Modelado y Control Backstepping de un Brazo Robótico de 6 GDL

Santiago Madariaga Collado
Departamento de Ingeniería Electrónica
Universidad de Ingeniería y Tecnología, Barranco 15063, Lima, Lima, Perú
E-mail: santiago.madariaga@utec.edu.pe

***Abstract*—En el presente informe se presenta el modelado y control backstepping de un brazo robotico de 6 grados de libertad no lineal de un robot de 6 grados de libertad. Para el modelado se utilizaron las ecuaciones de Lagrange para el modelado no lineal. En este paper se propone 1 controladores en este proyecto, un controlador backstepping. Para diseñar el sistema de control , empleando para ello la representactión en el espacio de estado del brazo robotico, así como también el algoritmo de control backstepping. Estos estudios de simulación demuestran que las 6 posiciones angulares del brazo robotico controladomsiguen simultáneamente a sus señalesde referencia para el caso de regulación (referencias constantes), y para el caso de seguimiento (señales cambiantes arbritrarias). Cumoliendo los parámetros preestablecidos.**

***Index Terms*—modelado dinámico brazo robótico, control backstepping, simulación no lineal, método de Euler-Lagrange.**

I. Introducción

En el presente trabajo, el modelo dinámico MIMO de un robot de 6 GDL es obtenido usando el método de las ecuaciones de Lagrange. Para el control del brazo robótico, se diseñó un controlador con el método de Backstepping. Este controlador esta basado en el modelo del brazo debido a que el diseño de este controlador requiere el modelo dinámico del brazo robotico.

El objectivo del sismta de control es diseñar un vector de contrul u que haga a las 6 salidas del controlador seguir las trayectorias mostradas, en el modo de seguimiento, y alcanzar el valor de referencia cuando las señales de referencia son constantes. Se utilizó el código de un controlador backstepping diseñado para un brazo robot de 3 GDL como punto de partida[1]. Además, el modelo del sistema parte de trabajos previos de brazos de 3 GDL[2]. Se utilizó el modo simbólico de MATLAB para hallar los valores de las matrices y realizar la simulación del controlador backstepping.

II. Modelado del Brazo Robótico de 6GDL

La dinámica del robot se ocupa de la relación entre las fuerzas que actúan sobre un cuerpo y el movimiento que en él se origina. Por tanto, el modelo dinámico de un robot tiene por objetivo conocer la relación entre el movimiento del robot y las fuerzas implicadas en el mismo.

Esta relación se obtiene mediante el denominado modelo dinámico, que establece la relación matemática entre las entradas (señal de control) y las salidas (ángulos).

El modelo extraído de [1] consiste de un brazo robótico de 3GDL, este modelo se expandió para el modelado de un brazo

Fig. 1. Robot industrial de 6GDL[3]

robótico de 6GDL. El modelo del brazo robótico es expresado como:

$$M(q)\dot{+}P(q,\dot{q})\dot{q} + d(q) = u \tag{1}$$

Donde $M(q)\ddot{q}$ es la matriz de inercia, es simétrica 6x6. El término $P(q,\dot{q})\dot{q}$ incluye las fuerzas centrípetas y de Coriolis. Por último, el vector 'd' contiene la gravedad y demás términos que actuan en las uniones del brazo robótico.

Para obtener al modelo en la forma matricial (1) se debe de unir la parte mecánica y eléctrica. Es decir, las ecuaciones dinámicas que describen a los ángulos (que son las salidas del sistema a controlar) en términos del modelo mecánico del sistema (velocidad y energía potencial), deben relacionarse con las entradas del sistema, las cuales son los voltajes. Esto se logra utilizando las ecuaciones de lagrange, las cuales relacionan torques y energías.

$$L = V - Ut_i = \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} \tag{2}$$

Donde V y P son la sumatoria de energías cinéticas y potenciales respectivamente.

Los valores de los parámetros se tomaron de [2]. A continuación se muestra la tabla extraída de la misma referencia.

Símbolo	Descripcióon	Valor	Unidades
u1, u2, u3	Voltaje de entrada a cada actuador		V
u	Voltaje de entrada a un actuador		V
KA	Ganancia del amplificador	8.5	
Va	Voltaje de armadura		V
Ra	Resistencia de armadura	2.49	■
La	Inductancia de armadura	2.63×103	H
ia	Corriente de armadura	A	
Km	Constante del torque motor	0.0458	N-m/A
Tm	Torque motor		N-m
L	Torque de carga		N-m
Tg1	Torque de entrada a los engranajes		N-m
Tg2	Torque de salida de los engranajes		N-m
Jm	Inercia del motor	7.1×106	kg-m2
Jg	Inercia de los engranajes	0.053	kg-m2
Ja	Inercia del eje o arbol conectado al motoresductor		kg-m2
Jr	Inercia del rodamiento conectado al arbol		kg-m2
J1	Inercia de la columna		kg-m2
J2	Inercia del brazo		kg-m2
J3	Inercia del antebrazo		kg-m2
Bm	Constante de friccion del motor	0.0001	N-m/rad/s
Bg	Constante de friccion en engranajes	0.01	N-m/rad/s
ma	Masa del eje o arbol conectado al motoreductor	1.5	kg
mr	Masa del rodamiento conectado al arbol	1.48	kg
m1	Masa total de la columna	2.98	kg
m2	Masa del brazo	0.89	kg
m3	Masa del antebrazo	0.25	kg
la	Longitud del eje o arbol conectado al motoreductor	0.3378	m
lr	Longitud del rodamiento y disco conectado al arbol	0.054	m
L1	Longitud total de la columna	0.391	m
L2	Longitud del brazo	0.24	m
L3	Longitud del antebrazo	0.245	m
ra	Radio del eje o arbol conectado al motoreductor	0.015	m
rr	Radio del rodamiento y disco conectado al arbol	0.0427	m
Vb	Voltaje contraelectromotriz		V
Kb	Constante contraelectromotriz	0.0458	V/rad/s
g	Aceleracion de la gravedad	9.81	m/s2
N1, N2	No de dientes de los engranajes (N2 N1)		
n	Relacion de engranajes (n = N2/N1)	65.5	
m	Posicion angular del motor		rad
g	Posicion angular del eje de salida del motor		rad
m	Velocidad angular del motor (m = ng)	rad/s	
g	Velocidad angular de la carga		rad/s
qL	Coordenada articular de un eslabon		rad
q1, q2, q3	Coordenadas articulares para cada eslabon		rad

Tabla 1: Parámetros y variables del brazo robótico de 6GDL

Para hallar la relación entre torques y voltajes, se debe de realizar el análisis eléctrico del motor.

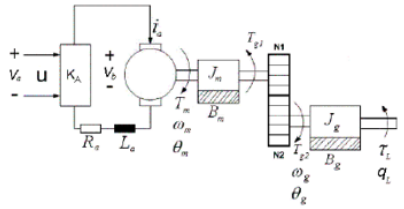

Fig. 2. Diagrama del actuador del brazo robótico

Ecuaciones del actuador:

$$\tau_i = -J_{eq}\ddot{q}_i - (B_{eq} + \frac{n^2 K_m K_b}{R_a}) * \dot{q}_i + \frac{nK_m K_A}{R_a} u_i$$

A. *Modelado del Sistema en MATLAB*

El modelado del sistema se llevó a cabo definiendo los parámetros y las variables del sistema en el modo simbólico de MATLAB, permitiendo inspeccionar las relaciones entre las variables del sistema. El modo simbólico también permite hallar las derivadas parciales y totales, las cuales son necesarias para hallar las ecuaciones de Lagrange.
Código de matlab: q1_d(t) = diff(q1(t))
Salida:

$$\frac{\partial}{\partial t} q_2(t)$$

Esto se realizará con todos los ángulos. En segundo lugar, se definen las coordenadas en el espacio de cada unión del brazo robótico:

```
% Coordenadas

x1 = 0;
y1 = 0;
z1 = (ma*la + mr*lr)/(ma + mr);

x2 = L2/2 * sin(q2(t))*cos(q1(t));
y2 = L2/2 * sin(q2(t))*sin(q1(t));
z2 = L1 + L2/2*cos(q2(t));

x3 = ( L2 * sin(q2(t)) + L3/2 * sin(q3(t)
    ) ) * cos(q1(t));
y3 = ( L2 * sin(q2(t)) + L3/2 * sin(q3(t)
    ) ) * sin(q1(t));
z3 = L1 + L2/2 * cos(q2(t)) + L3/2 * cos(
    q3(t));

x4 = x3 + L3/2 * sin(q3(t)) * cos(q1(t))
    + ( L4/2 * sin(q4(t)) * cos(q1(t)) );
y4 = y3 + L3/2 * sin(q3(t)) * sin(q1(t))
    + ( L4/2 * sin(q4(t)) * sin(q1(t)) );
z4 = z3 + L3/2 * cos(q3(t)) + L4/2 * cos(
    q4(t));

x5 = x4 + L4/2 * sin(q4(t)) * cos(q1(t))
    + ( L5/2 * sin(q5(t)) * cos(q1(t)) );
y5 = y4 + L4/2 * sin(q4(t)) * sin(q1(t))
    + ( L5/2 * sin(q5(t)) * sin(q1(t)) );
z5 = z4 + L4/2 * cos(q4(t)) + L5/2 * cos(
    q5(t));

x6 = x5 + L5/2 * sin(q5(t)) * cos(q1(t))
    + ( L6/2 * sin(q6(t)) * cos(q1(t)) );
y6 = y5 + L5/2 * sin(q5(t)) * sin(q1(t))
    + ( L6/2 * sin(q6(t)) * sin(q1(t)) );
```

Como se Puede observar en el código, las coordenadas mayores dependen de las anteriores. Al momento de expandir las expresiones en función de los parámetros y no las variables, las expresiones resultantes tienen un tamaño considerable. Sin embargo, al reemplazar los parámetros por valores numéricos las expresiones se reducen.

Posteriormente, se hallan las energías (potencial y cinética) de los segmentos del brazo.

```
% Energias:
syms J1 J2 J3 J4 J5 J6
V1 = 1/2*J1*q1_d(t)^2;
U1 = m1*g*(ma*la + mr*lr)/(ma + mr);

V2 = 1/2*J2*q2_d(t)^2 + 1/2*m2*(x2_d^2 +
    y2_d^2 + z2_d^2);
U2 = m2*g*(L1 + L2/2*cos(q2(t)));

V3 = 1/2*J3*q3_d(t)^2 + 1/2*m3*(x3_d^2 +
    y3_d^2 + z3_d^2);
U3 = m3*g*z3;

V4 = 1/2*J4*q4_d(t)^2 + 1/2*m4*(x4_d^2 +
    y4_d^2 + z4_d^2);
U4 = m4*g*z4;

V5 = 1/2*J5*q5_d(t)^2 + 1/2*m5*(x5_d^2 +
    y5_d^2 + z5_d^2);
U5 = m5*g*z5;

V6 = 1/2*J6*q6_d(t)^2 + 1/2*m6*(x6_d^2 +
    y6_d^2 + z6_d^2);
U6 = m6*g*z6;
```

Con las expresiones de las energías conocidas, se aplica las ecuaciones de Lagrange para hallar los torques.

```
V = V1 + V2 + V3 + V4 + V5 + V6;
U = U1 + U2 + U3 + U4 + U5 + U6;
L = simplify(expand(V - U));
```

Posteriormente, se calculan los torques para cada ángulo de la siguiente forma:

$$t_i = \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i}$$

Esto se aplicará para todos los ángulos, $i \in [1, 6]$.
Luego se hallan las dobles derivadas de los ángulos y se relacionan las entradas y los torques con la siguiente ecuación:

$$\tau_i = -J_{eq}\ddot{q}_i - (B_{eq} + \frac{n^2 K_m K_b}{R_a}) * \dot{q}_i + \frac{n K_m K_A}{R_a} u_i$$

Esto empleado para todas las salidas del sistema: $i \in [1, 6]$.
Por último, despejando las entradas (voltajes) u de las 6 ecuaciones y la expresión en MATLAB es la siguiente

```
u1 = (t1 - (-Jeq*q1dd - (Beq + n^2*Km*Kb/Ra)*q1_d(t)))*Ra/(n*Km*Ka);
u2 = (t2 - (-Jeq*q2dd - (Beq + n^2*Km*Kb/Ra)*q2_d(t)))*Ra/(n*Km*Ka);
u3 = (t3 - (-Jeq*q3dd - (Beq + n^2*Km*Kb/Ra)*q3_d(t)))*Ra/(n*Km*Ka);
u4 = (t4 - (-Jeq*q4dd - (Beq + n^2*Km*Kb/Ra)*q4_d(t)))*Ra/(n*Km*Ka);
u5 = (t5 - (-Jeq*q5dd - (Beq + n^2*Km*Kb/Ra)*q5_d(t)))*Ra/(n*Km*Ka);
u6 = (t6 - (-Jeq*q6dd - (Beq + n^2*Km*Kb/Ra)*q6_d(t)))*Ra/(n*Km*Ka);
```

Donde qidd es la doble derivada del ángulo número i.

Con las expresiones de las entradas se arman las matrices del modelo dinámico del sistema, de la siguiente forma.

$$u_1 = M_{11}\ddot{q_1} + P_{11}\dot{q_1} + P_{12}q_2 + \dot{P}_{13}\dot{q_3} + P_{14}\dot{q_4} + P_{15}\dot{q_5} + P_{16}\dot{q_6} \tag{3}$$

$$u_2 = M_{22}\ddot{q_2} + M_{23}\ddot{q_3} + M_{24}\ddot{q_4} + M_{25}\ddot{q_5} + M_{26}\ddot{q_6} + P_{21}\dot{q_1} + P_{12}q_2 d + \dot{P}_{13}\dot{q_3} + P_{14}\dot{q_4} + P_{15}\dot{q_5} + P_{16}\dot{q_6} + d_2 \tag{4}$$

$$u_3 = M_{32}\ddot{q_3} + M_{33}\ddot{q_3} + M_{34}\ddot{q_4} + M_{35}\ddot{q_5} + M_{36}\ddot{q_6} + P_{31}\dot{q_1} + P_{32}q_2 d + \dot{P}_{33}\dot{q_3} + P_{34}\dot{q_4} + P_{35}\dot{q_5} + P_{36}\dot{q_6} + d_3 \tag{5}$$

$$u_4 = M_{42}\ddot{q_2} + M_{43}\ddot{q_3} + M_{44}\ddot{q_4} + M_{45}\ddot{q_5} + M_{46}\ddot{q_6} + P_{41}\dot{q_1} + P_{42}q_2 d + \dot{P}_{43}\dot{q_3} + P_{44}\dot{q_4} + P_{45}\dot{q_5} + P_{46}\dot{q_6} + d_4 \tag{6}$$

$$u_5 = M_{52}\ddot{q_2} + M_{53}\ddot{q_3} + M_{54}\ddot{q_4} + M_{55}\ddot{q_5} + M_{56}\ddot{q_6} + P_{51}\dot{q_1} + P_{52}q_2 d + \dot{P}_{53}\dot{q_3} + P_{54}\dot{q_4} + P_{55}\dot{q_5} + P_{56}\dot{q_6} + d_5 \tag{7}$$

$$u_6 = M_{62}\ddot{q_2} + M_{63}\ddot{q_3} + M_{64}\ddot{q_4} + M_{65}\ddot{q_5} + M_{66}\ddot{q_6} + P_{61}\dot{q_1} + P_{62}q_2 d + \dot{P}_{63}\dot{q_3} + P_{64}\dot{q_4} + P_{65}\dot{q_5} + P_{66}\dot{q_6} + d_6 \tag{8}$$

Dado que se conocen las ecuaciones de las entradas, para hallar cada valor de las matrices se debe de factorizar los coeficientes de las variables de control respectivas. Este proceso se llevó a cabo con las funciones $coeffs()$ y $collect()$ en MATLAB. Luego, se arman las matrices con el siguiente código:

```
% Matriz M:
M = [M11  0  0  0  0  0;
      0 M22 M23 M24 M25 M26;
      0 M32 M33 M34 M35 M36;
      0 M42 M43 M44 M45 M46;
      0 M52 M53 M54 M55 M56;
      0 M62 M63 M64 M65 M66;];


% Patriz P:
P = [P11 P12 P13 P14 P15 P16;
     P21 P22 P23 P24 P25 P26;
     P22 P32 P33 P34 P35 P36;
     P23 P42 P43 P44 P45 P46;
     P24 P52 P53 P54 P55 P56;
     P25 P62 P63 P64 P65 P66;];

% Vector D:
D = [0; d21; d31; d41; d51; d61];
```

Con las matrices M y P, además del vector D, el modelo dinámico del sistema está completo y listo para probar el controlador seleccionado, en este caso el controlador backstepping.

III. El Algoritmo Backstepping

El controlador backstepping fue diseñado usando el código de [1] como punto de partida. El algoritmo en cuestión tiene la siguiente forma:

$$u = M(q)\ddot{q}_d + P(q,\dot{q})\dot{q}_r + d(q) - K_d(\dot{q} - \dot{q}_r) - K_1 z_1 \quad (9)$$

K_d y K_1 son matrices diagonales con elementos positivos. El error de seguimiento z se define de la siguiente forma:

$$z_1 = q - q_d \quad (10)$$

Donde q es el vector de posiciones medidas y q_d es el vector de las trayectorias deseadas.
La estimación del vector $\dot{q}$ del vector conocido q es la siguiente:

$$\dot{\hat{q}} = \dot{q}_d + L_d(q - \hat{q}) \quad (11)$$

L_d es uan matriz diagonal con elementos positivos.
El controlador backstepping resultante es el siguiente:

$$u = M(q)\ddot{q}_d + P(q,\dot{\hat{q}})\dot{q}_r + d(q) - K_d(\dot{\hat{q}} - \dot{q}_r) - K_1 z_1 \quad (12)$$

IV. Simulación en MATLAB

Para llevar a cabo la simulación, se sustituyeron las constantes simbólicas por el valor del parámetro que representaban. Esto se llevó a cabo utilizando la función $subs()$.
Sin embargo, debido a que las salidas (los ángulos) aún están presentes en las matrices, las matrices se deben de convertir a un objecto en MATLAB que permita la rápida sustitución de sus valores para la simulación. El modo simbólico no es apto para este caso ya que éste representa las operaciones de números racionales en forma fraccionaria y no decimal. Dicho cálculo tarda un tiempo considerable. Por lo tanto, las matrices simbólicas se convirtieron en funciones one-line de MATLAB, usando el siguiente código:

```
M = matlabFunction(vpa(simplify(expand(M)))));
P = matlabFunction(vpa(simplify(expand(P)))));
D = matlabFunction(vpa(simplify(expand(D)))));
```

En primer lugar, se expande la expresión simbólica de cada matriz para poder reagrupar los coeficientes de las variables, en segundo lugar, se simplifica la expresión expandida y se utiliza la función vpa() para convertir las fracciones simbólicas en expresiones de punto flotante, sin embargo, los ángulos aún son expresiones simbólicas. Por último, la función matlabFunction() convierte a la expresión simbólica resultante en una función de matlab la cual tiene los ángulos como parámetros. La función resultante tiene la siguiente forma:

```
M = function_handle with value:
@(q2_,q3_,q4_,q5_,q6_)reshape([sin(q2_).*
    sin(q3_).  [...]
```

Al tener las matrices en forma de funciones se puede proceder a simular el controlador en MATLAB. Se definen los parámetros del controlador, los valores iniciales, el tiempo de muestreo y las referencias deseadas.

V. Resultados de la simulación

La simulación de seguimiento se llevó a cabo con un tiempo de 10 segundos, mientras que en la simulación con referencias constantes el tiempo fue de 4 segundos.

A. Simulación con referencias constantes

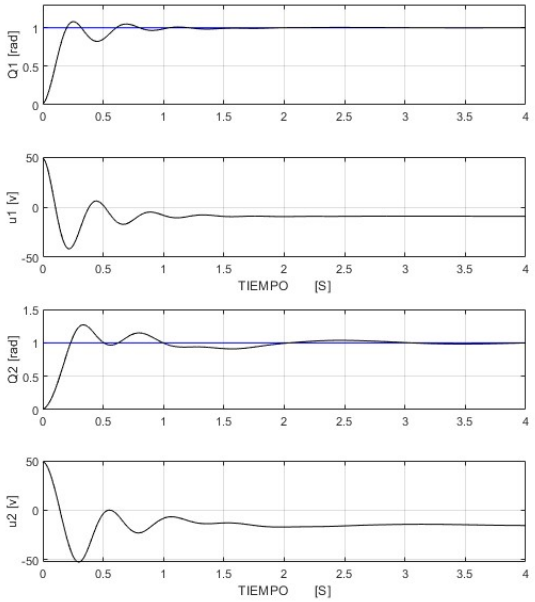

Fig. 3. Señales de entrada y salida 1 y 2

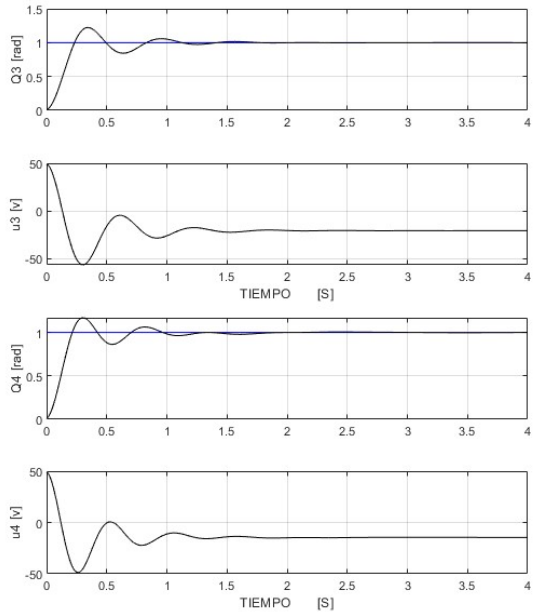

Fig. 4. Señales de entrada y salida 3 y 4

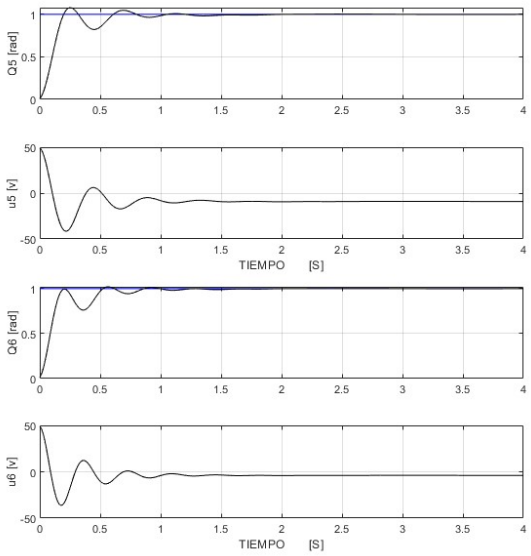

Fig. 5. Señales de entrada y salida 5 y 6

B. Simulación de seguimiento

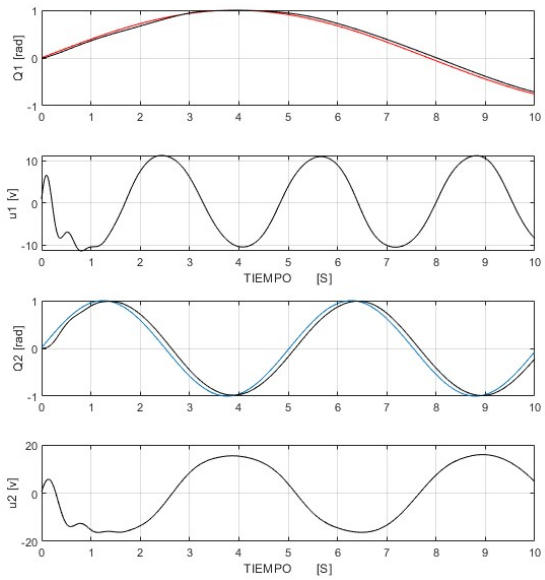

Fig. 6. Señales de entrada y salida 1 y 2

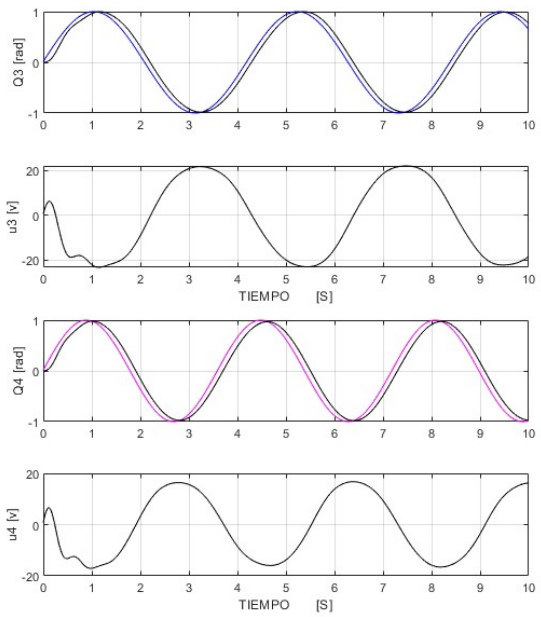

Fig. 7. Señales de entrada y salida 3 y 4

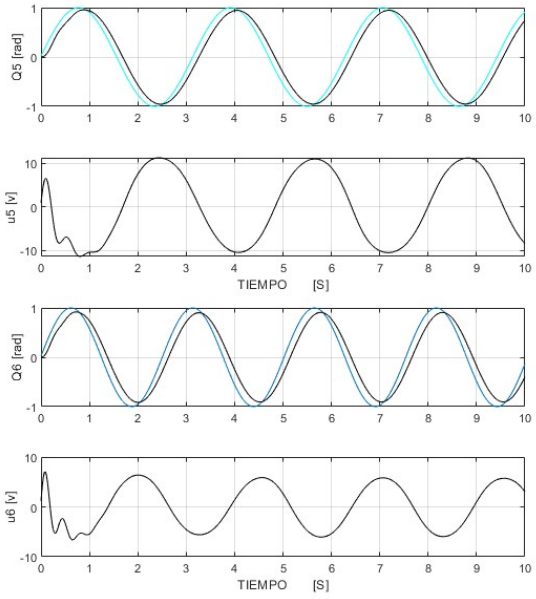

Fig. 8. Señales de entrada y salida 5 y 6

VI. Conclusiones

El modelado del sistema se llevó a cabo de manera eficiente utilizando el modo simbólico de MATLAB. El algoritmo backstepping es del tipo que se basa en el modelo ya que requiere el modelo dinámico del brazo robótico.

El modo simbólico en MATLAB no es eficiente para su uso en el lazo de control, por lo tanto las variables simbólicas deben convertirse en una función de MATLAB para simular el controlador.

Todas las salidas del sistema de control diseñado fueron

capaces de seguir las señales arbitrarias diseñadas deseadas, cumpliendo los parámetros establecidos: un overshoot menor al 10%. un tiempo de establecimiento menor a 3 segundos y un error en estado estable menor al 1%.

REFERENCES

[1] H. Ventura, "Diseño e Imlementación de un Controlador Fuzzy Adaptativo para un Brazo Robótico de 3GL", Editorial: UNI, 2017

[2] A. Rojas, "Control No Lineal Multivariable", Editorial: UNI, 2012

[3] ROBOT INDUSTRIAL DE 6 GRADOS DE LIBERTAD (s.f.) [fotografía] http://equinlabsac.com/sites/default/files/5b74439912482.jpg