temps: 50 ms/div

Fig. 5.26 Sistema real per observador reduït amb pols BESSEL de 0v a 4v

Comparant els dos valors, els dos s'estabilitzen a 4v i amb un temps de 60 ms.

El observador reduït, en la teoria té el mateix funcionament que el observador complet, però a la practica apareix un sobre esmorteïment, ja que es deixa de aproximar una variable d'estat.

6. Conclusions.

Les conclusions generals que he pogut extreure del projecte són les següents: s'ha estudiat les diferent possibilitats que tenen les targetes d'adquisició de dades, en concret les targetes: PC-LAB i PCI-6014. Les dues del fabricant Nacional Instruments, tenen la capacitat de treballar en temps real, però també s'ha pogut comprovar que per determinats temps de mostreig s'ha de treballar a baix nivell ja que a alt nivell es generen una sèrie de retards en la execució de les rutines.

Així mateix s'ha pogut crear un control de posició per un motor i estudiar els diferents comportaments segons el controlador i les diferencies que hi ha a nivell pràctic en el moment de implementar-los. Depenent de la velocitat de càlcul del ordinador i la velocitat del convertidor analògic digital, s'ha vist que la implementació de un controlador varia molt la seva resposta.

Per altre banda s'ha pogut crear un control per un sistema de dos LAG amb un Integrador que cada un tenia una tau de 10ms, per tan tenim un sistema molt ràpid, això fa que s'hagi pogut comprovar els efectes que fan cada tipus de controlador i veure com efecte la velocitat del ordinador i del convertidor analògic digital. Els problemes que comporten aquest tipus de targetes en el moment de fer un control per retorn d'estat complet, ja que no es poden capturar totes les variables d'estat en el mateix moment, per tan això fa que amb aquest tipus de controladors hi hagi unes diferencies amb la teoria. El control per retorn d'estat amb observador corrent és la comprovació més clara de com efecte la velocitat de càlcul del ordinador en el control.

A nivell personal, aquest projecte ha estat molt enriquidor, ja que he tingut que aprendre el funcionament de diferents programes com el Labview i el Visual Studio, que considero molt important a nivell professional. Per altre banda he pogut aprendre la problemàtica i solucions que es poden prendre per aquests tipus de controladors a nivell pràctic.

Per tant, la meva valoració d'aquesta tasca és altament positiva, per tot el que m'ha aportat, tant a nivell de coneixements com a nivell de superació personal.

7. Referències.

[1] Digital Control of Dynamic Systems de Gene F.Franklin.

[2] *Apunts de classe de Control avançat i simulació*, Enginyeria Tècnica Industrial, Escola Universitària Politècnica de Mataró, quatrimetre primevera 2011.

[3] *Web de Nacional Intruments,* ni.com

[4] *Web de Feedback group*, www.feedback-group.com

[5] *Web de MathWorks*, mathworks.es

[6] *Web de Microsoft Visual studio*, www.microsoft.com/spain/visualstudio