Universidad Peruana de Ciencias Aplicadas

Facultad de Ingeniería
Escuela de Ingeniería Electrónica

Curso: Robótica e Inteligencia Artificial

Informe del Examen Final

Estudiante: Manco Fernández, Fiorela

Profesor: Dante Anael Vargas Machuca Acevedo

Julio – 2021

1. Controlador difuso de posición y velocidad para el motor DC

Se realiza la simulación de un controlador difuso, con tres funciones de membresía, para el motor DC 222049. En la Figura 1 se puede observar los tres conjuntos de valores de las funciones de membresía para el error angular, la velocidad angular y el voltaje de salida del motor.

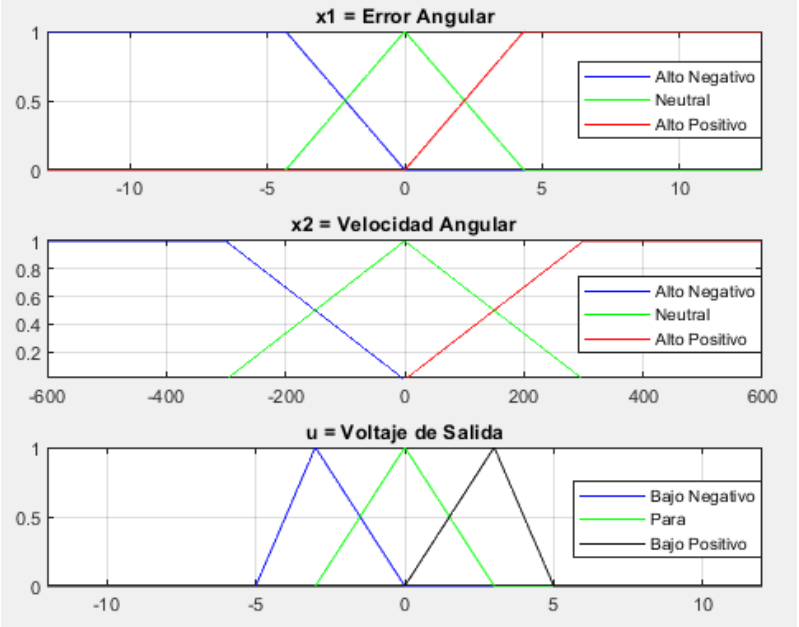

Figura 1. Tres funciones de membresía para el error angular (x1), velocidad angular (x2) y el voltaje de salida (u).
Fuente: Elaboración propia (2021)

En la Tabla 1 se muestra la Memoria Asociativa Difusa (FAM) calculada para cada caso de los valores de la velocidad y error. También, cabe resaltar que 'BN' significa bajo negativo, 'BP' significa bajo positivo y 'P' significa para.

Tabla 1. Memoria Asociativa Difusa (FAM) para 3 conjuntos difusos

Velocidad / Error	**Negativo**	**Neutral**	**Positivo**
Negativo	BN	BN	BP
Neutral	BN	P	BP
Positivo	BN	BP	BP

Fuente: Elaboración propia (2021)

Luego, se procede a realizar la defuzzificación, donde se requiere obtener la intersección de los valores alfa-cut y los vectores de la función de membresía de voltaje. En la Figura 2 se puede observar las salidas difusas de las 3 funciones de membresía, así como la salida de corte y del motor.

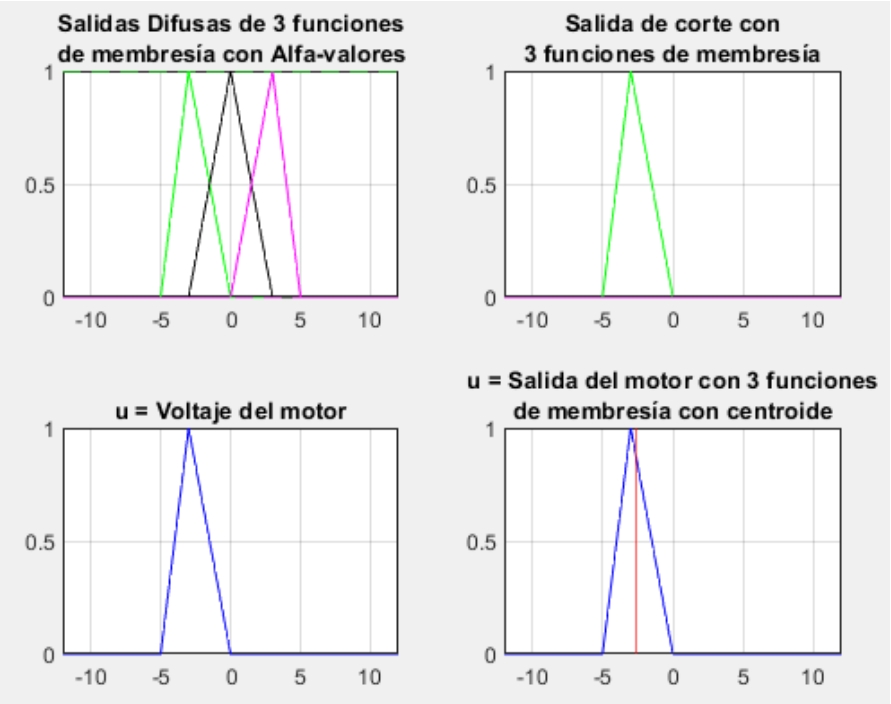

Figura 2. Salidas Difusas con 3 funciones de membresía.
Fuente: Elaboración propia (2021)

Se realiza el control difuso para el motor elegido. En la Figura 3 se visualiza la corriente X3, la velocidad X2 y la posición X1 con un setpoint de 5. También se observa el voltaje que empieza en el valor de 2.5 V hasta llegar a cero. De la misma manera, se muestra el error, el cual empieza en el valor de 5 (setpoint) hasta ser muy cercano a cero.

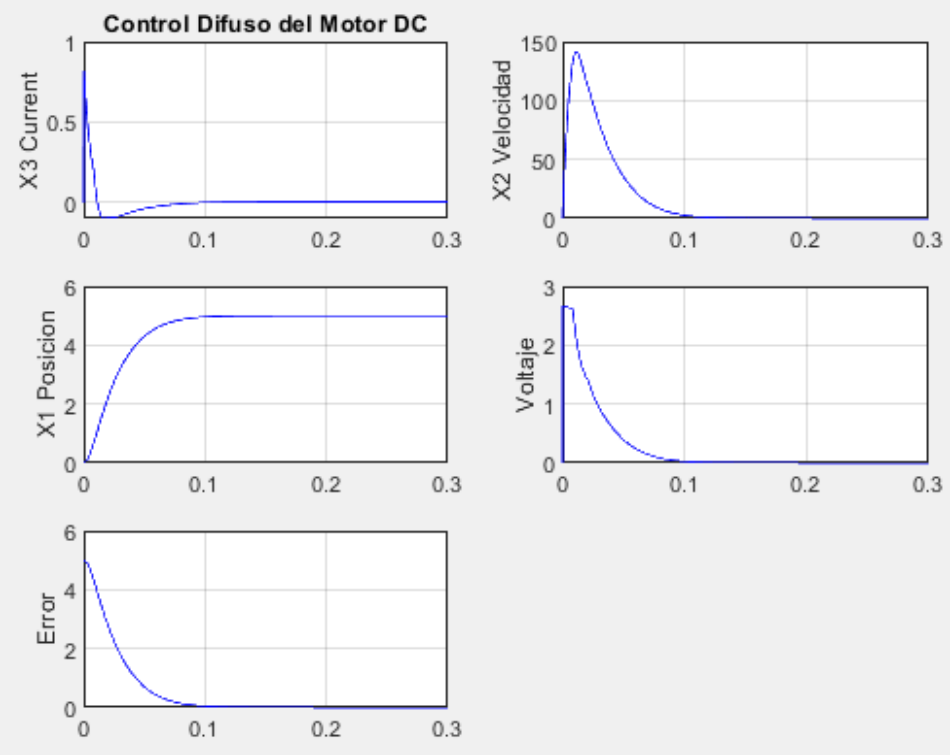

Figura 3. Control Difuso del Motor DC con *setpoint* igual a 5.
Fuente: Elaboración propia (2021)

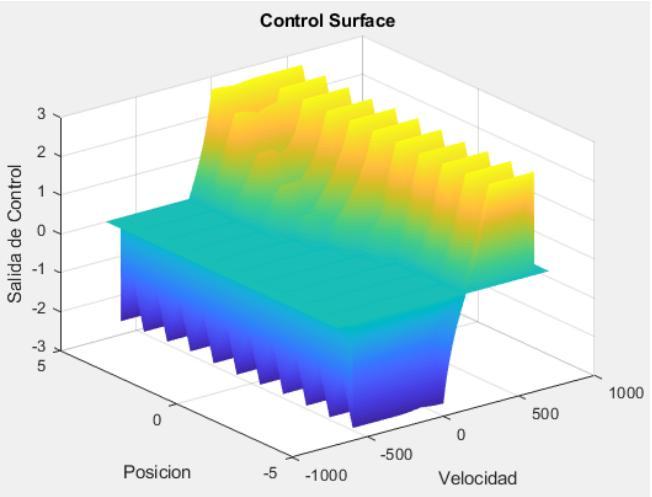

Figura 4. Gráfica de superficie de las 3 variables de control: posición, velocidad y salida del controlador.

Fuente: Elaboración propia (2021)

Se realiza el sistema estable no homogéneo, cuyas respuestas se pueden visualizar en las Figuras 5 y 6. En la primera Figura en en respuesta con una entrada de rampa y la segunda es en respuesta a una entrada sinusoidal.

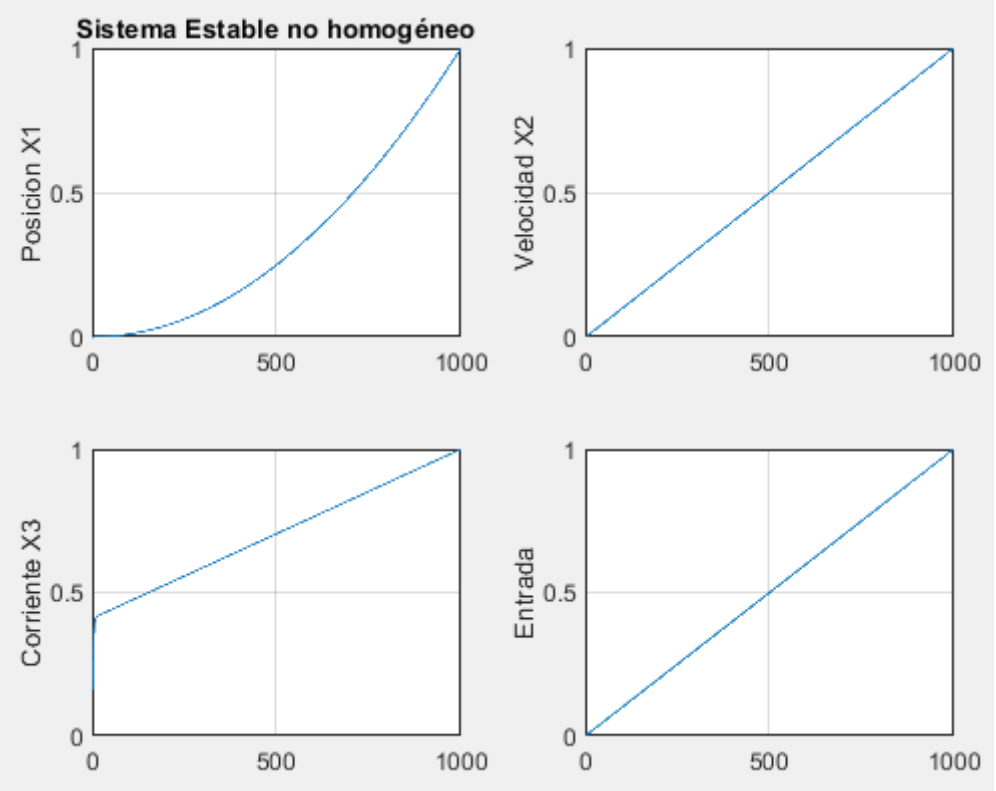

Figura 5. Sistema estable no homogéneo con entrada rampa.

Fuente: Elaboración propia (2021)

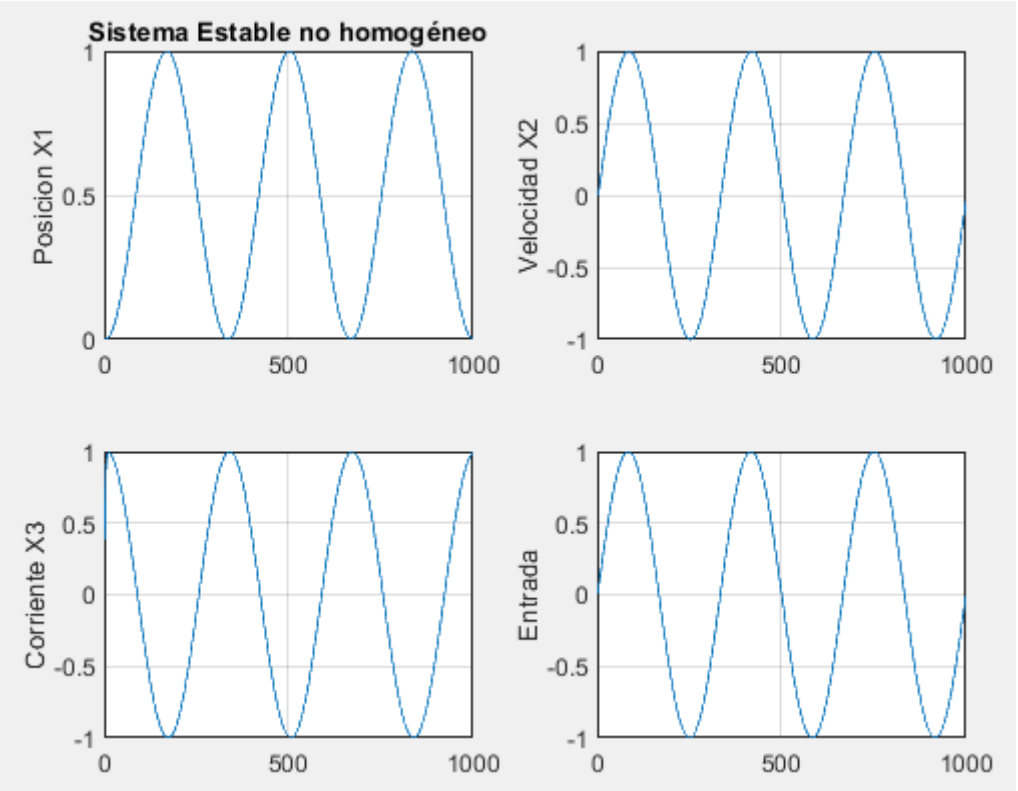

Figura 6. Sistema estable no homogéneo con entrada sinusoidal.
Fuente: Elaboración propia (2021)

2. Brazo robot de 2 eslabones

Se realiza el Robot Plotter de 2 eslabones, cuya tarea es la escritura de la letra 'T'. Para ello se realizó el diseño de un controlador PID, y su respectivo cálculo de ángulos mediante cinemática inversa.

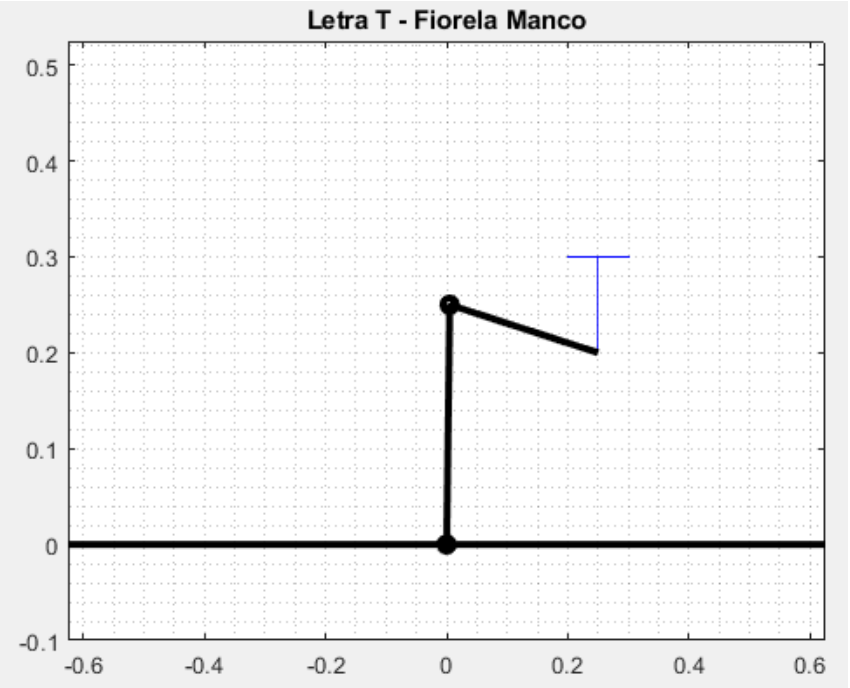

Figura 7. Brazo Robot de 2 eslabones escribiendo la letra 'T'.
Fuente: Elaboración propia (2021)

3. Red neuronal

Se implementa una red neuronal Deep FeedForward, cuyo objetivo es reconocer entre dos rostros animados. Para el entrenamiento, se eligieron los rostros que se visualizan en la Figura 8.

Figura 8. Imágenes de rostros de aprendizaje de la red neuronal con sus respectivos nombres.
Fuente: Elaboración propia (2021)

Para la validación, se utilizaron 2 imágenes diferentes de cada rostro para que la red neuronal pueda identificarla. En la Figura 9 se pueden visualizar los resultados obtenidos.

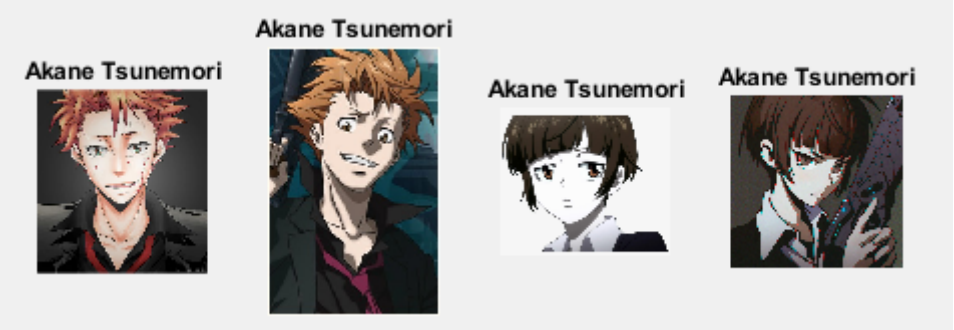

Figura 9. Resultados de la red neuronal.
Fuente: Elaboración propia (2021)

4. Anexos

I. Anexo Controlador Difuso

%% FMEN

```
clear all; close all; clc

% FUNCIONES DE MEMBRESÍA (MOTOR 222049)
x1=-13:0.1:13;

x1_EAN= trapmf(x1,[-13 -13 -4.33 0]);
x1_N = trimf(x1,[-4.33 0 4.33]);
```

```
x1_EAP =trapmf(x1,[0 4.33 13 13]);

figure(1)
subplot(311); plot(x1,x1_EAN,'b',x1,x1_N,'g',x1,x1_EAP,'r');
grid on; title('x1 = Error Angular'); axis([-13 13 0 1]);
legend('Alto Negativo','Neutral','Alto Positivo');


% FM Velocidad Angular
x2 = -600:0.1:600;
x2_VAN = trapmf(x2,[-600 -600 -300 0]);
x2_N = trimf(x2,[-300 0 300]);
x2_VAP = trapmf(x2,[0 300 600 600]);

subplot(312); plot(x2,x2_VAN,'b',x2,x2_N,'g',x2,x2_VAP,'r');
grid on; title('x2 = Velocidad Angular'); axis([-600 600 0 1]);
legend('Alto Negativo','Neutral','Alto Positivo');

% FM Ley de control
u = -12:0.1:12;

u_BN = trimf(u,[-5 -3 0]*1);
u_P= trimf(u,[-3 0 3]*1);
u_BP= trimf(u,[0 3 5]*1);

subplot(313);
plot(u,u_BN,'b',u,u_P,'g',u,u_BP,'k');
grid on; title('u = Voltaje de Salida'); axis([-12 12 0 1]);
legend('Bajo Negativo','Para','Bajo Positivo');

save FM_Motor
```

%% FAM

```
clear all; close all; clc

load FM_Motor
%Condiciones iniciales
x1o = -6; %Error en degree
x2o = 0; %Grados por segundo

%Fuzzificacion
x1_EANI=interp1(x1,x1_EAN,x1o);
%error alto negativo interpolado
x1_NI=interp1(x1,x1_N,x1o);
%error neutro interpolado
x1_EAPI=interp1(x1,x1_EAP,x1o);
%error alto positivo interpolado
x2_VANI=interp1(x2,x2_VAN,x2o);
%velocidad alto negativo interpolado
x2_NI=interp1(x2,x2_N,x2o);
%velocidad neutro interpolado
x2_VAPI=interp1(x2,x2_VAP,x2o);

%Reglas difusas (debemos construir las FAM)
r1=min(x1_EANI,x2_VANI);
r2=min(x1_NI,x2_VANI);
r3=min(x1_EAPI,x2_VANI);
r4=min(x1_EANI,x2_NI);
r5=min(x1_NI,x2_NI);
r6=min(x1_EAPI,x2_NI);
r7=min(x1_EANI,x2_VAPI);
r8=min(x1_NI,x2_VAPI);
```

```
r9=min(x1_EAPI,x2_VAPI);
%FAM
FAM=[r1 r2 r3
   r4 r5 r6
   r7 r8 r9];

%Conjuntos Difusos de salida
mu_BN = [r1 r2 r3 r4];
mu_P = [r5];
mu_BP = [r6 r7 r8 r9];

% Union
%mu_AN = max(mu_AN);
mu_BN = max(mu_BN);
mu_P = max(mu_P);
mu_BP = max(mu_BP);
%mu_AP = max(mu_AP);

% alfa-cut values
%mu_AN = mu_AN*ones(size(u_AN));
mu_BN = mu_BN*ones(size(u_BN));
mu_P = mu_P*ones(size(u_P));
mu_BP = mu_BP*ones(size(u_BP));
%mu_AP = mu_AP*ones(size(u_AP));

figure(1)
subplot(221);
plot(u,u_BN,'g',u,u_P,'k',u,u_BP,'m',u ,mu_BN,'g--',u,mu_P,'k--',u,mu_BP,'m--')
title({'Salidas Difusas de 3 funciones';'de membresía con Alfa-valores'})
grid on; axis([-12 12 0 1]);

% Interseccion
%mu_AN=min(mu_AN,u_AN);
mu_BN=min(mu_BN,u_BN);
mu_BP=min(mu_BP,u_BP);
mu_P=min(mu_P,u_P);
%mu_AP=min(mu_AP,u_AP);

subplot(222);
plot(u,mu_BN,'g',u,mu_P,'k',u,mu_BP,'m')
title({'Salida de corte con';'3 funciones de membresía'})
grid on; axis([-12 12 0 1]);

% Union
muZ=max([mu_BN;mu_P;mu_BP]);

subplot(223);
plot(u,muZ,'b'); grid on; axis([-12 12 0 1])
title('u = Voltaje del motor')

%Defuzzyficación
Z = defuzz(u,muZ,'centroid')
subplot(224);
plot(u,muZ,'b',[Z Z],[0 1], 'r');
grid on; axis([-12 12 0 1])
title({'u = Salida del motor con 3 funciones';'de membresía con centroide'})
mu_P=min(mu_P, u_P);
```

%% MotorControl

```
close all; clear all; clc

% constantes motor 222049
```

```
LA = 0.154*10^(-3);
RA = 3.18;
KT = 15.1*10^(-3);
KB = 1/(634*(pi)/30);
J = 4.22*10^(-7);
TN =12.4*10^(-3);
WN = 5870*(pi)/30;
B = 0.01*TN/WN;


AP = [0 1 0
     0 -B/J KT/J
     0 -KB/LA -RA/LA];
BP = [0; 0; 1/LA];
CP = [1 0 0];
DP = [0];

% Initial Conditions.
x1 = 0; %position
x2 = 0; %speed
x3 = 0; %current
x = [x1 x2 x3]';
u = 0;
r = 5;    % set point
ti=0;   dt = 0.00001;   tf=0.3;

% Simulation:
k = 1;
for t = ti:dt:tf
   X1(k,1) = x(1,1); %POSITION
   X2(k,1) = x(2,1); %SPEED
   X3(k,1) = x(3,1); %CURRENT
   U(k,1) = u; %VOLTAGE
   T(k,1)=t;
   % System
   xp = AP*x + BP*u; % + Wi*r;
   % Control
   err=r-x(1,1);%error
   Error(k,1)=err;
   u=motorfuzz(err,x(2,1)); %motorfuzz
   % Integration
   x = x + xp*dt;
   k = k + 1;
end

figure(1)
subplot(321); plot(T,X3,'b')
title('Control Difuso del Motor DC')
grid on; ylabel('X3 Current')

subplot(322); plot(T,X2,'b')
grid on; ylabel('X2 Velocidad')

subplot(323); plot(T,X1,'b')
grid on; ylabel('X1 Posicion')

subplot(324); plot(T,U,'b')
grid on; ylabel('Voltaje')

subplot(325); plot(T,Error,'b')
grid on; ylabel('Error')

k = 1; k1 = 1;
```

```
for x1 = -5:1:5;
   P(k,1) = x1;
   for x2 = -700:1:700;
      u = motorfuzz(x1,x2);
      V(k1,1) = x2;
      F(k,k1) = u;
      k1 = k1 + 1;
   end
   k = k+1;
end

% Gráfica de Superficie
save scpf P V F

figure(2);
surf(V,P,F)
shading interp
xlabel('Velocidad')
ylabel('Posicion')
zlabel('Salida de Control')
title('Control Surface')
```

%% MotorSistm

```
% MOTORMAXON

clear all; close all; clc
load FM_MOTOR

% constantes motor 222049
LA = 0.154*10^(-3);
RA = 3.18;
KT = 15.1*10^(-3);
KB = 1/(634*(pi)/30);
J = 4.22*10^(-7);
TN = 12.4*10^(-3);
WN = 5870*(pi)/30;
B = 0.01*TN/WN;

%MODEL
AP = [0 1 0
     0 -B/J KT/J
     0 -KB/LA -RA/LA];
BP = [0; 0; 1/LA];
CP = [1 0 0];
DP = [0];

%Time Definitions
ti=0;   dt=0.00001;    tf=3;

%Initial Conditions
xen=[0.0 0.0 0.0]';
u=1;
k=1;
TORC=0;
kk=0;
kkk=1;
ii=1;
Ureducido=0;

% Simulacion
for tiempo=ti:dt:tf
   u=sin(2*pi*1*tiempo);  %senoidal
```

```
    %u=1*tiempo;           %rampa

    uu(k,1)=u;
    tt(k,1)=tiempo;
    kk=kk+1;

    if(kk==300)    % captura la señal cada kk valores
       Ureducido(ii,1)=u;
       Xreducido(ii,1)=xen(1,1);
       Xreducido2(ii,1)=xen(2,1);
       Xreducido3(ii,1)=xen(3,1);
       kk=0;
       ii=ii+1;
       ttt(kkk,1)=ii;
       kkk=kkk+1;
    end

    % Stable Non-Homogeneous System
    x1en(k,1) = xen(1,1);
    x2en(k,1) = xen(2,1);
    x3en(k,1) = xen(3,1);

    xenp = AP*xen + BP*u + [0; -1/J; 0]*TORC; %System
    yen = CP*xen + DP*u;
    xen = xen + xenp*dt; %Integration
    YP1N(k,1)=yen(1,1);
    k = k+1;
end

tam1=size(Ureducido);

% entradas
X =
[Ureducido/max(Ureducido),Xreducido/max(Xreducido),Xreducido2/max(Xreducido2),Xreducido3/max(Xreduci
do3)];

% salida
D = [Xreducido/max(Xreducido)];

figure(1)
subplot(221);plot(ttt,(Xreducido/max(Xreducido)))
title('Sistema Estable no homogéneo')
grid on; ylabel('Posicion X1')
subplot(222);plot(ttt,(Xreducido2/max(Xreducido2)))
grid on; ylabel('Velocidad X2')
subplot(223);plot(ttt,(Xreducido3/max(Xreducido3)))
grid on; ylabel('Corriente X3')
subplot(224);plot(ttt,(Ureducido/max(Ureducido)))
grid on; ylabel('Entrada')
```

%% motorfuzz

```
function [Z] = motorfuzz(x1o,x2o);
load FM_Motor
%Fuzzification
x1_EANI=interp1(x1,x1_EAN,x1o);

% FM de Error posición angular
x1_NI =interp1(x1,x1_N,x1o);
x1_EAPI=interp1(x1,x1_EAP,x1o);
x2_VANI=interp1(x2,x2_VAN,x2o);

% FM de velocidad angular
```

```
x2_NI =interp1(x2,x2_N,x2o);
x2_VAPI=interp1(x2,x2_VAP,x2o);

%Reglas Difusas (debemos contruir las FAM)
r1 = min(x1_EANI,x2_VANI);
r2 = min(x1_NI,x2_VANI);
r3 = min(x1_EAPI,x2_VANI);
r4 = min(x1_EANI,x2_NI);
r5 = min(x1_NI,x2_NI);
r6 = min(x1_EAPI,x2_NI);
r7 = min(x1_EANI,x2_VAPI);
r8 = min(x1_NI,x2_VAPI);
r9 = min(x1_EAPI,x2_VAPI);

%FAM
FAM=[r1 r2 r3
   r4 r5 r6
   r7 r8 r9];

%Posicion
%mu_AN = [0];
mu_BN = [r1 r2 r3 r4];
mu_P = [r5];
mu_BP = [r6 r7 r8 r9];
%mu_AP = [0];

%Union
%mu_AN = max(mu_AN);
mu_BN = max(mu_BN);
mu_P = max(mu_P);
mu_BP = max(mu_BP);
%mu_AP = max(mu_AP);

%Valores alfa-cut
%mu_AN = mu_AN*ones(size(u_AN));
mu_BN = mu_BN*ones(size(u_BN));
mu_P = mu_P*ones(size(u_P));
mu_BP = mu_BP*ones(size(u_BP));
%mu_AP = mu_AP*ones(size(u_AP));

% Intersección
mu_BN = min(mu_BN,u_BN);
mu_P = min(mu_P,u_P);
mu_BP = min(mu_BP,u_BP);

% Union
muZ = max([mu_BN;mu_P;mu_BP]);

%Defuzzyfication
Z = defuzz(u,muZ,'centroid');
```

II. Anexo Robot 2 eslabones

```
close all, clc, clear all;

% Letra T
G1 = [0.30 0.20 0.25 0.25];
G2 = [0.30 0.30 0.30 0.20];
movement(G1,G2);

% ROBOT PLOTTER
function movement(POSX,POSY)
m1 = 0.0644;
```

```
m2 = 0.0644;
l1 = 0.25;
l2 = 0.25;
g = 9.81;
Xi = POSX(1);
Yi = POSY(1);
k = 1;
TT = 0;
for pts = 1:(length(POSX)-1)
   % Inverse Cinematic
   [q1m1,q2m1]=Cinv(Xi,Yi,l1,l2);
   [q1m2,q2m2]=Cinv(POSX(pts+1),POSY(pts+1),l1,l2);
   p0 = [q1m1 q2m1]';
   pf = [q1m2 q2m2]';
   pfp =[0 0]';
   t0 = 0;
   dt = 0.005;
   x = [p0' pfp' 0 0]';
   Txpts=[Xi POSX(pts+1)];
   Typts=[Yi POSY(pts+1)];
   [~,R1,V1,A1] = PLOTTER(false,50,Txpts,Typts,l1,l2,dt,14);
   QD = R1;
   QDP = V1;
   QDPP = A1;
   T = 0:(max(size(QD))-1);
   T = T'*dt;
   TT = [TT; T+TT(end,1)];
   qd = QD(:,1:2)';
   qdp = QDP(:,1:2)';
   qdpp = QDPP(:,1:2)';
   % Controller Parameters
   Kp = 500*eye(2);
   Kv = 200*eye(2);
   Ki = 10*eye(2);
   kk = 1;
   for t=0:dt:(max(T))
      q1 = x(1) ; q2 = x(2);
      q1p = x(3); q2p = x(4);
      % Following Errors
      e = qd(:,kk)-x(1:2);    % Planned Trajectory - Actual
      ep = qdp(:,kk)-x(3:4);    % Planned Speed - Actual Speed
      % Dinamical Model Robot Matrix Calculation
      % Inercial Matrix
      M = [(m2*(2*l1^2+2*cos(q2)*l1*l2+l2^2/2))/2+(l1^2*m1)/4
(l2*m2*(l2+2*l1*cos(q2)))/4;(l2*m2*(l2+2*l1*cos(q2)))/4 (l2^2*m2)/4];
      % Coriolis Matriz
      C = [-(l1*l2*m2*q2p*sin(q2))/2 -(l1*l2*m2*sin(q2)*(q1p+q2p))/2;(l1*l2*m2*q1p*sin(q2))/2 0];
      N = C*[q1p q2p]';
      % Gravity Vector
      G = [g*m2*((l2*cos(q1+q2))/2+l1*cos(q1))+(g*l1*m1*cos(q1))/2;(g*l2*m2*cos(q1 + q2))/2];
      % Control Torque
      S = qdpp(:,kk) + Kv*ep + Kp*e+ Ki*x(5:6);
      tau = M*S+N+G;
      % Space State
      xp = [x(3:4); inv(M)*(-N-G+tau);e];
      % Integration
      x = x + xp*dt;
      % Direct Cinematic
      [T1,T2] = Cdir(x(1),x(2),l1,l2);
      x1(k,1) = T1(1,4);
      y1(k,1) = T1(2,4);
      x2(k,1) = T2(1,4);
      y2(k,1) = T2(2,4);
```

```
        [~,IT2] = Cdir(QD(kk,1),QD(kk,2),l1,l2);
        ix2(k,1) = IT2(1,4);
        iy2(k,1) = IT2(2,4);
        kk = kk + 1;
        k = k + 1;
    end
    Xi = x2(end,1);
    Yi = y2(end,1);
end
TT = TT(2:end);
% SIMULATION
for i = 1:floor(length(x1)/10):length(x1)
    figure(1)
    plot(ix2(:,1),iy2(:,1),'r--');hold on;
    plot(0,0,'kO','LineWidth',3);
    plot([-(l1*2.5) (l1*2.5)],[0 0],'k-','LineWidth',3);
    plot(x1(i,1),y1(i,1),'kO','LineWidth',3);
    plot(x2(1:i,1),y2(1:i,1),'b');
    plot([0 x1(i,1) x2(i,1)],[0 y1(i,1) y2(i,1)],'k','LineWidth',3);hold off;
    title("Letra T - Fiorela Manco");grid minor;
    axis([-(l1*2.5) l1*2.5 -0.1 (l1*2.5)-0.1]);
    pause(0.2)
end
figure(1);
plot(ix2(:,1),iy2(:,1),'r--');hold on;
plot(0,0,'kO','LineWidth',3);
plot([-(l1*2.5) (l1*2.5)],[0 0],'k-','LineWidth',3);
plot(x1(i,1),y1(i,1),'kO','LineWidth',3);
plot(x2(:,1),y2(:,1),'b');
plot([0 x1(end,1) x2(end,1)],[0 y1(end,1) y2(end,1)],'k','LineWidth',3);hold off;
title("Letra T - Fiorela Manco");grid minor;
axis([-(l1*2.5) l1*2.5 -0.1 (l1*2.5)-0.1]);
end
function [T,R,V,A] = PLOTTER(conti,npts,px,py,l1,l2,dt,time)
    POSX = linspace(px(1),px(2),npts);
    POSY = linspace(py(1),py(2),npts);
    i=1;
    for pts=1:npts
        [q1(i,1),q2(i,1)]=Cinv(POSX(pts),POSY(pts),l1,l2);
        i=i+1;
    end
    [T1,R1,V1,A1] = MovC(npts,q1,dt,time,conti);
    [T2,R2,V2,A2] = MovC(npts,q2,dt,time,conti);
    k = max([length(T1) length(T2)]);
    R = ones(k,2);
    R(:,1) = R1(length(R1),1); R(:,2) = R2(length(R2),1);
    R(1:length(R1),1)=R1; R(1:length(R2),2)=R2;
    V = zeros(k,2);
    V(1:length(V1),1)=V1; V(1:length(V2),2)=V2;
    A = zeros(k,2);
    A(1:length(A1),1)=A1; A(1:length(A2),2)=A2;
    T = zeros(k,2);
    T(1:length(T1),1)=T1; T(1:length(T2),2)=T2;
end
function [T1,R1,V1,A1] = MovC(npts,pos,dt,tf,conti)
    th0=pos(1);
    thf=pos(end);
    t0 = 0;
    time = linspace(t0,tf,npts);
    k = 1;
    if conti
        [a] = MatrixCon(th0,0,thf,0,0,tf);
        vel = a(2)+2*a(3)*time+3*a(4)*time.^2;
```

```
    else
        vel = zeros(1,npts);
    end
    for i = 1:(length(time)-1)
        [a] = MatrixCon(pos(i),vel(i),pos(i+1),vel(i+1),time(i),time(i+1));
        for t =time(i):dt:(time(i+1)-dt)
            T(k,1) = t;
            Pfuncion1 = a(1) + a(2)*t + a(3)*t^2 + a(4)*t^3;
            Pf(k,1)=Pfuncion1;
            Vfuncion1 = a(2) + 2*a(3)*t + 3*a(4)*t^2;
            Vf(k,1)=Vfuncion1;
            Afuncion1 = 2*a(3) + 6*a(4)*t;
            Af(k,1)=Afuncion1;
            k=k+1;
        end
    end
    T1=T;R1=Pf;V1=Vf;A1=Af;
end
function [a] = MatrixCon(q0,v0,q1,v1,t0,tf)
    M = [ 1 t0 t0^2 t0^3;
          0 1 2*t0 3*t0^2;
          1 tf tf^2 tf^3;
          0 1 2*tf 3*tf^2];
    b = [q0;v0;q1;v1];
    a = inv(M)*b;
end
function [T1,T2] = Cdir(q1,q2,l1,l2)
    DH = [l1 0 0  q1;
          l2 0 0  q2];
    A1 = matra(DH(1,1),DH(1,2),DH(1,3),DH(1,4));
    A2 = matra(DH(2,1),DH(2,2),DH(2,3),DH(2,4));
    T1 = A1;
    T2 = T1*A2;
end
function [q1,q2] = Cinv(x,y,l1,l2)
    cosq2 = (x^2+y^2-l1^2-l2^2)/(2*l1*l2);
    q2 = atan2(-(sqrt(1 - cosq2^2)),cosq2);
    beta = atan2(y,x);
    alpha = atan2((l2*sin(q2)),(l1+l2*cos(q2)));
    q1 = beta-alpha;
end
function [A] = matra(a,d,a1,th)
A = [ cos(th) -cos(a1)*sin(th)  sin(a1)*sin(th) a*cos(th);
    sin(th)  cos(a1)*cos(th) -sin(a1)*cos(th) a*sin(th);
    0 sin(a1) cos(a1) d;
    0 0 0 1];
end
```

III. Anexo Red Neuronal

```
close all; clc; clear all;
ruta = "C:\Users\Fiorela\Documents\MATLAB\Robótica e IA codes\Ex Final - Manco\Red Neuronal\imagenes\";

% IMÁGENES DE APRENDIZAJE
A1 = imread(ruta+"A1.png");
%A1r = imresize(A1, 1.3);
A1BE = edge(A1(:,:,1), 'Canny');

B1 = imread(ruta+"B1.png");
%B1r = imresize(B1, 1.3);
B1BE = edge(B1(:,:,1), 'Canny');

personajes = ["Shusei Kagari" "Akane Tsunemori"];
```

```
figure(1);
subplot(121); imshow(A1); title(personajes(1));
subplot(122); imshow(B1); title(personajes(2));

% ENTRADA
global ndim;
ndim = 269;
X = zeros(ndim, ndim, 2);

%%
% AJUNTAR IMAGS A ENTRADA
X(:,:,1) = A1BE(1:ndim,1:ndim);
X(:,:,2) = B1BE(1:ndim,1:ndim);

D = diag([1 1]);    %respuesta deseada

% PESOS ENTRE NEURONAS
W1 = 2*rand(4, ndim^2) - 1;
W2 = 2*rand(2, 4) - 1;
W3 = 2*rand(1, 2) - 1;
W4 = 2*rand(2, 1) - 1;

% ENTRENAMIENTO DE LA RED NEURONAL
for epoch = 1:10000
   [W1, W2, W3, W4] = DeepDropout(W1, W2, W3, W4, X, D);
end
%%
%Resize de imágenes
A2 = imread(ruta+"A5.png");
A2r = imresize(A2, 0.6);
A2b = edge(A2r(:,:,1), 'Canny');

A3 = imread(ruta+"A6.png");
A3r = imresize(A3, 1.2);
A3b = edge(A3r(:,:,1), 'Canny');

B2 = imread(ruta+"B5.png");
B2r = imresize(B2, 0.45);
B2b = edge(B2r(:,:,1), 'Canny');

%B3 = imread(ruta+"B3.png");
B3 = imread(ruta+"B4.png");
B3r = imresize(B3, 0.4);
B3b = edge(B3r(:,:,1), 'Canny');
%%
X(:,:,1) = A2b(1:ndim,1:ndim);       %269 x 269
X(:,:,2) = A3b(1:ndim,1:ndim);
X(:,:,3) = B2b(1:ndim,1:ndim);
X(:,:,4) = B2b(1:ndim,1:ndim);
%X(:,:,4) = B3b(1:269,200:468);

% Guarda los resultados
salida_list = ["" "" "" ""];

%CALCULO DE RESPUESTA
N = 4;
for k = 1:N
  x  = reshape(X(:, :, k), ndim^2, 1);
  v1 = W1*x;
  y1 = Sigmoid(v1);

  v2 = W2*y1;
```

```
  y2 = Sigmoid(v2);

  v3 = W3*y2;
  y3 = Sigmoid(v3);

  v = W4*y3;
  y = Softmax(v);
  AA = round(y);
  SALIDA = personajes(find(AA == 1))
  salida_list(k) = SALIDA;
end



imgs = {A2 A3 B2 B3};

% SUBPLOT MUESTRA RESULTADOS
for i =1:4
    figure(2);
    subplot(1,4,i);
    imshow(imgs{1,i});
    title(salida_list(i));
end

%% FUNCIONES
function [W1, W2, W3, W4] = DeepDropout(W1, W2, W3,  W4, X, D)
alpha = 0.01;
N = 2;
for k = 1:N
    x = reshape(X(:, :, k), 269*269, 1);
    v1 = W1*x;
    y1 = Sigmoid(v1);
    y1 = y1 .* Dropout(y1, 0.1);
    v2 = W2*y1;
    y2 = Sigmoid(v2);
    y2 = y2 .* Dropout(y2, 0.1);
    v3 = W3*y2;
    y3 = Sigmoid(v3);
    y3 = y3 .* Dropout(y3, 0.1);
    v = W4*y3;
    y = Softmax(v);
    d = D(k, :)';
    e = d - y;
    delta = e;
    e3 = W4'*delta;
    delta3 = y3.*(1-y3).*e3;
    e2 = W3'*delta3;
    delta2 = y2.*(1-y2).*e2;
    e1 = W2'*delta2;
    delta1 = y1.*(1-y1).*e1;
    dW4 = alpha*delta*y3';
    W4 = W4 + dW4;
    dW3 = alpha*delta3*y2';
    W3 = W3 + dW3;
    dW2 = alpha*delta2*y1';
    W2 = W2 + dW2;
    dW1 = alpha*delta1*x';
    W1 = W1 + dW1;
end
end
function ym = Dropout(y, ratio)
  [m, n] = size(y);
  ym     = zeros(m, n);
```

```
  num     = round(m*n*(1-ratio));
  idx     = randperm(m*n, num);
  ym(idx) = 1 / (1-ratio);
end
function y = Sigmoid(x)
  y = 1 ./ (1 + exp(-x));
end
function y = Softmax(x)
  ex = exp(x);
  y  = ex / sum(ex);
end
```