**Guayaquil, 24 de abril de 2018**

**INFORME PRUEBA DE ALGORITMO EN MATLAB**

El presente informe tiene como objetivo dar a conocer los puntos más importantes sobre la prueba de algoritmo en Matlab, la misma que muestra la simulación en Matlab del control proporcional derivativo (PD) desarrollado por Takegaki y Arimoto, la función de control es aplicada al modelo de un robot de dos grados de libertad. El robot manipulador debe describir una trayectoria circular dentro de su área de movimiento.

Los parámetros de **Denavit Hartenberg** son cuatro (d, a, α, θ) estos dependen exclusivamente de la geometría de cada eslabón y la relación de las articulaciones que lo conectan con el anterior y el siguiente eslabón. Estos se pueden clasificar en dos grupos los que son referidos al tamaño y forma del eslabón y los que son referidos a un eslabón y a su antecesor.

**Parámetros relativos al tamaño y forma del eslabón**

Parámetro **a**: este parámetro es el referente a la distancia entre los ejes **i** e **i-1** este parámetro es el encargado de definir la longitud del eslabón el signo de este ángulo lo da la regla de la mano derecha.

Parámetro **α**: este parámetro es el que sería formado entre los ejes **i** e **i-1** de las articulaciones. Con este parámetro de cierto modo se puede medir la forma del eslabón ya que se mide respecto al ángulo que sobre el mismo se encuentra girado, por esto se le conoce como el parámetro o ángulo de torsión del eslabón.

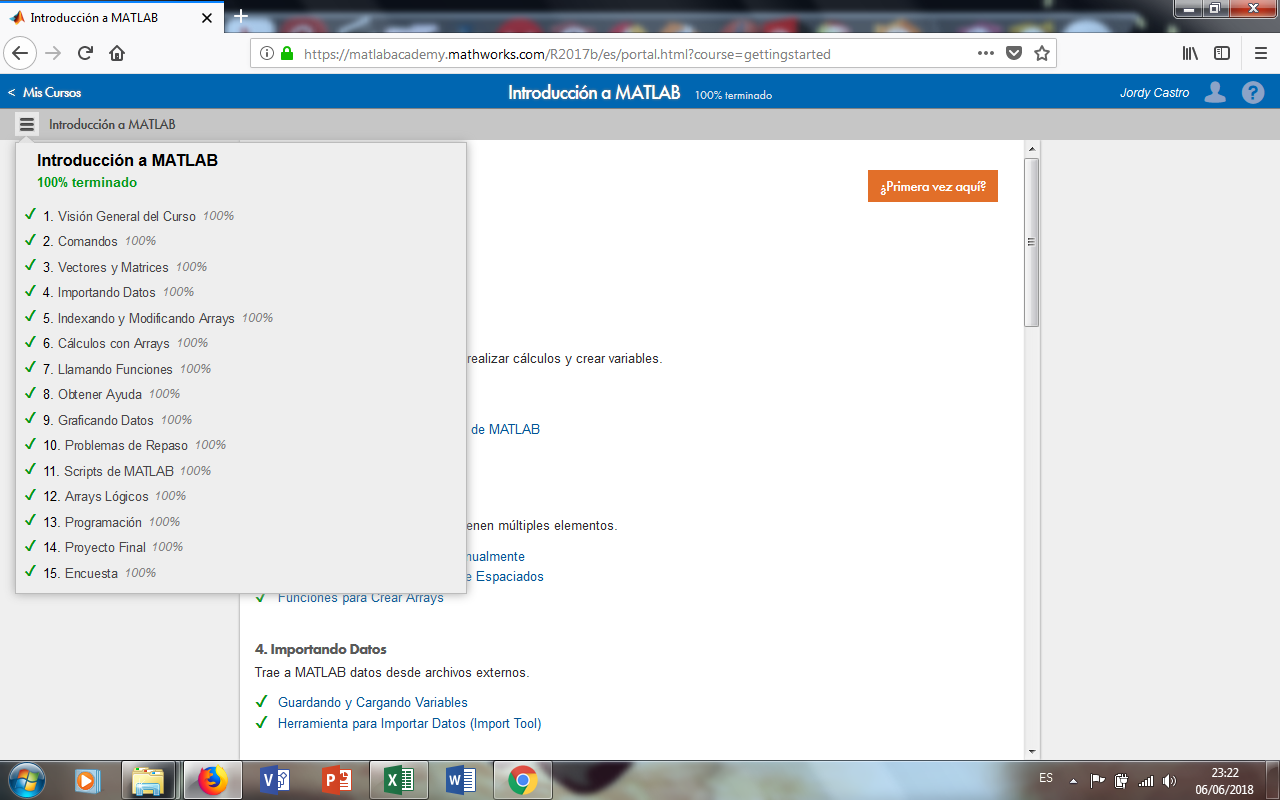
**Parámetros relativos a un eslabón y a su antecesor**

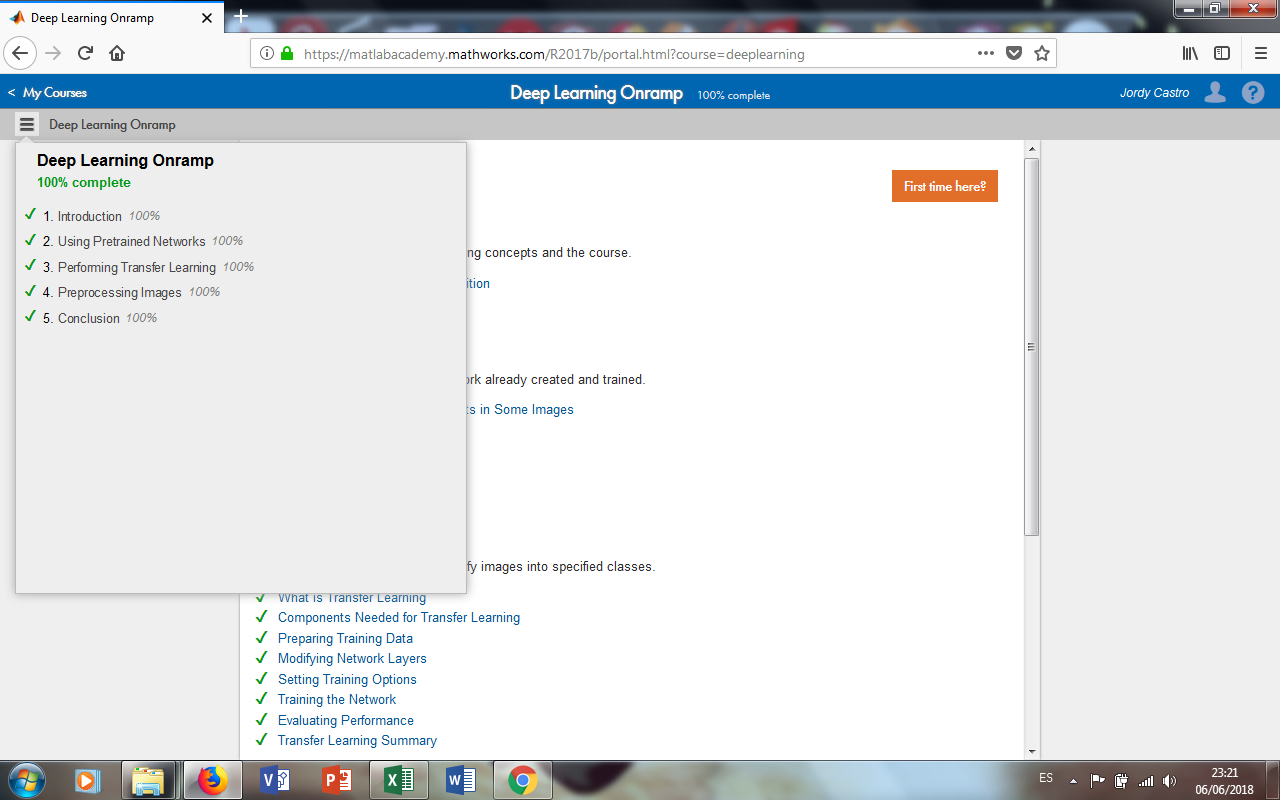
Parámetro **d**: es la distancia tomada desde el origen de un sistema articulación eslabón hasta la intersección del eje **Z** y **X** del sistema anterior, esta medición determina en cierto modo la distancia entre dos eslabones determinados por el tamaño y la forma de la articulación, también se llama longitud articular.

Parámetro **θ**: es el ángulo formado por el eje **x** con respecto al eje **X-1** ósea el eje del sistema inmediatamente anterior expresa de alguna forma el ángulo formado entre dos articulaciones también se llama ángulo articular y su signo viene dado por la regla de la mano derecha.

Se define como **cinemática directa** a una técnica en Software de gráficos para calcular la posición de partes de una estructura articulada a partir de sus componentes fijas y las transformaciones inducidas por las articulaciones d la estructura.

Se define la **cinemática inversa** en encontrar valores que deben adoptar las coordenadas articulares del robot para que su extremo se posición y oriente según una determinada localización espacial.

Si antes mencionar que realicé un curso online acerca del uso de la herramienta Matlab, tal y como se puede visualizar en las siguientes imágenes.



A continuación, podemos observar el desarrollo del código en Matlab para una mejor comprensión.

**Código 1: Código el Matlab que permite graficar trayectoria de un robot manipulador de dos grados de libertad**

clc;

clear all;

close all;

format short

syms q1 q2 beta1 beta2 l1 l2 alpha1 alpha2 real

H20=H\_r2gdl();

disp('Transformación homogénea del robot 2gdl');

disp(H20);

[R20, cinemat\_r2gdl,cero, c]=H\_DH(H20);

disp('Matriz de rotación'); disp(R20);

disp('cinemática directa');

disp(cinemat\_r2gdl);

[x0, y0, z0] = cinematica\_r2gdl(beta1,l1,q1,beta2,l2,q2);

jac\_r2gdl=jacobian([x0; y0], [q1;q2]);

det\_r2gdl=simplify(det(jac\_r2gdl));

% det[J]=l\_1l\_2 sin(q\_2)

%ejemplo numérico

%t=0:0.0001:4.7122;

t=0:0.0001:6.2830;

%parámetros del círculo:[x\_c,y\_c]'=[0.3,-0.3]' y radio r=0.2

xc=0.3; yc=-0.3; r=0.20;

l1=0.45; l2=0.45;

beta1=0.1; beta2=0.1;

q1=[]; q2=[];

% ecuación del círculo

x=xc+r\*sin(t);

%x=xc+r\*sin(2\*t);

%y=yc+r\*cos(t);

y=yc+r\*cos(t)+r\*sin(t);

% cinemática inversa

[q1,q2]=cinv\_r2gdl(l1,l2,x,y);

%coordenas cartesianas del extremo final del robot de 2 gdl

[x0, y0, z0] = cinematica\_r2gdl(beta1,l1,q1,beta2,l2, q2);

figure

plot(x0,y0,'LineWidth',2,'color',[0.8,0.1,0.1])

El código 1, hace llamadas a otras funciones, las cuales son:

**Código 2: función H\_r2gdl precisa los parámetros Denavit-Hartenberg del manipulador de dos grados de libertad.**

function H=H\_r2gdl()

syms q1 q2 beta1 beta2 l1 l2 alpha1 alpha2 real

disp('Parámetros Denavit-Hartenberg del robot planar vertical de 2 gdl')

disp('[ l alpha d q]')

dh=[l1, 0, beta1, q1; l2, 0, beta2, q2];

disp(dh)

%$H10=HDH{0}{q\_1}{beta\_1}{l\_1}{0}

H10=HRz(q1)\*HTz(beta1)\*HTx(l1)\*HRx(0);

%$H21=HDH{0}{q\_2}{beta\_2}{l\_2}{0}

H21=HRz(q2)\*HTz(beta2)\*HTx(l2)\*HRx(0);

H20=simplify(H10\*H21); %H20=H10 H21

[R20, cinemat\_r2gdl, cero, c]=H\_DH(H20);

H=[R20, cinemat\_r2gdl; %R20(q\_1,q\_2),f\_R(q\_1,q\_2)

cero, c];

end

**Código 3: Función cinv\_r2gdl son ecuaciones de la cinemática inversa.**

function [q1,q2]=cinv\_r2gdl(l1,l2,x0,y0)

q2=acos((x0.\*x0+y0.\*y0-l1\*l1-l2\*l2)/(2\*l1\*l2));

q1=atan(y0./x0)-atan((l2\*sin(q2))./(l1+l2\*cos(q2)));

end

**Código 4: Función cinematica\_r2gdl son las ecuaciones de la cinemática directa del manipulador de dos grados de libertad.**

function [x0, y0, z0]=cinematica\_r2gdl(beta1,l1,q1,beta2,l2,q2)

dato1=whos('beta1'); dato2=whos('l1');

dato3=whos('q1');

dato4=whos('beta2'); dato5=whos('l2');

dato6=whos('q2');

v1=strcmp(dato1.class, 'sym');

v2=strcmp(dato2.class, 'sym');

v3=strcmp(dato3.class, 'sym');

v4=strcmp(dato4.class, 'sym');

v5=strcmp(dato5.class, 'sym');

v6=strcmp(dato6.class, 'sym');

digits(3);

if (v1 & v2 & v3 & v4 & v5 & v6) %caso simbólico

x0=simplify(vpa(l1\*cos(q1)+l2\*cos(q1+q2),3));

y0=simplify(vpa(l1\*sin(q1)+l2\*sin(q1+q2),3));

z0=vpa(beta1+beta2,3);

else %caso numérico

x0=l1\*cos(q1)+l2\*cos(q1+q2);

y0=l1\*sin(q1)+l2\*sin(q1+q2);

z0=beta1+beta2;

end

end

Cálculo de la variable Theta

function RHz=HRz(theta)

dato=whos('theta');

if strcmp(dato.class, 'sym') %variables simbólicas

RHz=[cos(theta), -sin(theta), 0, 0;

sin(theta), cos(theta), 0, 0;

0, 0, 1, 0;

0, 0, 0, 1];

else

digits(3); %cálculos numéricos

RHz=round([ vpa(cos(theta),3), vpa(-sin(theta),3), 0, 0;

vpa(sin(theta),3),vpa(cos(theta),3), 0, 0;

0, 0, 1, 0;

0, 0, 0, 1]);

end

end

Función HTz para el cálculo de la variable simbólica d.

function Tz=HTz(d)

Tz=[ 1 0 0 0; 0 1 0 0;

0 0 1 d; 0 0 0 1];

end

Función HTx para el cálculo de la variable simbólica d.

function Tx=HTx(d)

Tx=[ 1 0 0 d; 0 1 0 0;

0 0 1 0; 0 0 0 1];

end

Función HTy para el cálculo de la variable simbólica d.

function Tz=HTy(d)

Tz=[ 1 0 0 0; 0 1 0 d;

0 0 1 0; 0 0 0 1];

end

Función HRx para el cálculo de la variable simbólica theta.

function RHx=HRx(theta)

dato=whos('theta');

if strcmp(dato.class, 'sym') %variables simbólicas

RHx=[1, 0, 0, 0;

0, cos(theta), -sin(theta), 0;

0, sin(theta), cos(theta), 0;

0, 0, 0, 1];

else

digits(3); %cálculos numéricos

RHx=round([1, 0, 0, 0;

0, vpa(cos(theta),3), vpa(-sin(theta),3), 0;

0, vpa(sin(theta),3),vpa(cos(theta),3), 0;

0, 0, 0, 1]);

end

end

Función HRy para el cálculo de la variable simbólica theta.

function RHy=HRy(theta)

dato=whos('theta');

if strcmp(dato.class, 'sym') %variables simbólicas

RHy=[cos(theta), 0, sin(theta), 0;

0, 1, 0, 0;

-sin(theta), 0, cos(theta), 0;

0, 0, 0, 1];

else

digits(3); %cálculos numéricos

RHy=round([ vpa(cos(theta),3), 0, vpa(sin(theta),3), 0;

0, 1, 0, 0;

vpa(-sin(theta),3), 0, vpa(cos(theta),3), 0;

0, 0, 0, 1]);

end

end

Función H\_DH para el cálculo de la matriz de transformación homogénea.

function [R vect\_d vect\_cero c]=H\_DH(H)

for i=1:3

for j=1:3 R(i,j)=H(i,j);

end

end

%estructura de la matriz de transformación homogénea

vect\_d=[H(1,4); H(2,4); H(3,4)];

vect\_cero=[0;0;0]';

c=1;

end

**RESULTADOS DE LAS PRUEBAS**

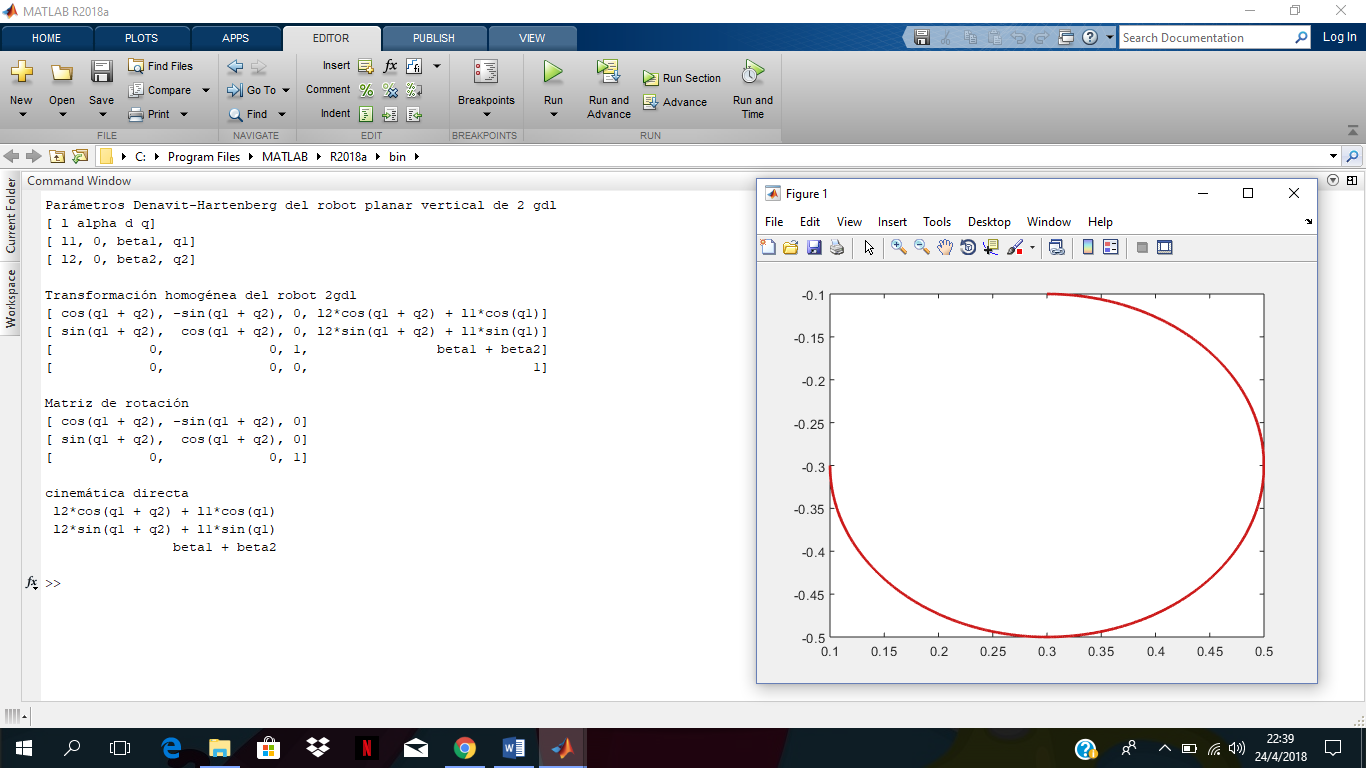


Figura 1: Resultados en forma matricial y gráfica.

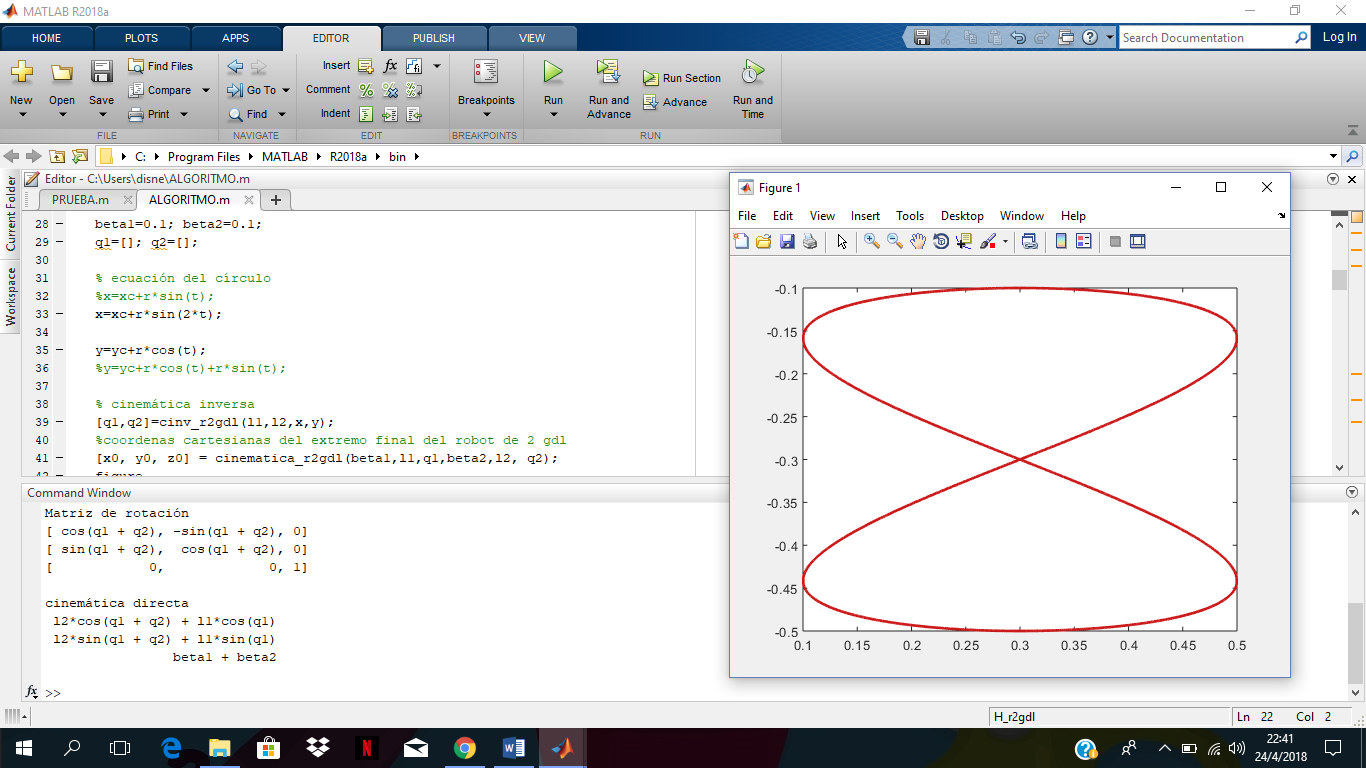


Figura 2: grafico en forma de 8, con el siguiente comando x=xc+r\*sin(2\*t);

Doy por finalizado este informe, esperando se haga uso del que se estime conveniente.

Atentamente,

Jordy Andrés Castro López

Practicante Comunitarias Proyecto FCI