Bilaga 4, Skapa grafiskt användargränssnitt med *guide*

Enklast till en början är att vid MATLABS kommandoprompt skriva *guide* vilket ger dels ett figurfönster och det som kallas *Guide Control Panel* som enkelt kan liknas vid en verktygslåda där man kan plocka fram de verktyg man behöver.

Följande steg bör tas för att skaffa sig ett grafiskt användargränssnitt eller kort och gott ett GUI.

- Planering av gränssnittet med papper och penna
- Vad skall hända då användaren gör något och om något blir fel. Hur skall det meddelas och/eller åtgärdas.
- Ungefärlig placering av ”pushbutton”, ”edit”-rutor ”StaticText”, ”Slider” och så vidare.
- Snygga till sitt GUI
- Skapa de filer som behövs för att GUI skall fungera.

4.1 Property editor

Detta verktyg är till för att ge de olika objekten i sitt GUI de egenskaper man vill ge det. Dessa egenskaper växlar en del beroende på vilket objekt man arbetar med. De som är gemensamma för de flesta är:

BackgroundColor: bakgrundsfärg
ForegroundColor: textfärg
HandleVisibility: hur skall objektet uppträda
Callback: vad skall hända då objektet aktiveras
Position: var och hur stort skall objektet vara [x-led y-led längd höjd]
Style: vilket objekt är det
Tag: Etikett, kan vara bra då många likadana objekt används
TooltipString: ger ett tips då pekaren placeras över objektet

4.2 Callback editor

Detta verktyg används främst för att ange vad som skall ske då ett objekt används. det kan vara att exekvera ett enskilt kommando eller flera kommandon eller en fil. Man markerar sitt objekt i ”*Objekt browser*” och skriver in det/de

kommandon (även m-filer som sagt) som skall utföras då objektet aktiveras.

4.3 Alignment tool

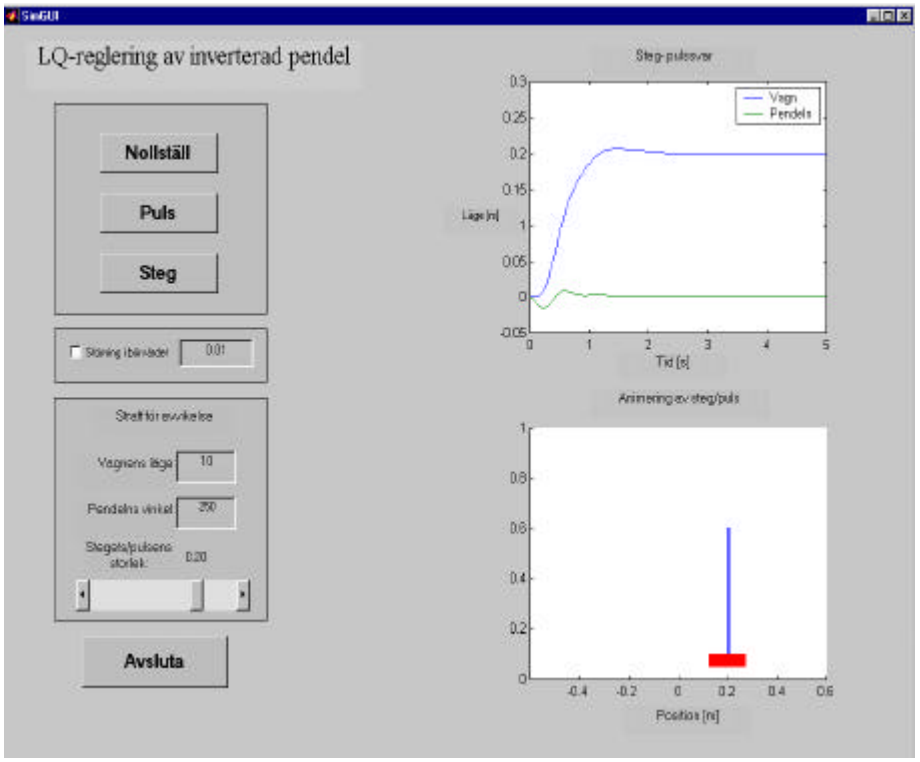
Med detta verktyg snyggar man helt enkelt till sitt GUI så att det ser snyggt och propert ut med jämna avstånd mellan ”pushbutton” och ”edit”-rutor och så vidare.

4.4 Menu editor

Med detta verktyg kan man skapa och ändra menyer i sitt GUI.

4.5 Resultat av ett GUI som gjorts

I figur 1 redovisas den GUI som skapats för att ge möjlighet att experimentera med LQ-reglering av en inverterad pendel.



Figur 1. *Det grafiska gränssnittet till SimGUI*

Programkoden redovisas nedan för LQ-regleringen samt den animering som beskriver steget/impulsen som simulerats. Däremot redovisas inte den kod som ligger till grund för det som visas i figur 1 eftersom detta inte tillför något annat än utfyllnad.

```
function simFUN(action)
%Simulerar och animerar ett steg eller puls  för en
inverterad pendel.
%-------------------------------------------------------------
%                  Ingår i examensarbetet:
%             LQ-reglering av inverterad pendel
%                  Micael Karlsson H97Ei
%-------------------------------------------------------------
if action == 1
```

```
%Simulerar ett steg med aktuella parametrar
%Modellen hämtad ur FEEDBACKs dokumentation
A = [0     0         1        0;...
     0     0         0        1;...
     0   -93.82    0.08485 1.599;...
     0   -93.82   -0.1818     0];
B = [0; 0;  0; 3.6364];

C = [1 0  0 0;...
     0 1 0 0];
D = [0;0];

vagn   = findobj('Tag','vagn');
     x = eval(get(vagn,'String'));
pendel = findobj('Tag','pendel');
     y = eval(get(pendel,'String'));

if x <= 0
   x=1;
   XstraffPtr = findobj('Tag','vagn');
   set(XstraffPtr,'String',x);
end
if y <= 0
   y = 1;
   YstraffPtr = findobj('Tag','pendel');
   set(YstraffPtr,'String',y)
end

Q =[x 0 0 0;...
    0 y 0 0;...
    0 0 0 0;...
    0 0 0 0];
R = 1;

%LQ-reglering
%------------
K  = lqr(A,B,Q,R);
Ac = [(A-B*K)];
Bc = [B];
Cc = [C];
Dc = [D];

Nbar = K(1);
Bcn = [Nbar*B];

stegPtr  = findobj('Tag','stegSlide');
stegval = get(stegPtr,'Value');

%Läser av värdet av rutan "Störning i börvärdet"
brusPtr = findobj('Tag','brus');
brusig = get(brusPtr,'Value');

%Är den markerad görs detta.
if brusig == 1
    brusValPtr = findobj('Tag','brusval');
    Brus = eval(get(brusValPtr,'String'));

    u = stegval; %Stegets storlek
    T = 0:0.05:5;U = u*ones(size(T))+ Brus*randn(size(T));
```

```
    end

    %Om rutan omarkerad görs detta.
    if brusig == 0
       u = stegval; %Stegets storlek
       T = 0:0.05:5;U = u*ones(size(T));
    end


   [Y2,X2] = lsim(Ac,Bcn,Cc,Dc,U,T,[0 0 0 0]);

   vagn   = Y2(:,1);
   pendel = Y2(:,2);
   subplot(2,2,4)
   cla
   %Skapar "objekten" vagn och pendel
   vagn = patch([0 .15 .15 0],[0.05 0.05 .1 1],'r',
   'LineStyle','none');
   pendel = patch([0 .01 .1 0],[.1 .1 .6 .6],'b',
   'LineStyle','none');

   %Skapar axlar fär simuleringen
   axis([-.6 .6  0 1]);

   %Sätter "Grafikbufferten" till dubbel storlek för
   flimmerfri simulering
   set(gcf,'DoubleBuffer','on');

   %Utgångsläge för vagn och pendel
   set(vagn,'XData',[0 .15 .15 0]-0.075+Y2(1,1))
   set(pendel,'XData',[0 .01 .01 0]+Y2(1,1))
   drawnow
   subplot(2,2,2);plot(T,Y2);legend('Vagn','Pendeln')
   a=.01;
   b=0;
   %Ritar ut förflyttningen
   for i = 1:1:size(Y2,1)
      set(vagn,'XData',Y2(i,1)+[0 .15 .15 0]-0.075)
      set(pendel,'XData',Y2(i,1)+[0 .01
      0.01+(0.5*sin(Y2(i,2))) (0.5*sin(Y2(i,2)))])
      drawnow
   end
%-------------------------------------------------------------
elseif action == 2
   %Simulerar en puls med bredden 10 sampel
   A = [0     0         1        0;...
        0     0         0        1;...
        0   -93.82    0.08485 1.599;...
        0   -93.82   -0.1818     0];
   B = [0; 0;  0; 3.6364];

   C = [1 0  0 0;...
        0 1 0 0];
   D = [0;0];

   vagn   = findobj('Tag','vagn');
        x = eval(get(vagn,'String'));
   pendel = findobj('Tag','pendel');
        y = eval(get(pendel,'String'));
```

```
if x <= 0
   x=1;
   XstraffPtr = findobj('Tag','vagn');
   set(XstraffPtr,'String',x);
end
if y <= 0
   y = 1;
   YstraffPtr = findobj('Tag','pendel');
   set(YstraffPtr,'String',y)
end


Q =[x 0 0 0;...
    0 y 0 0;...
    0 0 0 0;...
    0 0 0 0];
R = 1;


%LQ-reglering
%------------
K  = lqr(A,B,Q,R);
Ac = [(A-B*K)];
Bc = [B];
Cc = [C];
Dc = [D];

Nbar = K(1);
Bcn = [Nbar*B];

stegPtr = findobj('Tag','stegSlide');
stegval = get(stegPtr,'Value');

%Läser av värdet av rutan "Störning i börvärdet"
brusPtr = findobj('Tag','brus');
brusig  = get(brusPtr,'Value');

%Är den markerad görs detta.
if brusig == 1
   brusValPtr = findobj('Tag','brusval');
   Brus = eval(get(brusValPtr,'String'));

   u = stegval; %Stegets storlek
   T = 0:0.05:5;U = zeros(size(T));
   U(1:10) = stegval+ Brus*randn(1,10);
 end

 %Om rutan omarkerad görs detta.
 if brusig == 0
    u = stegval; %Stegets storlek
    T = 0:0.05:5;U = zeros(size(T));U(1:10)=stegval;
 end


[Y2,X2] = lsim(Ac,Bcn,Cc,Dc,U,T,[0 0 0 0]);

vagn   = Y2(:,1);
pendel = Y2(:,2);
```

```
   subplot(2,2,4)
   cla
   %Skapar "objekten" vagn och pendel
   vagn = patch([0 .15 .15 0],
   [0.05 0.05 .1 .1],'r','LineStyle','none');
   pendel = patch([0 .01 .1 0],
   [.1 .1 .6 .6],'b','LineStyle','none');

   %Skapar axlar fär simuleringen
   axis([-.6 .6  0 1]);

   %Sätter "Grafikbufferten” till dubbel storlek för
   flimmerfri simulering
   set(gcf,'DoubleBuffer','on');

   %Utgångsläge för vagn och pendel
   set(vagn,'XData',[0 .15 .15 0]-0.075+Y2(1,1))
   set(pendel,'XData',[0 .01 .01 0]+Y2(1,1))
   drawnow
   subplot(2,2,2);
   plot(T,Y2);legend('Vagn','Pendeln')
   a=.01;
   b=0;
   %Ritar ut förflyttningen
   for i = 1:1:size(Y2,1)
      set(vagn,'XData',Y2(i,1)+[0 .15 .15 0]-0.075)
      set(pendel,'XData',Y2(i,1)+[0 .01
      0.01+(0.5*sin(Y2(i,2))) (0.5*sin(Y2(i,2)))])
      drawnow
   end

elseif action == 3
   %Utgångsläge för vagn och pendel
   subplot(2,2,4)
   cla
   set(gcf,'DoubleBuffer','on');
   vagn = patch([0 .15 .15 0],[0.05 0.05 .1
   .1],'r','LineStyle','none');
   pendel = patch([0 .01 .1 0],[.1 .1 .6
   .6],'b','LineStyle','none');
   set(vagn,'XData',[0 .15 .15 0]-0.075)
   set(pendel,'XData',[0 .01 .01 0])
   drawnow
   subplot(2,2,2);cla


elseif action == 4
   %Tar hand om steg/puls-slider
   global stegval
   stegPtr  = findobj('Tag','stegSlide');
   stegval = get(stegPtr,'Value');
   textPtr = findobj('Tag','steg_storlek');
   set(textPtr,'String',num2str(stegval,2));
end
```