University Of Balamand

Faculty of Engineering

Fuzzy Logic Control

Fuzzy Control of a Two-Tank System

Presented to: Dr. Issam Dagher

Presented by: Kifah DAHER

Date: 23-10-2012

Introduction:

This report introduces two tank system and a typical procedure used to design and realize of a fuzzy controller. To simulate a fuzzy control system it is necessary to specify a mathematical model of the Two Tank system. Using MATLAB a code representing the mathematical model for the pendulum is integrated, membership functions are realized. As well implementing the system using SIMULINK is another way to simulate the case. The model of the Two Tank System was built up by combining the MATLAB functions, which describes the tank dynamics, and the function implemented inside the block Fuzzy-Controller. The last function was build with the aid of the MATLAB Fuzzy Toolbox.

The two tank system represented by a two tank x1 and x2, and a valve as an input u. first tank x1 receives liquid from u, then water is flowed into tank x2. The goal is to decide when to open the valves, realize what will happen in the tanks and at what time the whole system can stop.

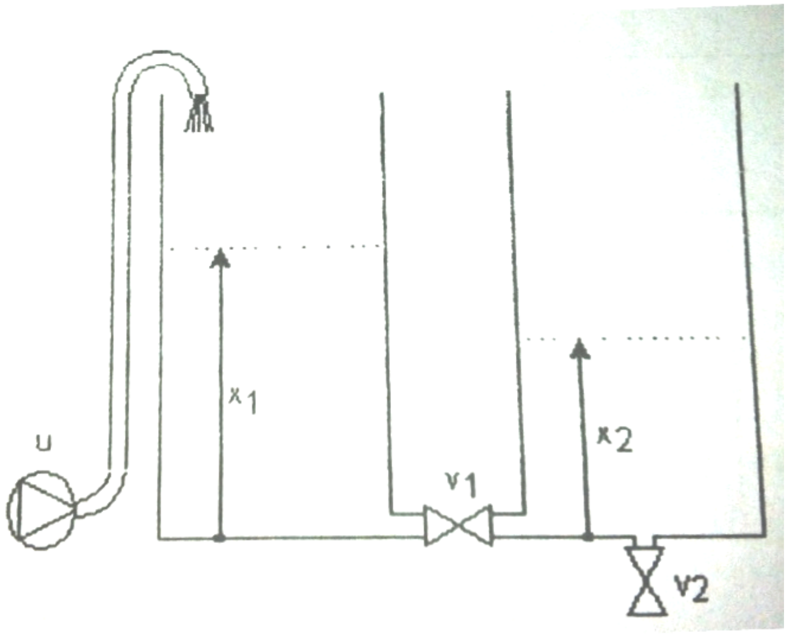


Figure 1: the Two Tank system

The mathematical model is represented by a differential equation of second order which require using the ode23 command in MATLAB in order to solve.

One model for the Two Tank system is given by the set of nonlinear ordinary differential equation:

$$f = 0.06624 v1 \sqrt{|x1 - x2|}\ sign(x1 - x2)$$

$$\dot{x}1 = 0.067u - f$$

$$\dot{x}2 = f - 0.0605\, r\, v2\, |x2|^{0.43}$$

Where

$$r = \begin{cases} 1.2 & : x2 < 16\ cm \\ 1 & : x2 \geq 16\ cm \end{cases}$$

The system includes characteristics of the liquid. The valve positions v1=0.4 and v2=0.3. The task is to control the liquid level x2.

The fuzzy controller used accordingly is the following:

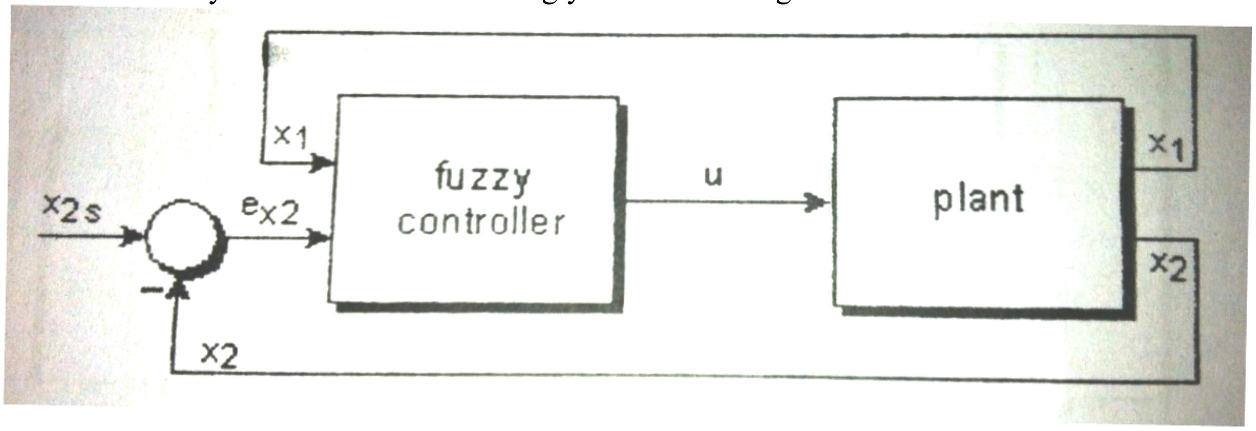


Figure 2: Fuzzy control of liquid level x2

First of all the membership function of the system are implemented in MATLAB, where the liquid levels x1 & x2 are the inputs whereas u is the output of the fuzzy controller.

The two input membership functions are of triangular and trapezoidal types which involve the uses of "trimf" & "trapmf" commands.

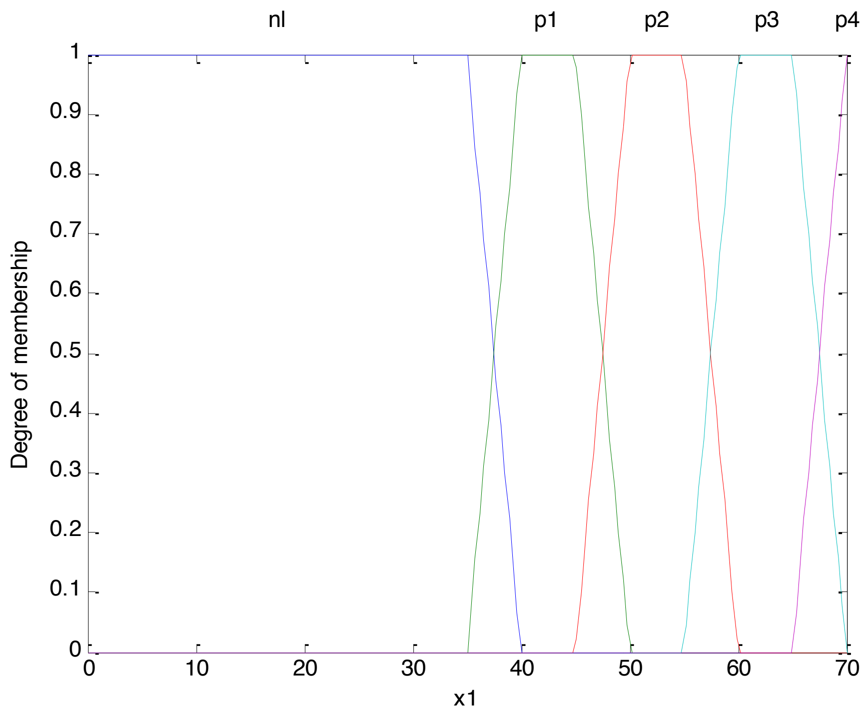


Figure 3: Membership function of input1 x1

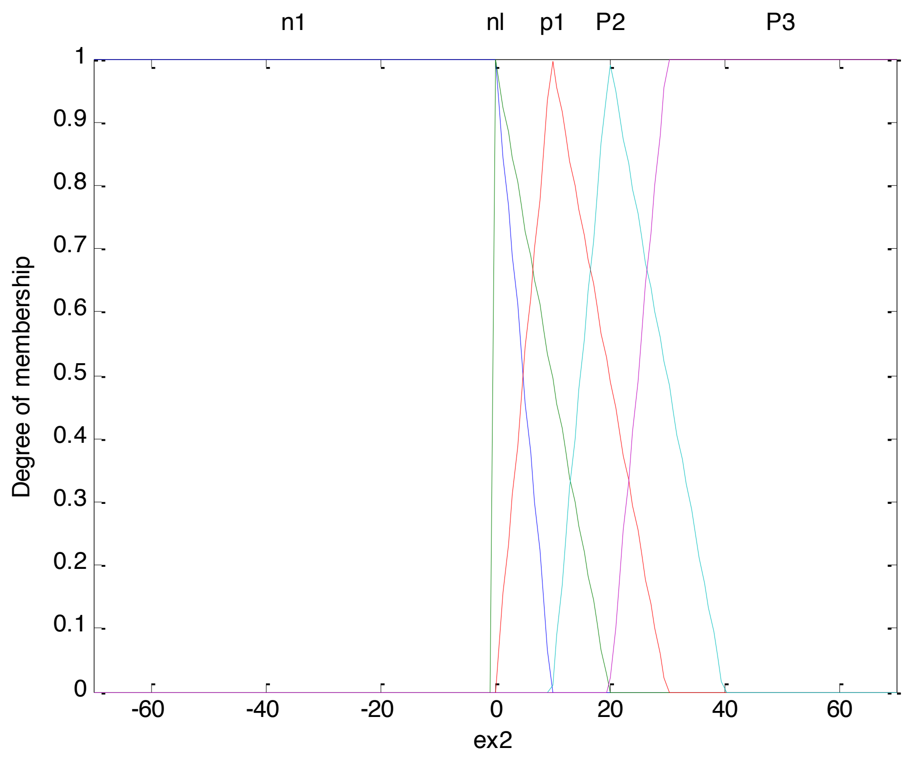


Figure 4: Membership Function of input2 x2

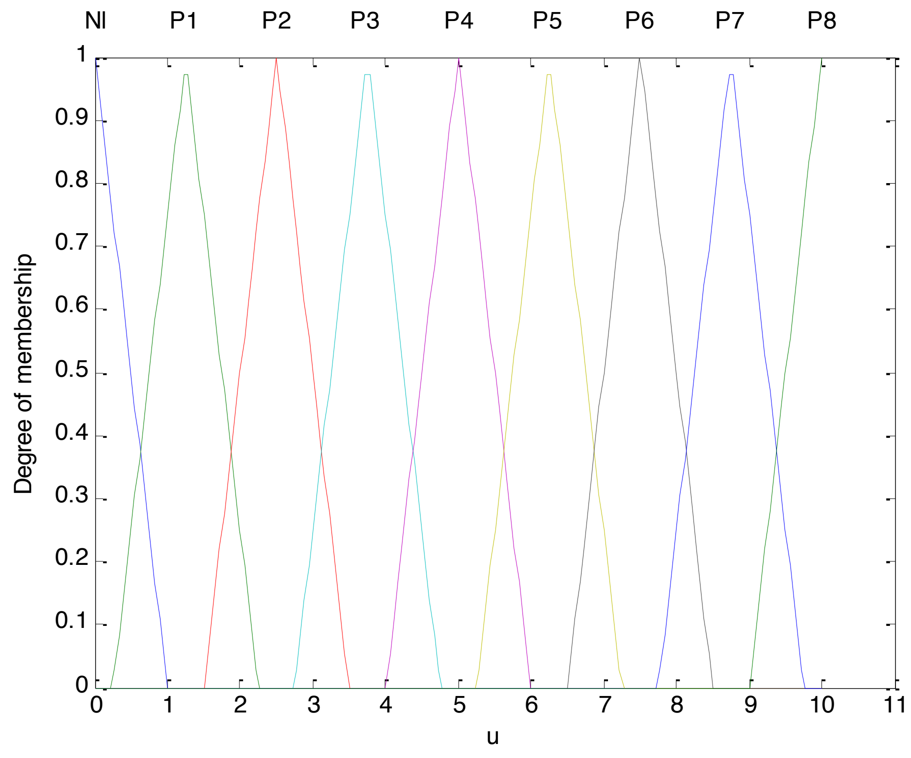


Figure 5: Membership Function of the output for FC1

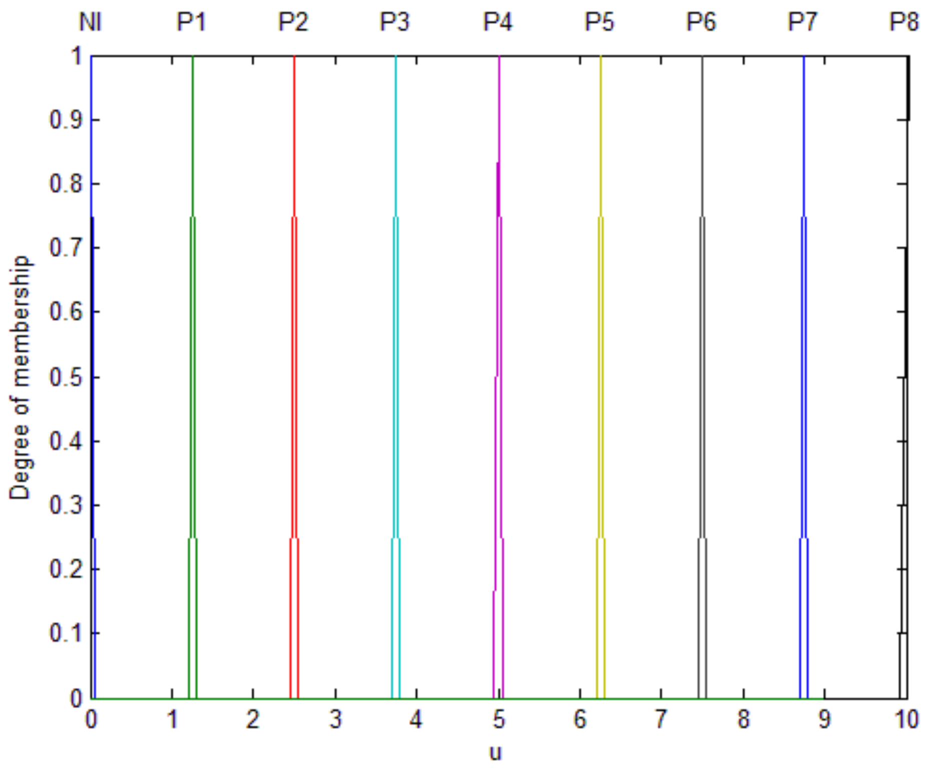


Figure 6: Membership function of the output for FC2

They describe the liquid level of the tanks as well the response of the output which is built based on the following rules:

x1 / ex2	nl	p1	p2	p3	p4
p3	p8	p7	p5	p3	nl
p2	p7	p6	p4	p3	nl
p1	p7	p5	p3	p2	nl
nl	p4	p3	p2	p1	nl
n1	nl	nl	nl	nl	nl

Task a:

The system is solved and described using MATLAB via Code and SIMULINK fuzzy controller:

(a1) The Matlab code describing the system is the following:

```
function dx = TwoTankFC1(t,x)
a=newfis('TwoTankFC1');
```

```
a=addvar(a,'input','x1',[0 70]);
a=addmf(a,'input',1,'nl','trapmf',[0 0 35 40]);
a=addmf(a,'input',1,'p1','trapmf',[35 40 45 50]);
a=addmf(a,'input',1,'p2','trapmf',[45 50 55 60]);
a=addmf(a,'input',1,'p3','trapmf',[55 60 65 70]);
a=addmf(a,'input',1,'p4','trapmf',[65 70 70 70]);
a=addvar(a,'input','ex2',[-70 70]);
a=addmf(a,'input',2,'n1','trapmf',[-70 -70 0 10]);
a=addmf(a,'input',2,'nl','trimf',[0 0 20]);
a=addmf(a,'input',2,'p1','trimf',[0 10 30]);
a=addmf(a,'input',2,'P2','trimf',[10 20 40]);
a=addmf(a,'input',2,'P3','trapmf',[20 30 70 70]);
a=addvar(a,'output','u',[0 10]);
a=addmf(a,'output',1,'Nl','trimf',[0 0 1]);
a=addmf(a,'output',1,'P1','trimf',[0.25 1.25 2.25]);
a=addmf(a,'output',1,'P2','trimf', [1.5 2.5 3.5]);
a=addmf(a,'output',1,'P3','trimf', [2.75 3.75 4.75]);
a=addmf(a,'output',1,'P4','trimf', [4 5 6]);
a=addmf(a,'output',1,'P5','trimf', [5.25 6.25 7.25]);
a=addmf(a,'output',1,'P6','trimf', [6.5 7.5 8.5]);
a=addmf(a,'output',1,'P7','trimf', [7.75 8.75 9.75]);
a=addmf(a,'output',1,'P8','trimf', [9 10 10]);
rulelist=[1 1 1 1 1;
          2 1 1 1 1;
          3 1 1 1 1;
          4 1 1 1 1;
          5 1 1 1 1;
          1 2 5 1 1;
          2 2 4 1 1;
          3 2 3 1 1;
          4 2 2 1 1;
          5 2 1 1 1;
          1 3 8 1 1;
          2 3 6 1 1;
          3 3 4 1 1;
          4 3 3 1 1;
          5 3 1 1 1;
          1 4 8 1 1;
          2 4 7 1 1;
          3 4 5 1 1;
          4 4 4 1 1;
          5 4 1 1 1;
          1 5 9 1 1;
          2 5 8 1 1;
          3 5 6 1 1;
          4 5 4 1 1;
          5 5 1 1 1];
      a=addrule(a,rulelist);
      inp=[x(1) -x(2)];
      global U;
      U=evalfis(inp,a);
      if x(2)<16
          r=1.2;
      else
          r=1;
      end
      v1=0.4;
```

```
        v2=0.3;
        f= 0.06624*v1*sqrt(abs(x(1)-x(2)))*sign(x(1)-x(2));
        dx(1)=0.067*U-f;
        dx(2)=f-0.0605*r*v2*(abs(x(2)))^0.43;
        dx=dx';
end
```

Using SIMULINK the plant of the system is constructed representing the derivatives of x1 and x2 and the differential equations:

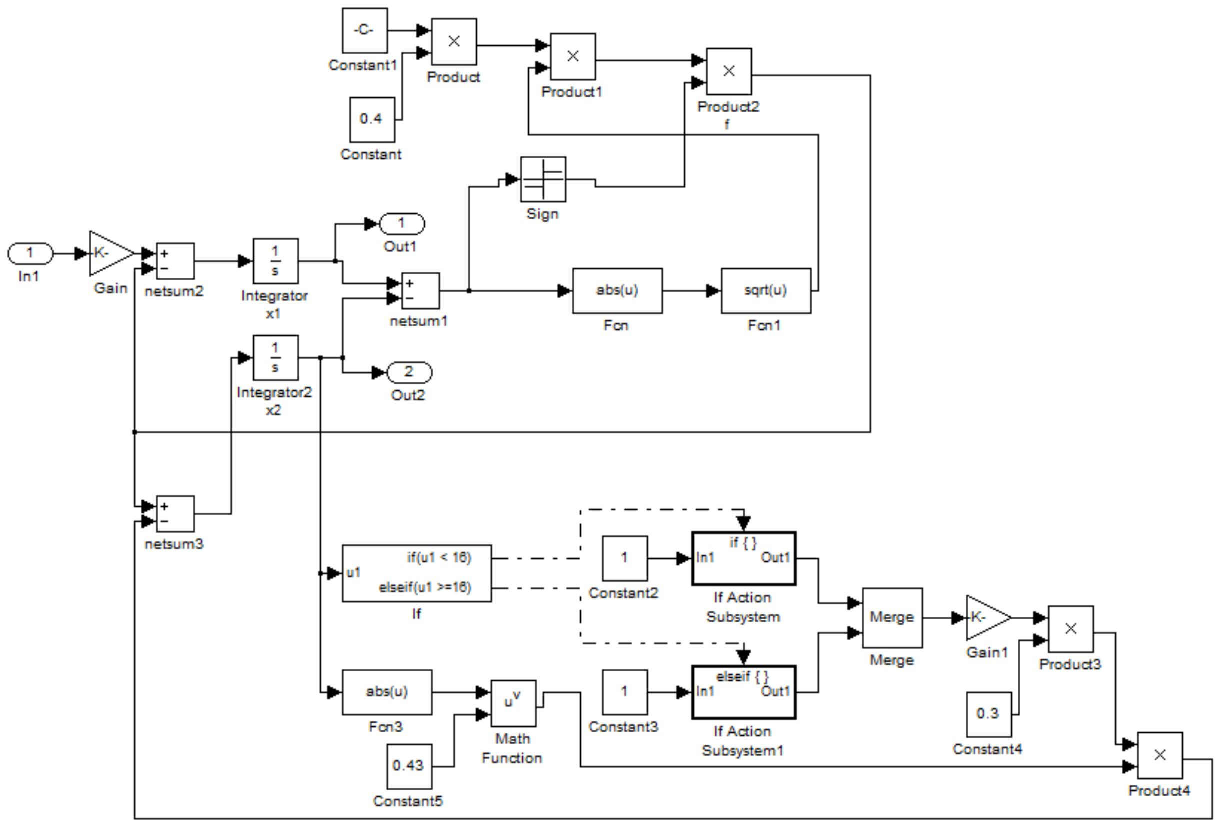


Figure 7: Plant of the system

We represent the plant as masked block (subsystem) with one input and two outputs. It is described in the figure below, simulated with a fuzzy logic controller. The liquid levels x1 and x2 as well u the output are displayed on the scope.

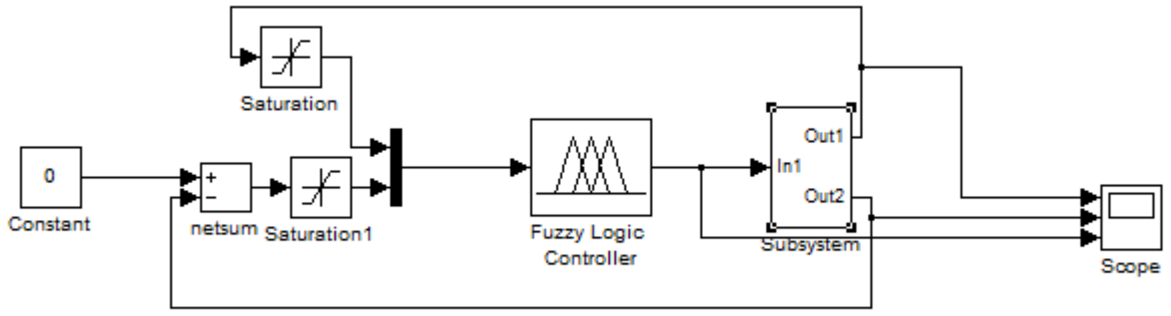


Figure 8: The overall system

fuzzy command to creat a new fis then implemented in the fuzzy logic controller to simulate the system.

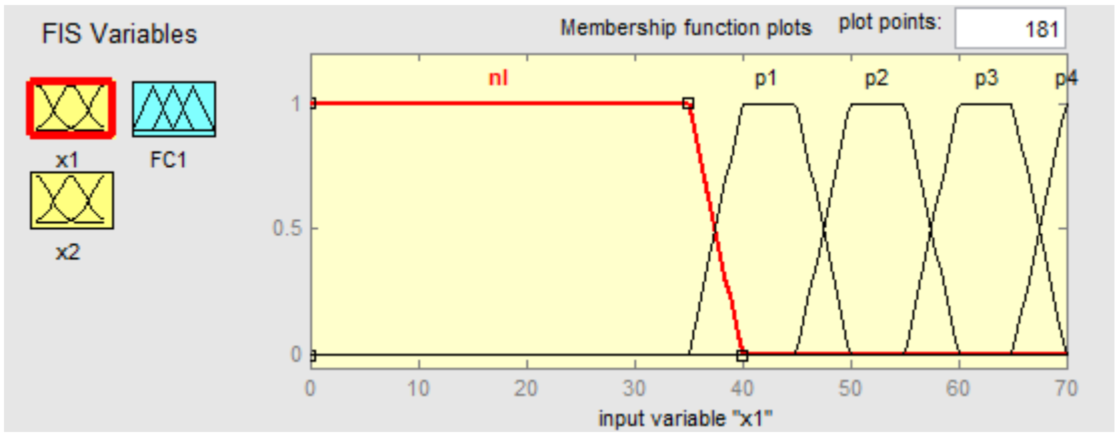


Figure 9: membership function of input x1 for FC1

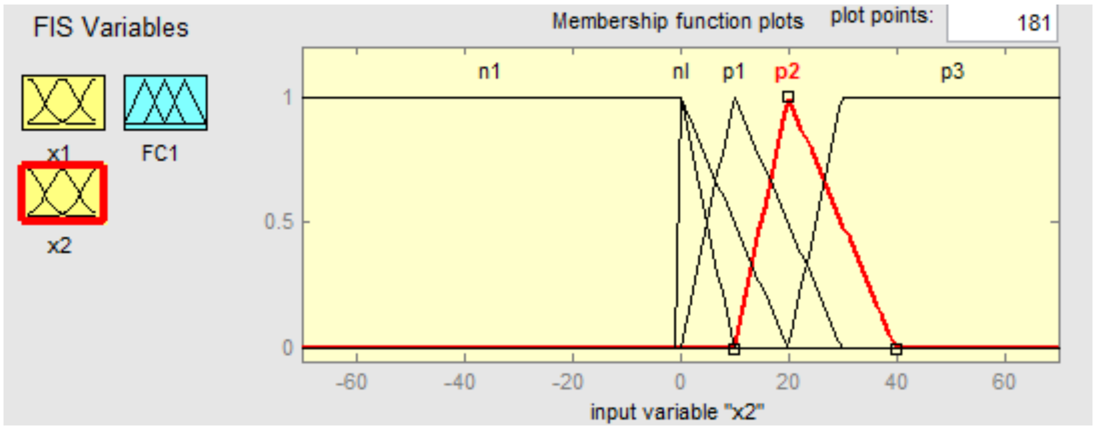


Figure 10: membership function of input x2 for FC1

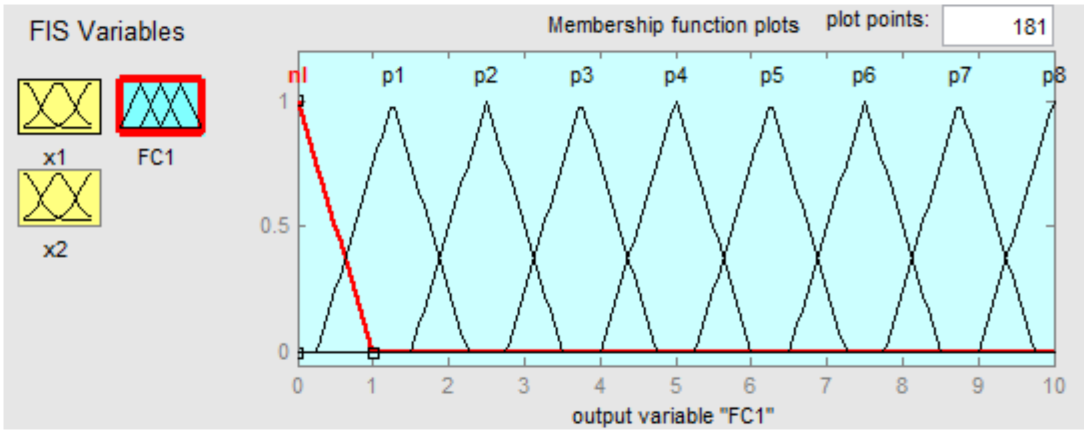
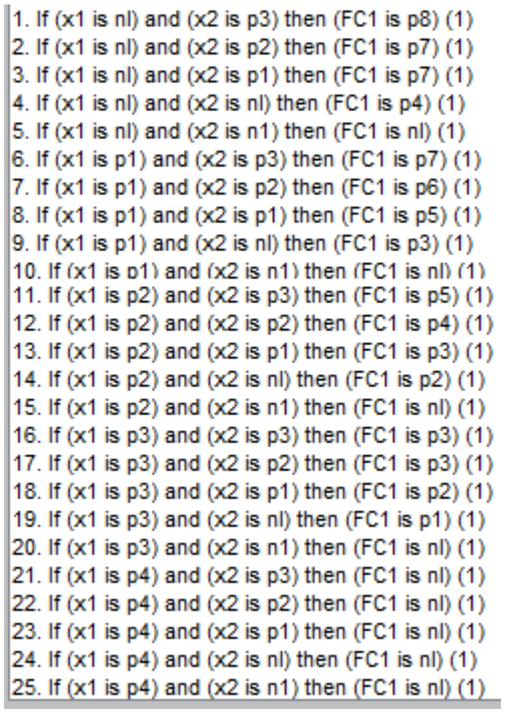


Figure 11: membership function of output u for FC1

The table below represents the rules related between the input and the outputs:


1. If (x1 is nl) and (x2 is p3) then (FC1 is p8) (1)
2. If (x1 is nl) and (x2 is p2) then (FC1 is p7) (1)
3. If (x1 is nl) and (x2 is p1) then (FC1 is p7) (1)
4. If (x1 is nl) and (x2 is nl) then (FC1 is p4) (1)
5. If (x1 is nl) and (x2 is n1) then (FC1 is nl) (1)
6. If (x1 is p1) and (x2 is p3) then (FC1 is p7) (1)
7. If (x1 is p1) and (x2 is p2) then (FC1 is p6) (1)
8. If (x1 is p1) and (x2 is p1) then (FC1 is p5) (1)
9. If (x1 is p1) and (x2 is nl) then (FC1 is p3) (1)
10. If (x1 is p1) and (x2 is n1) then (FC1 is nl) (1)
11. If (x1 is p2) and (x2 is p3) then (FC1 is p5) (1)
12. If (x1 is p2) and (x2 is p2) then (FC1 is p4) (1)
13. If (x1 is p2) and (x2 is p1) then (FC1 is p3) (1)
14. If (x1 is p2) and (x2 is nl) then (FC1 is p2) (1)
15. If (x1 is p2) and (x2 is n1) then (FC1 is nl) (1)
16. If (x1 is p3) and (x2 is p3) then (FC1 is p3) (1)
17. If (x1 is p3) and (x2 is p2) then (FC1 is p3) (1)
18. If (x1 is p3) and (x2 is p1) then (FC1 is p2) (1)
19. If (x1 is p3) and (x2 is nl) then (FC1 is p1) (1)
20. If (x1 is p3) and (x2 is n1) then (FC1 is nl) (1)
21. If (x1 is p4) and (x2 is p3) then (FC1 is nl) (1)
22. If (x1 is p4) and (x2 is p2) then (FC1 is nl) (1)
23. If (x1 is p4) and (x2 is p1) then (FC1 is nl) (1)
24. If (x1 is p4) and (x2 is nl) then (FC1 is nl) (1)
25. If (x1 is p4) and (x2 is n1) then (FC1 is nl) (1)

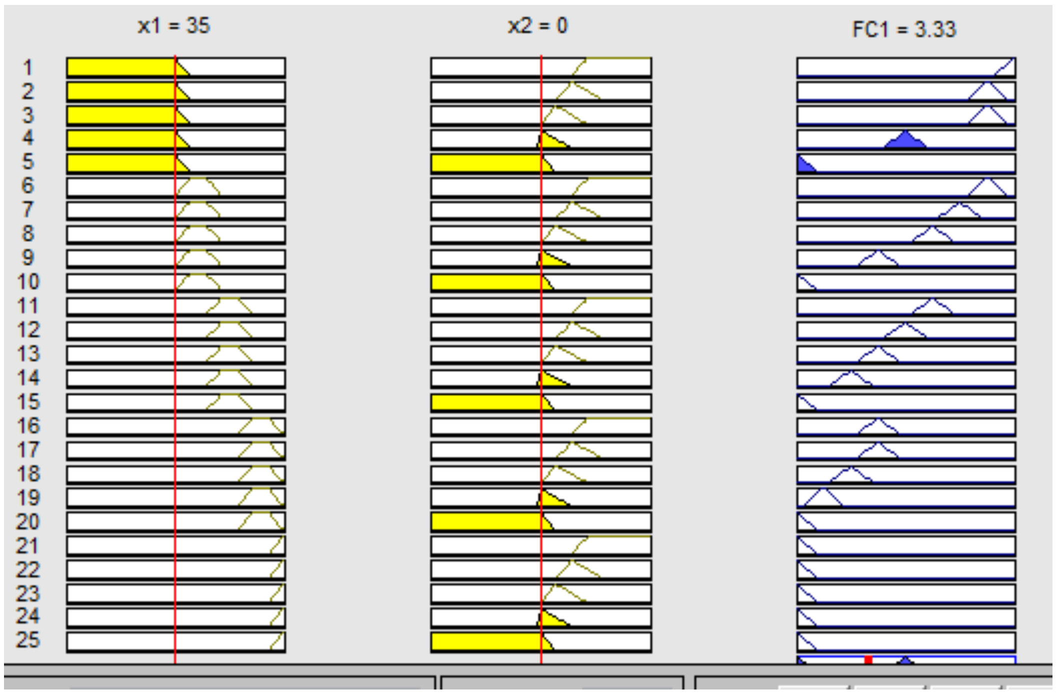
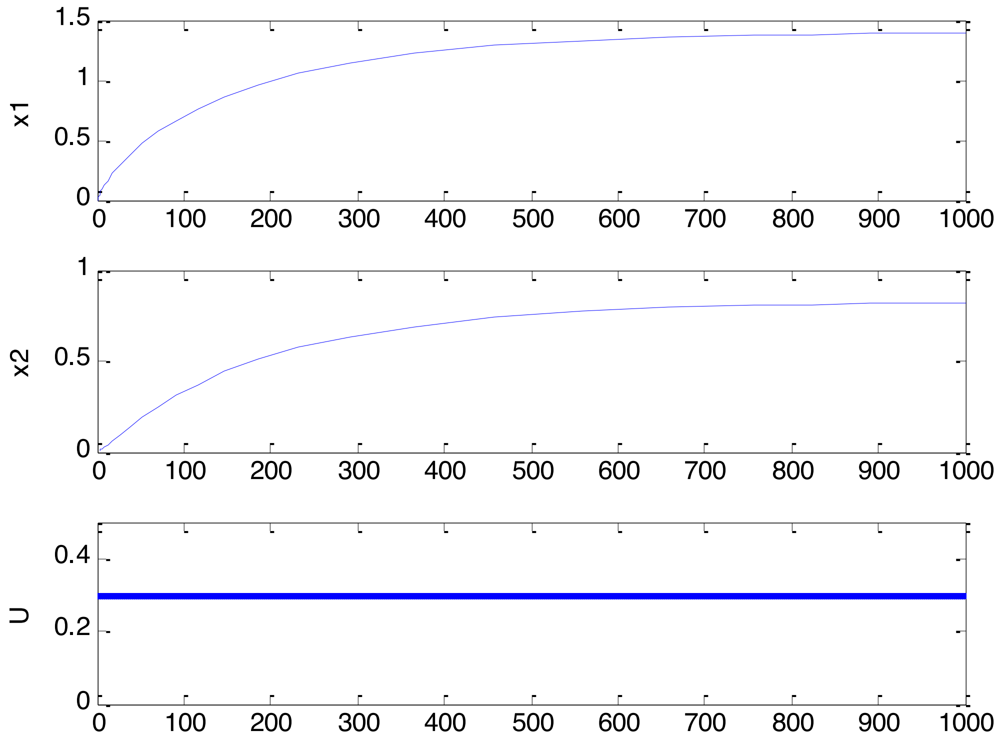


Figure 12: Rules set

(a2) Results from Matlab code for zero initial condition for FC1:

```
tic
[t0,y]=ode23('TwoTankFC1',[0 1000],[0 0])
subplot(3,1,1),plot(t0,y(:,1))
ylabel ('x1')
subplot(3,1,2),plot(t0,y(:,2))
ylabel ('x2')
subplot(3,1,3),plot(t0,U)
ylabel ('U')
axis ([0 1000 0 0.5])
timeFC1=toc
```



The calculation time ta_{fc1} is timeFC1 =   0.8603

The generated surface of the fuzzy controller characteristic in 3D is:

```
gensurf(a);
```

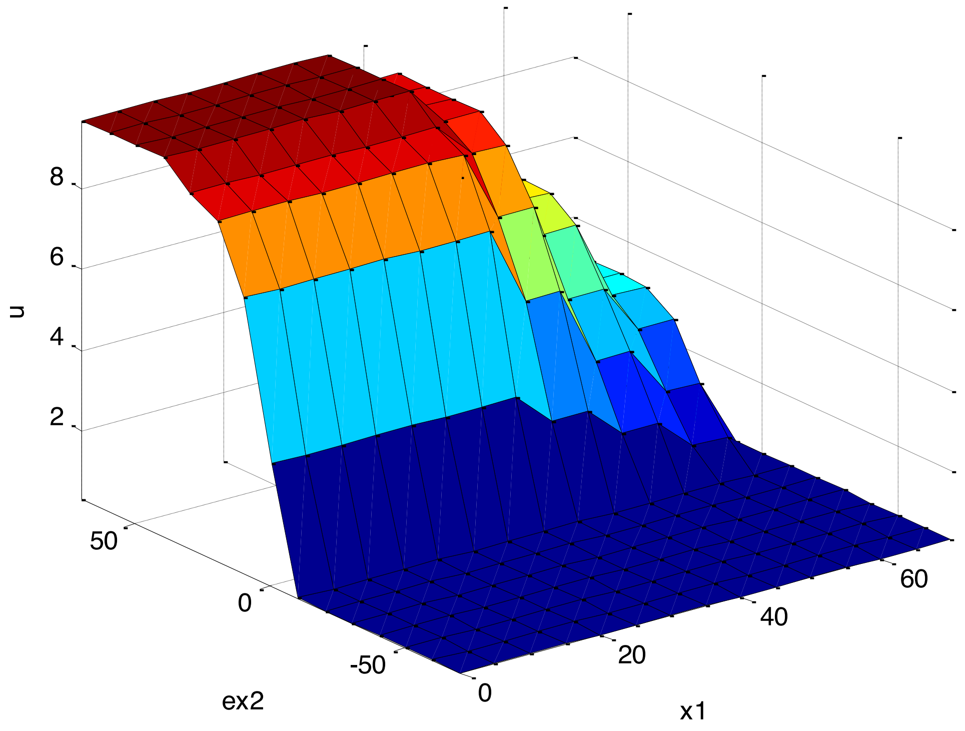


Figure 13: 3D dimensional characteristic for FC1

Results from SIMULINK and fuzzy controller for zero initial condition for FC1:

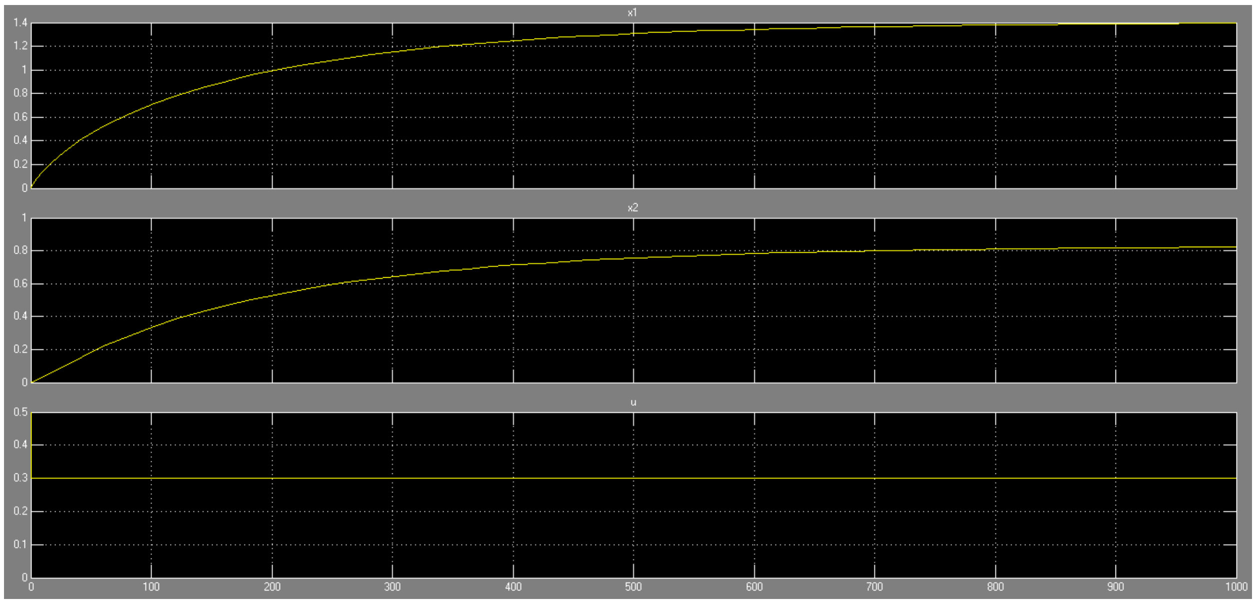


Figure 14: Output from the scope

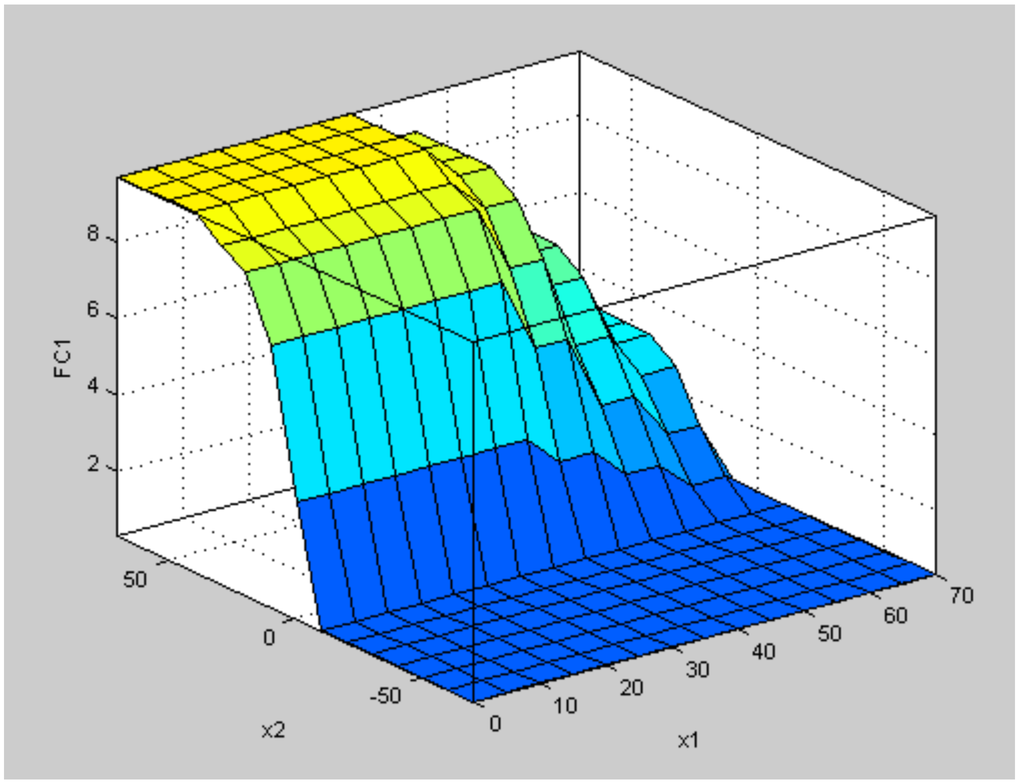


Figure 15: 3 dimensional characteristic surface for FC1

(a3) repeating the same task of (a2) for FC2 for singleton membership function shown in figure 6

Matlab code:

```
function dx = TwoTankFC2(t,x)
a=newfis('TwoTankFC2');
a=addvar(a,'input','x1',[0 70]);
a=addmf(a,'input',1,'nl','trapmf',[0 0 35 40]);
a=addmf(a,'input',1,'p1','trapmf',[35 40 45 50]);
a=addmf(a,'input',1,'p2','trapmf',[45 50 55 60]);
a=addmf(a,'input',1,'p3','trapmf',[55 60 65 70]);
a=addmf(a,'input',1,'p4','trapmf',[65 70 70 70]);
a=addvar(a,'input','ex2',[-70 70]);
a=addmf(a,'input',2,'nl','trapmf',[-70 -70 0 10]);
a=addmf(a,'input',2,'nl','trimf',[0 0 20]);
a=addmf(a,'input',2,'p1','trimf',[0 10 30]);
a=addmf(a,'input',2,'P2','trimf',[10 20 40]);
a=addmf(a,'input',2,'P3','trapmf',[20 30 70 70]);
a=addvar(a,'output','u',[0 10]);
a=addmf(a,'output',1,'Nl','trimf',[0 0 1]);
a=addmf(a,'output',1,'P1','trimf',[1.25 1.25 1.25]);
a=addmf(a,'output',1,'P2','trimf', [2.5 2.5 2.5]);
a=addmf(a,'output',1,'P3','trimf', [3.75 3.75 3.75]);
a=addmf(a,'output',1,'P4','trimf', [5 5 5]);
a=addmf(a,'output',1,'P5','trimf', [6.25 6.25 6.25]);
a=addmf(a,'output',1,'P6','trimf', [7.5 7.5 7.5]);
a=addmf(a,'output',1,'P7','trimf', [8.75 8.75 8.75]);
a=addmf(a,'output',1,'P8','trimf', [10 10 10]);
```

```
rulelist=[1 1 1 1 1;
          2 1 1 1 1;
          3 1 1 1 1;
          4 1 1 1 1;
          5 1 1 1 1;
          1 2 5 1 1;
          2 2 4 1 1;
          3 2 3 1 1;
          4 2 2 1 1;
          5 2 1 1 1;
          1 3 8 1 1;
          2 3 6 1 1;
          3 3 4 1 1;
          4 3 3 1 1;
          5 3 1 1 1;
          1 4 8 1 1;
          2 4 7 1 1;
          3 4 5 1 1;
          4 4 4 1 1;
          5 4 1 1 1;
          1 5 9 1 1;
          2 5 8 1 1;
          3 5 6 1 1;
          4 5 4 1 1;
          5 5 1 1 1];
      a=addrule(a,rulelist);
      inp=[x(1) -x(2)];
      global U;
      U=evalfis(inp,a);
      if x(2)<16
          r=1.2;
      else
          r=1;
      end
      v1=0.4;
      v2=0.3;
      f= 0.06624*v1*sqrt(abs(x(1)-x(2)))*sign(x(1)-x(2));
      dx(1)=0.067*U-f;
      dx(2)=f-0.0605*r*v2*(abs(x(2)))^0.43;
      dx=dx';
end

tic
[t0,y]=ode23('TwoTankFC2',[0 1000],[0 0])
subplot(3,1,1),plot(t0,y(:,1))
ylabel ('x1')
subplot(3,1,2),plot(t0,y(:,2))
ylabel ('x2')
subplot(3,1,3),plot(t0,U,'-b')
ylabel ('U')
axis ([0 1000 0 0.5])
timeFC2=toc
```

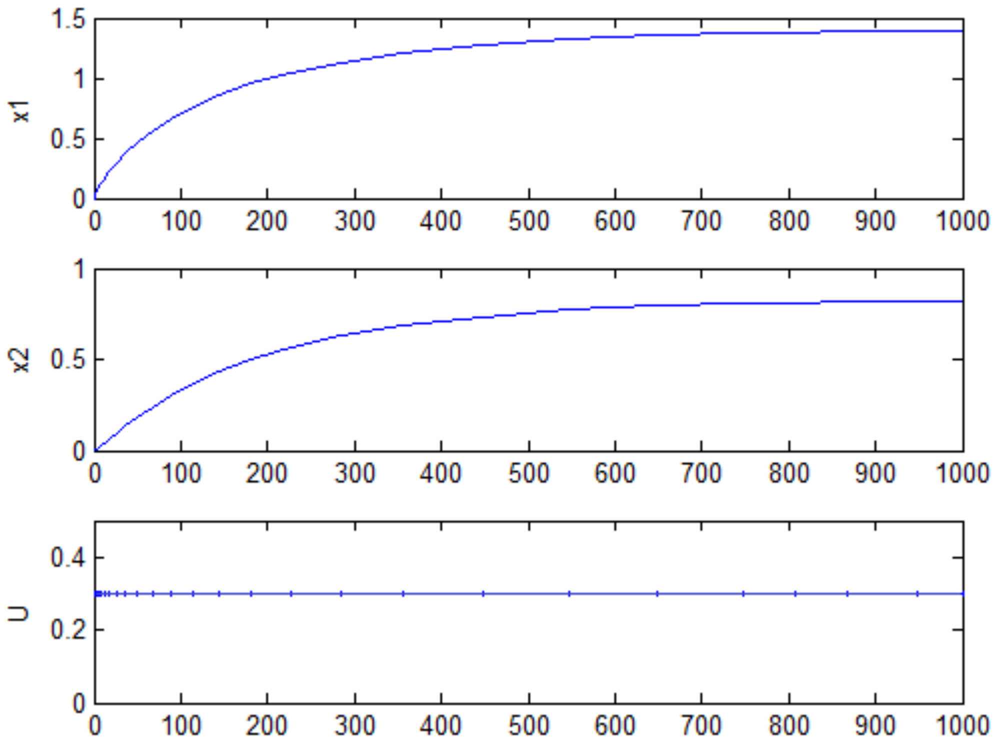


Figure 16: Matlab result for FC2 zero initial condition

The calculation time ta$_{fc1}$ is timeFC2 = 0.8020

The generated surface of the fuzzy controller characteristic in 3D is:

```
gensurf(a);
```

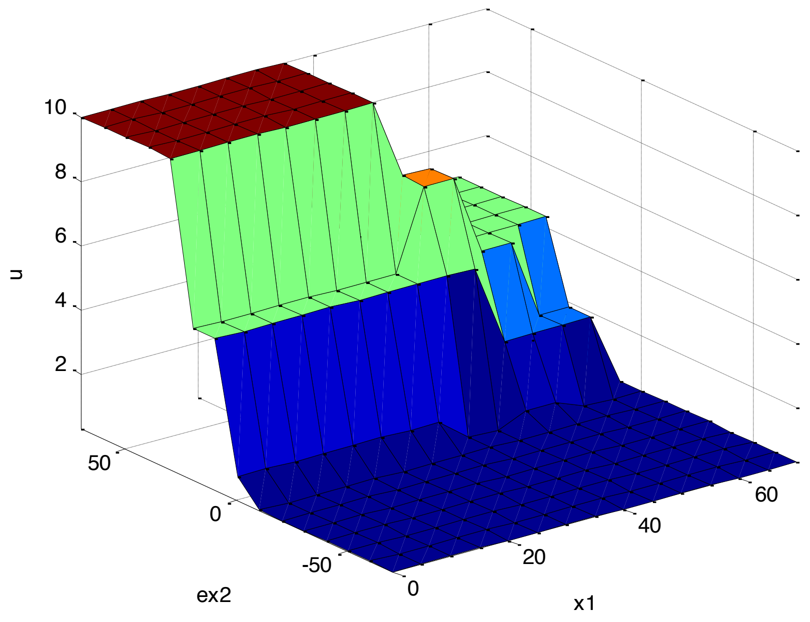


Figure 17: 3D characteristic surface for FC2

Results from SIMULINK and fuzzy controller for zero initial condition for FC2:

Keeping the plant as it is for FC1 and just modifying the output FC2 as singletons running the simulation on FC2 fuzzy logic controller the results are:

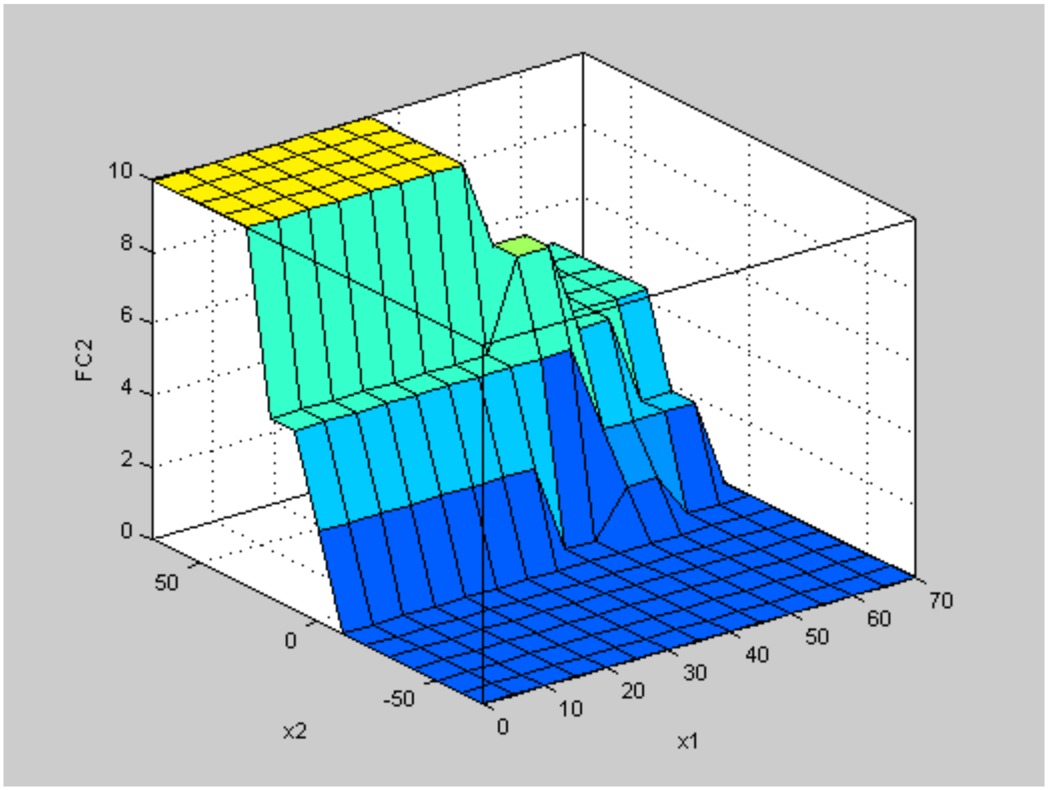
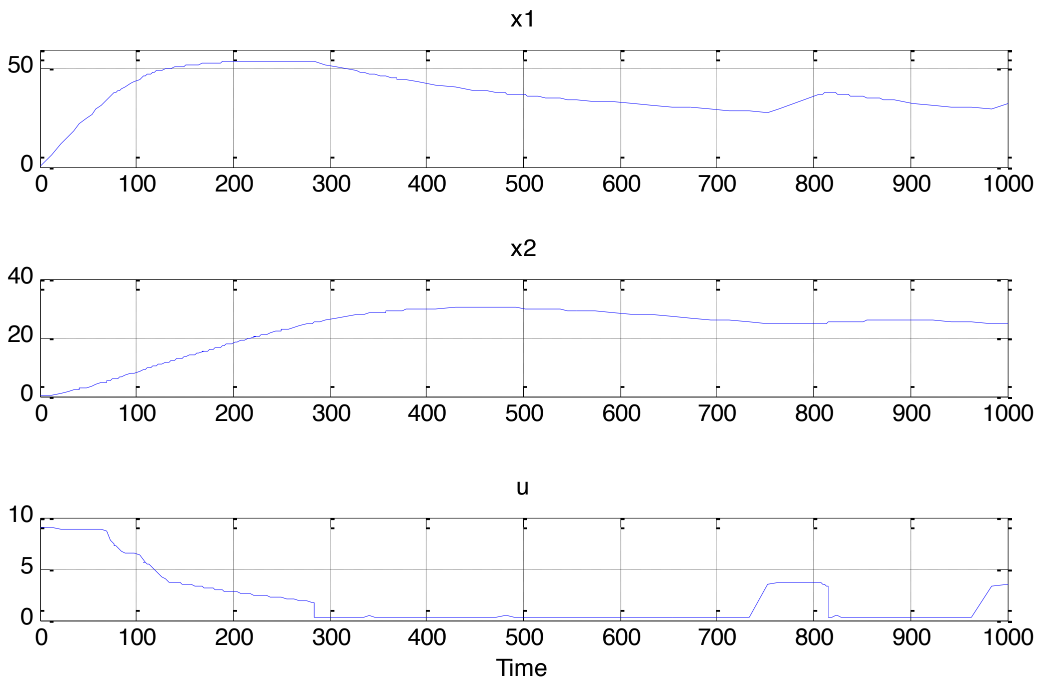


Figure 18: 3D surface for FC2

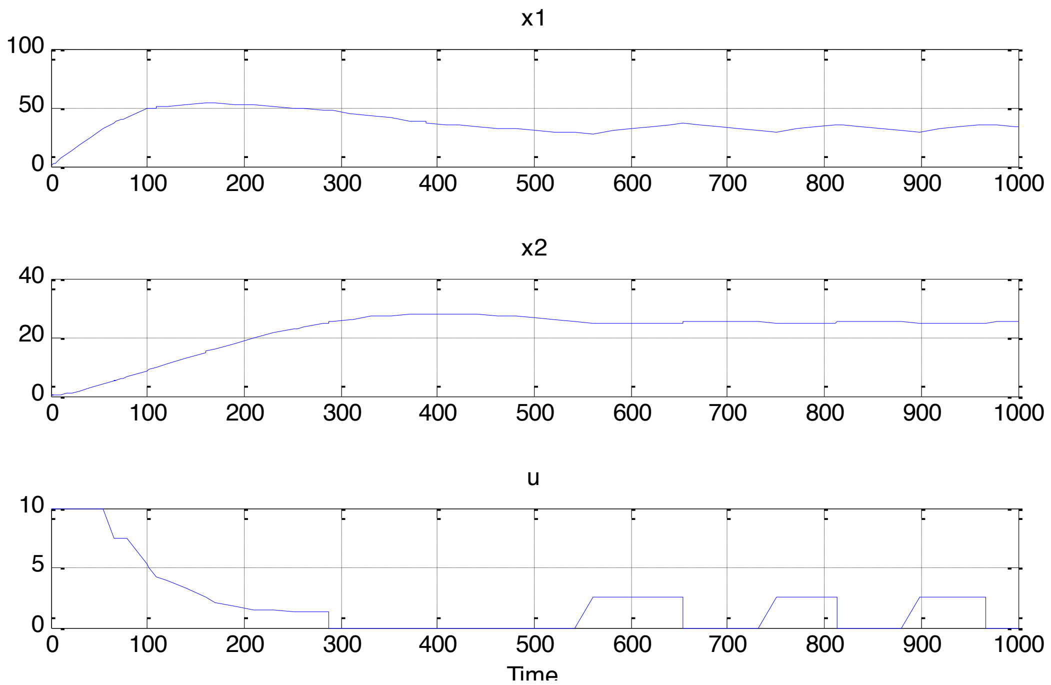
Task b:

(b1) simulating FC1 for $x_{2s} = 25$ cm and for 1000seconds.



$tb_{fc1} = 1.0669$

(b2) for FC2



tb_{fc2} = timeFC2 = 1.4324

The ratio tb_{fc1}/tb_{fc2} = 1.0669/1.4324 = 0.7448

Task c:

(c1) it is the case when adding weights to the rules of FC2 according to the following table:

ex2 \ x1	nl	p1	p2	p3	p4
p3	1	1	0.1	1	1
p2	1	1	0.1	1	1
p1	1	1	0.1	1	1
nl	1	1	0.1	1	1
n1	1	1	0.1	1	1

For approximately the same results tc_{fc3} = timeFC3 = 0.8524 which took more time then weight 1.

To implement the weight in Matlab we have to modify the mapping rules as following:

```
rulelist=[1 1 1 1 1;
          2 1 1 1 1;
          3 1 1 0.1 0.1;
          4 1 1 1 1;
          5 1 1 1 1;
          1 2 5 1 1;
          2 2 4 1 1;
          3 2 3 0.1 0.1;
          4 2 2 1 1;
          5 2 1 1 1;
          1 3 8 1 1;
          2 3 6 1 1;
          3 3 4 0.1 0.1;
          4 3 3 1 1;
          5 3 1 1 1;
          1 4 8 1 1;
          2 4 7 1 1;
          3 4 5 0.1 0.1;
          4 4 4 1 1;
          5 4 1 1 1;
          1 5 9 1 1;
          2 5 8 1 1;
          3 5 6 0.1 0.1;
          4 5 4 1 1;
          5 5 1 1 1];
```

As well in the fuzzy toolbox we can change the weight when mapping rules from 1 to 0.1.

(c2) The outstanding features of the Fuzzy Toolbox are:

- The graphical capabilities of the *FIS Editor*.
- The *Rule Viewer* can display the whole fuzzy interference process based on the interference diagram.
- The *SurfaceViewer* helps to visualize the output of the system versus any one or two inputs to the system.

Introduction:

The system was simulated using Matlab and fuzzy toolbox as well Simulink to build the plant system. The simulation results are almost the same. The difference is that Fuzzy toolbox and Simulink are easy user interface simulators and GUI which allow the user to input, modify and run the simulation easily. Moreover, the rules mapping is easier than the code process since it is graphical and rule viewer can display the whole fuzzy interference process.