Cleveland State University
Fenn College of Engineering

MCE 647: Robot Dynamics and Control

Term Project

Kinematics, Dynamics, Control and Simulation of RP Planar Arm

Instructor:	Dr. Hanz Richter Mechanical Engineering Department
Submitted by:	Dmitry Storozhev

December 12, 2008

Department of Mechanical Engineering
Cleveland State University

MCE/EEC 647/747: Robot Dynamics and Control
Term Project - Fall 2008
Kinematics, Dynamics, Control and Simulation of RP Planar Arm

Due: Friday, December 12 by 5 PM

Consider a 2-link RP planar manipulator. Assume that the links have lenghts a_1, and a_2 and moments of inertia I_{1z} and I_{2y} for planar rotation about the center of mass. Suppose the center of mass is at the center of each link. It is recommended to use the angle of link 1 and the distance between the centers of mass of link 1 and link 2 as joint variables q_1 and q_2.

Kinematics and Dynamics

1. Set up appropriate coordinate systems and find the D-H parameter table.
2. Find the T matrices relating each link-attached frame to the world frame, according to your choice of origin points.
3. Find the two velocity Jacobians and the two angular velocity Jacobians relative to each center of mass.
4. Find the $D(q)$, $C(q,\dot{q})$ and $g(q)$ matrices.
5. Set up a Matlab/Simulink simulation implementing the dynamic equation $D(q)\ddot{q}+C(q,\dot{q})\dot{q}+g(q)=\tau$. Please set up a parametric simulation system, that is, all parameters (masses, lengths, etc.) should be symbolic. Set up a separate m-file to load values for the parameters.
6. Try out your simulation by setting all parameters to 1 and applying $\tau=[1\ 1]^T$. Simulate for 1 second and plot all joint positions and velocities against time.
7. Consult with the instructor to see whether you're on the right track and ready for the next stage (control analysis/design).

Linear Parameterization

1. Let the masses of the links be m_1 and m_2, and let the angle of link be q_1 and the distance between centers of mass be q_2. Show that the following regressor and parameter vectors satisfy the dynamic equations derived above:

$$Y(q,\dot{q},\ddot{q}) = \begin{bmatrix} \ddot{q}_1 & q_2^2\ddot{q}_1 + 2q_2\dot{q}_1\dot{q}_2 + gq_2\cos(q_1) & 2\dot{q}_1\dot{q}_2 + 2q_2\ddot{q}_1 + g\cos(q_1) & g\cos(q_1) \\ 0 & \ddot{q}_2 - q_2\dot{q}_1^2 + g\sin(q_1) & -\dot{q}_1^2 & 0 \end{bmatrix} \tag{1}$$

$$\Theta = \begin{bmatrix} \frac{1}{4}m_1a_1^2 + I_{1z} + \frac{1}{4}m_2a_1^2 + I_{2y} \\ m_2 \\ \frac{1}{2}m_2a_1 \\ \frac{1}{2}m_1a_1 \end{bmatrix} \tag{2}$$

2. Find the regressor $Y(a,v,q,\dot{q})$ corresponding to the term $\hat{M}a + \hat{C}v + \hat{g}$
3. Take $g = 9.8\ \mathrm{m/s^2}$ and the following nominal values for the parameters:

$a_1 = 0.6$ m
$a_2 = 0.4$ m
$m_1 = 10$ kg
$m_2 = 8$ kg
$I_{1z} = 26$ kg-m^2
$I_{2y} = 12$ kg-m^2

Let these parameters generate the nominal parameter vector Θ_0. Supposing that each basic parameter (m_1 , m_2, a_1, a_2, I_{1z} and I_{2y}) can be off by p percent, determine the maximum error bound for the parameter vector. That is, find $\rho(p)$ so that

$$||\Theta - \Theta_0|| \leq \rho(p)$$

Do this for p=10 percent, p=25 percent and p =80 percent by taking the worst-case.

Control Design Simulation Setup

The desired joint trajectories are as follows:

$$\begin{aligned} q_1^d(t) &= 10t \\ q_2^d(t) &= 1 + \cos(30t) \end{aligned}$$

You must pre-compute the desired velocities and accelerations to be used directly in the control algorithms (avoid online differentation). Build a simulation using Matlab/Simulink to test each one of the following cases:

1. **Robust passivity-based control**: Generate a "true" set of parameters using p=10 percent (random perturbation). Choose a suitable value of deadzone ε and gain matrices Λ and K. Use the true parameters in the plant and the nominal ones in the control law. Plot $\tilde{q}_1(t)$ and $\tilde{q}_2(t)$. Then plot the y position vs. the x position of the end-effector corresponding to $q_1^d(t)$ and $q_2^d(t)$ with a red dashed line. On the same plot, plot the actual y vs. x (using $q_1(t)$ and $q_2(t)$) with a solid line of another color. Finally plot the two components of the control input against time.
2. Repeat the above for p =25 percent. Re-adjust the gains and ε if needed.
3. **Adaptive passivity-based control**: Generate a "true" set of parameters using p=80 percent (random perturbation). Choose suitable gain matrices Λ, K and Γ. Use the true parameters in the plant and the nominal ones as initial parameter guesses in the control law. Run the adaptive system and tune the gain matrices to obtain good parameter and error convergence. Plot each adapted parameter vs. time. Plot $\tilde{q}_1(t)$ and $\tilde{q}_2(t)$. Then plot the y position vs. the x position of the end-effector corresponding to $q_1^d(t)$ and $q_2^d(t)$ with a red dashed line. On the same plot, plot the actual y vs. x (using $q_1(t)$ and $q_2(t)$) with a solid line of another color. Finally plot the two components of the control input against time. Did the parameters converge to their true values?

How to turn in the exam:

All hand calculations must be explained. Please scan any handwritten sheets. Email the instructor, attaching the hand calculations and m-file(s). Although the use of packages like *Robotica* is allowed for verification purposes, the Jacobians and all required transformation matrices must be shown in relationship to the chosen coordinate systems. Students are encouraged to use symbolic engines (Matlab, Python, Mathematica, Maple, etc.) to assist with symbolic matrix multiplications and simplification, although it is not necessary to use these packages to carry the operations.

Part #1: Kinematics and Dynamics

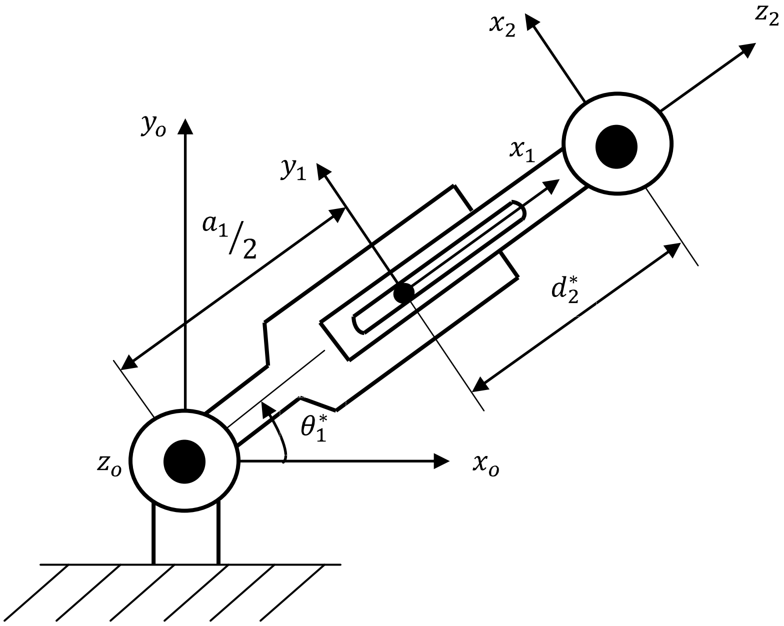


Figure 1. Coordinate systems of the RP planar arm

Table 1 shows D-H parameters of the RP planar arm corresponding to figure 1.

Link	a_i	α_i	d_i	θ_i
1	$a_1/2$	0	0	θ_1^*
$1'$	0	0	0	90^o
$1''$	0	90^o	0	0
2	0	0	d_2^*	0

Table 1. D-H parameters for the RP planar manipulator

Matlab software was used to find:

a) T matrices relating each link-attached frame to the world frame,

b) Two velocity Jacobians (J_{v1} and J_{v2}) and two angular velocity Jacobians (J_{w1} and J_{w2})

relative to each center of mass,

c) $D(q), C(q,\dot{q})$ and $g(q)$ matrices.

M-file syntax:

```
clear;clc

syms
('a1','d2','theta1','Ixx_1','Ixy_1','Ixz_1','Iyx_1','Iyy_1','Iyz_1','Izx_1','Izy_1','
Izz_1','Ixx_2','Ixy_2','Ixz_2','Iyx_2','Iyy_2','Iyz_2','Izx_2','Izy_2','Izz_2','m1','
m2','g');


%-------------- DH parameters for link 1 ----------------
theta1=input('Enter  "theta1" (rotation angle for link 1)   =  ');
d1=0;
a1=input('Enter "a1" (length of link 1)  =  ');
ac1=a1/2;
alpha1=0;

%-------------- DH parameters for link 2 -----------------
theta1_1=90;
d1_1=0;
a1_1=0;
alpha1_1=0;

theta1_2=0;
d1_2=0;
a1_2=0;
alpha1_2=90;

theta2=0;
d2=input('Enter "d2" (distance between the centers of mass of link 1 and link 2) =  ');
a2=0;
alpha2=0;

%----------------------------------------------------------------

Ac1=[cos(theta1) -sin(theta1)*cos(alpha1*pi/180) sin(theta1)*sin(alpha1*pi/180)
ac1*cos(theta1);
    sin(theta1) cos(theta1)*cos(alpha1*pi/180) -cos(theta1)*sin(alpha1*pi/180)
ac1*sin(theta1);
    0 sin(alpha1*pi/180) cos(alpha1*pi/180) d1;
    0 0 0 1];
```

```
A1_1=[cos(theta1_1) -sin(theta1_1)*cos(alpha1_1*pi/180)
sin(theta1_1)*sin(alpha1_1*pi/180) a1_1*cos(theta1_1);
    sin(theta1_1) cos(theta1_1)*cos(alpha1_1*pi/180)
-cos(theta1_1)*sin(alpha1_1*pi/180) a1_1*sin(theta1_1);
    0 sin(alpha1_1*pi/180) cos(alpha1_1*pi/180)) d1_1;
    0 0 0 1];

A1_2=[cos(theta1_2)) -sin(theta1_2)*cos(alpha1_2*pi/180)
sin(theta1_2)*sin(alpha1_2*pi/180) a1_2*cos(theta1_2);
    sin(theta1_2) cos(theta1_2)*cos(alpha1_2*pi/180)
-cos(theta1_2)*sin(alpha1_2*pi/180) a1_2*sin(theta1_2);
    0 sin(alpha1_2*pi/180) cos(alpha1_2*pi/180) d1_2;
    0 0 0 1];

Ac2=[cos(theta2) -sin(theta2)*cos(alpha2*pi/180) sin(theta2)*sin(alpha2*pi/180)
a2*cos(theta2);
    sin(theta2) cos(theta2)*cos(alpha2*pi/180)
-cos(theta2)*sin(alpha2*pi/180) a2*sin(theta2);
    0 sin(alpha2*pi/180) cos(alpha2*pi/180) d2;
    0 0 0 1];

%---------------------------------------------------------------

T_c1_0=Ac1        % Transformation matrices from origin of the world frame to
T_c2_0=Ac1*A1_1*A1_2*Ac2    %the center of the mass for each link

%---------------Velocity Jacobians and Angular Velocity Jacobians------------

z0=[0;0;1];         % the first three elements in the third column of T_i_0
z1=T_c1_0(1:3,3);   % where i=0,1,2

o0=[0;0;0];         % the first three elements in the fourth column of T_i_0
oc1=T_c1_0(1:3,4);  % where i=0,1,2
oc2=T_c2_0(1:3,4);


Jc1=[cross(z0,(oc1-o0)) [0;0;0];          % The 6x2 Jacobian relative to the
            z0          [0;0;0]]          % center of mass of link 1

Jc2=[cross(z0,(oc2-o0)) z1;               % The 6x2 Jacobian relative to the
            z0     [0;0;0]]               % center of mass of link 2


Jv1=Jc1(1:3,1:2);
Jw1=Jc1(4:6,1:2);

Jv2=Jc2(1:3,1:2);
Jw2=Jc2(4:6,1:2);

%--------------------------------------------------------------------------
R_1_0=T_c1_0(1:3,1:3);               % Rotation matrices
R_2_0=T_c2_0(1:3,1:3);

%--------------------------------------------------------------------------
I_1=[Ixx_1 Ixy_1 Ixz_1;              % The inertia tensor
     Iyx_1 Iyy_1 Iyz_1;
     Izx_1 Izy_1 Izz_1];
```

```
I_2=[Ixx_2 Ixy_2 Ixz_2;
     Iyx_2 Iyy_2 Iyz_2;
     Izx_2 Izy_2 Izz_2];


II_1=(R_1_0)*(I_1)*(R_1_0).'  ;
II_2=(R_2_0)*(I_2)*(R_2_0).'  ;

D_1=(Jw1.')*(II_1)*(Jw1);
D_2=(Jw2.')*(II_2)*(Jw2);

D=D_1+D_2;

% Translational part of kinetic energy
K=(m1)*(Jv1.')*(Jv1)+(m2)*(Jv2.')*(Jv2);

% Inertia matrix
D_q=K+D;
D_q=simplify(D_q)

q=[theta1;d2];
for i=1:2
    for j=1:2
        for k=1:2
            c(i,j,k)=(1/2)*((diff(D_q(k,j),q(i)))+(diff(D_q(k,i),q(j)))-
(diff(D_q(i,j),q(k))));
         end
     end
end


syms ('theta1_dot','d2_dot')
for k=1:2
    for j=1:2
        C(k,j)=(c(1,j,k)*theta1_dot+c(2,j,k)*d2_dot);
    end
end

% Potential energy
P_1=m1*g*ac1*sin(theta1);
P_2=m2*g*(ac1+d2)*sin(theta1);


P=P_1+P_2;

g_1=diff(P,theta1);
g_2=diff(P,d2);

g_q=[g_1;g_2]
```

Command window syntax:

```
Enter  "theta1" (rotation angle for link 1)   =  theta1

Enter "a1" (length of link 1)  =  a1

Enter "d2" (distance between the centers of mass of link 1 and link 2)  =  d2

T_c1_0 =

[        cos(theta1),       -sin(theta1),                  0, 1/2*a1*cos(theta1)]

[        sin(theta1),        cos(theta1),                  0, 1/2*a1*sin(theta1)]

[              0    ,              0     ,                  1,         0         ]

[              0    ,              0     ,                  0,         1         ]

T_c2_0 =

[ -sin(theta1),           0,       cos(theta1),   cos(theta1)*d2+1/2*a1*cos(theta1)]

[  cos(theta1),           0,       sin(theta1),   sin(theta1)*d2+1/2*a1*sin(theta1)]

[        0    ,           1,             0    ,                    0               ]

[        0    ,           0,             0    ,                    1               ]

Jc1 =

[ -1/2*a1*sin(theta1),                    0]

[  1/2*a1*cos(theta1),                    0]

[        0,                               0]

[        0,                               0]

[        0,                               0]

[        1,                               0]

Jc2 =

[ -sin(theta1)*d2-1/2*a1*sin(theta1),                                    0]

[  cos(theta1)*d2+1/2*a1*cos(theta1),                                    0]

[                                  0,                                    1]

[                                  0,                                    0]

[                                  0,                                    0]

[                                  1,                                    0]
```

```
D_q =

[ 1/4*m1*a1^2+m2*d2^2+m2*d2*a1+1/4*m2*a1^2+Izz_1+Iyy_2,               0]

[                            0                         ,              m2]


C =

[       (m2*d2+1/2*m2*a1)*d2_dot,          (m2*d2+1/2*m2*a1)*theta1_dot]

[ (-m2*d2-1/2*m2*a1)*theta1_dot,                          0             ]


g_q =

[ 1/2*m1*g*a1*cos(theta1)+m2*g*(1/2*a1+d2)*cos(theta1)]

[                    m2*g*sin(theta1)                  ]
```

The dynamic equation for the RP planar arm is:

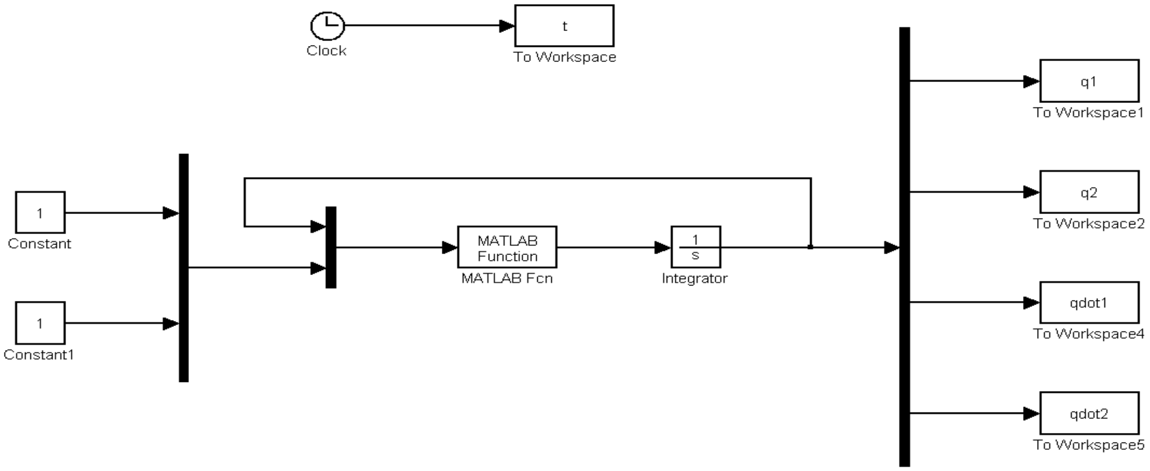
$$\begin{bmatrix} \frac{1}{4}m_1a_1^2+m_2d_2^2+m_2d_2a_1+\frac{1}{4}m_2a_1^2+I_{zz1}+I_{yy2} & 0 \\ 0 & m_2 \end{bmatrix}\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix}+\begin{bmatrix} \left(m_2d_2+\frac{1}{2}m_2a_1\right)\dot{d}_2 & \left(m_2d_2+\frac{1}{2}m_2a_1\right)\dot{\theta}_1 \\ \left(-m_2d_2-\frac{1}{2}m_2a_1\right)\dot{\theta}_1 & 0 \end{bmatrix}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

$$+\begin{bmatrix} \frac{1}{2}m_1ga_1\cos\theta_1+m_2g\left(\frac{1}{2}a_1+d_2\right)\cos\theta_1 \\ m_2g\sin\theta_1 \end{bmatrix}=\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \qquad (1)$$

where: θ_1 – joint variable of link 1 (angle of link 1)

d_2 – joint variable of link 2 (distance between centers of mass of link 1 and link 2)

Simulink was used to set up a simulation of the dynamic equation (1).



M-file that loads values for the parameters:

```
function qdot=project(t,q,u)

m1=1;
m2=1;
a1=1;
a2=1;
g=1;
Izz_1=1;
Iyy_2=1;

n=length(q);
q_1=q(1:n/2);
q_2=q(n/2+1:n);


D(1,1)=1/4*m1*a1^2+m2*q_1(2)^2+m2*q_1(2)*a1+1/4*m2*a1^2+Izz_1+Iyy_2;
D(1,2)=0;
D(2,1)=D(1,2);
D(2,2)=m2;
C(1,1)=(m2*q_1(2)+1/2*m2*a1)*(q_2(2));
C(1,2)=(m2*q_1(2)+1/2*m2*a1)*(q_2(1));
C(2,1)=(-m2*q_1(2)-1/2*m2*a1)*(q_2(1));
C(2,2)=0;
```

```
g_q(1,1)=1/2*g*m1*a1*cos(q_1(1))+m2*g*(1/2*a1+q_1(2))*cos(q_1(1));
g_q(2,1)=m2*g*sin(q_1(1));

qdot=[q_2;inv(D)*(u-C*q_2-g_q)];
```

Simulation was done by settling all parameters to 1 and applying $\tau = \begin{bmatrix}1\\1\end{bmatrix}$. Figure 2 shows positions and velocities of joint 1 and joint 2.

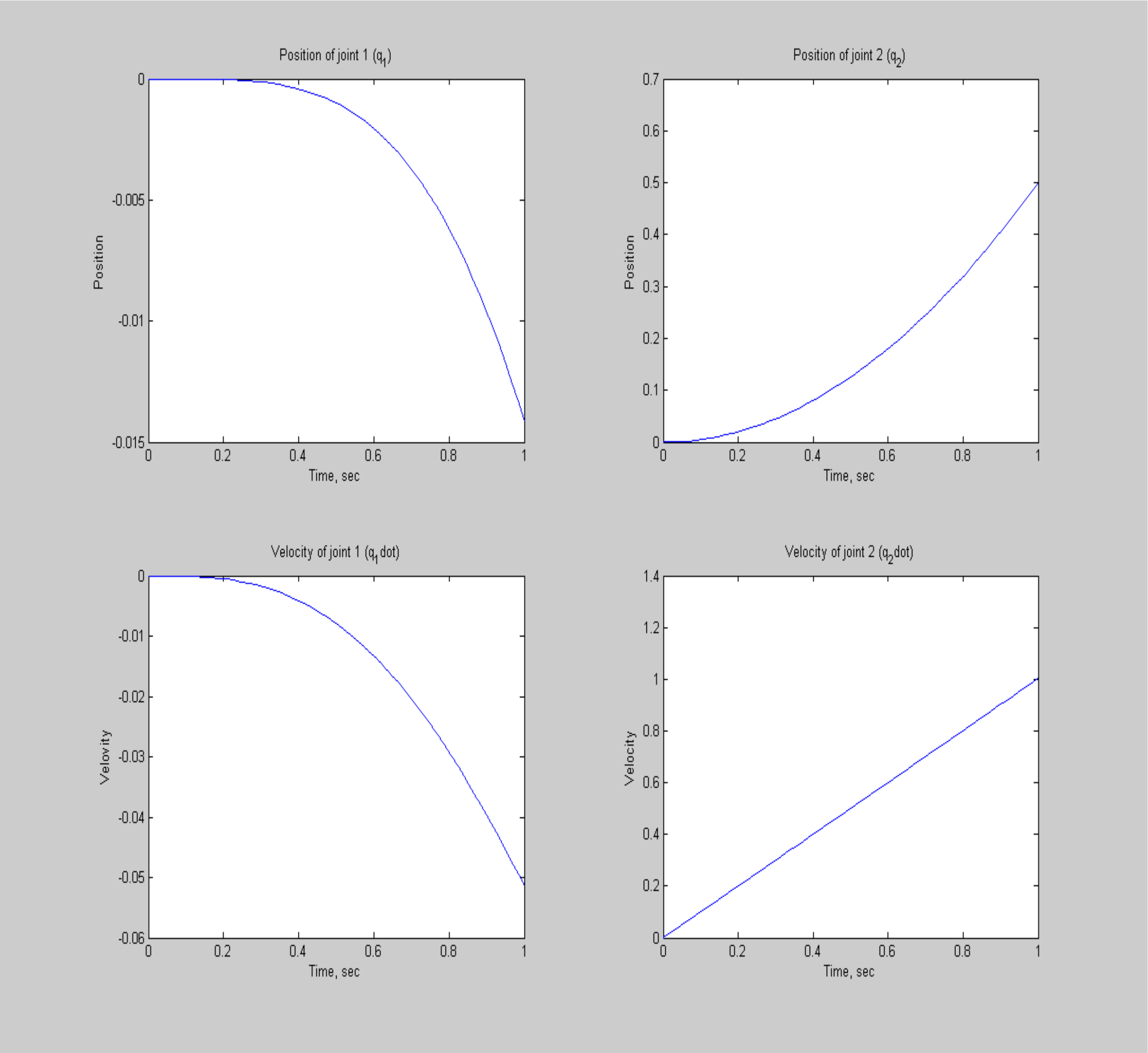


Figure 2. Positions and Velocities of Joint 1 and Joint 2

Part #2: Linear Parameterization

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = Y(q,\dot{q},\ddot{q})\theta \qquad (1)$$

$$Y\theta = \begin{bmatrix} \ddot{q}_1 & q_2^2\ddot{q}_1 + 2q_2\dot{q}_1\dot{q}_2 + gq_2\cos q_1 & 2\dot{q}_1\dot{q}_2 + 2q_2\ddot{q}_1 + g\cos q_1 & g\cos q_1 \\ 0 & \ddot{q}_2 - q_2\dot{q}_1^2 + g\sin q_1 & -\dot{q}_1^2 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{4}m_1a_1^2 + \frac{1}{4}m_2a_1^2 + I_{zz1} + I_{yy2} \\ m_2 \\ \frac{1}{2}m_2a_1 \\ \frac{1}{2}m_1a_1 \end{bmatrix} \qquad (2)$$

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \begin{bmatrix} \frac{1}{4}m_1a_1^2 + m_2d_2^2 + m_2d_2a_1 + \frac{1}{4}m_2a_1^2 + I_{zz1} + I_{yy2} & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix}$$

$$+ \begin{bmatrix} \left(m_2d_2 + \frac{1}{2}m_2a_1\right)\dot{d}_2 & \left(m_2d_2 + \frac{1}{2}m_2a_1\right)\dot{\theta}_1 \\ \left(-m_2d_2 - \frac{1}{2}m_2a_1\right)\dot{\theta}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{2}m_1ga_1\cos\theta_1 + m_2g\left(\frac{1}{2}a_1 + d_2\right)\cos\theta_1 \\ m_2g\sin\theta_1 \end{bmatrix} \qquad (3)$$

From equation (2):

First row of $Y\theta$:

$$Y\theta = \left(\frac{1}{4}m_1a_1^2 + \frac{1}{4}m_2a_1^2 + I_{zz1} + I_{yy2}\right)\ddot{q}_1 + (q_2^2\ddot{q}_1 + 2q_2\dot{q}_1\dot{q}_2 + gq_2\cos q_1)m_2$$

$$+(2\dot{q}_1\dot{q}_2 + 2q_2\ddot{q}_1 + g\cos q_1)\left(\frac{1}{2}m_2a_1\right) + (g\cos q_1)\left(\frac{1}{2}m_1a_1\right) \qquad (4)$$

Second row of $Y\theta$:

$$Y\theta = (\ddot{q}_2 - q_2\dot{q}_1^2 + g\sin q_1)m_2 - (\dot{q}_1^2)\left(\frac{1}{2}m_2a_1\right) \qquad (5)$$

$$Y\theta = \begin{bmatrix} \frac{1}{4}m_1a_1^2 + m_2q_2^2 + m_2q_2a_1 + \frac{1}{4}m_2a_1^2 + I_{zz1} + I_{yy2} & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix}$$

$$+ \begin{bmatrix} \left(m_2q_2 + \frac{1}{2}m_2a_1\right)\dot{q}_2 & \left(m_2q_2 + \frac{1}{2}m_2a_1\right)\dot{q}_1 \\ \left(-m_2q_2 - \frac{1}{2}m_2a_1\right)\dot{q}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{2}m_1ga_1\cos q_1 + m_2g\left(\frac{1}{2}a_1 + q_2\right)\cos q_1 \\ m_2g\sin q_1 \end{bmatrix} \qquad (6)$$

Comparing equation (2) and equation (6) with $\theta_1 = q_1$ and $d_2 = q_2$, we can conclude that the regressor $Y(q, \dot{q}, \ddot{q})$ and parameter vector θ satisfy the dynamic equations i.e. $D\ddot{q} + C\dot{q} + g = Y\theta$

Procedure for finding the regressor $Y(a, v, q, \dot{q})$ corresponding to the term $\widehat{M}a$+$\hat{C}v + \hat{g}$

Using given regressor and parameter vector:

$$[\widehat{M}] = \begin{bmatrix} \theta_1 + \theta_2q_2^2 + 2\theta_3q_2 & 0 \\ 0 & \theta_2 \end{bmatrix}$$

$$[\hat{C}] = \begin{bmatrix} (\theta_2q_2 + \theta_3)\dot{q}_2 & (\theta_2q_2 + \theta_3)\dot{q}_1 \\ (-\theta_2q_2 - \theta_3)\dot{q}_1 & 0 \end{bmatrix}$$

$$[\hat{g}] = \begin{bmatrix} \theta_4g\cos q_1 + \theta_3g\cos q_1 + \theta_2q_2g\cos q_1 \\ \theta_2g\sin q_1 \end{bmatrix}$$

$$\widehat{M}a + \hat{C}v + \hat{g} = \begin{bmatrix} \theta_1 + \theta_2q_2^2 + 2\theta_3q_2 & 0 \\ 0 & \theta_2 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} + \begin{bmatrix} (\theta_2q_2 + \theta_3)\dot{q}_2 & (\theta_2q_2 + \theta_3)\dot{q}_1 \\ (-\theta_2q_2 - \theta_3)\dot{q}_1 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

$$+ \begin{bmatrix} \theta_4g\cos q_1 + \theta_3g\cos q_1 + \theta_2q_2g\cos q_1 \\ \theta_2g\sin q_1 \end{bmatrix}$$

First row of $\widehat{M}a+\hat{C}v+\hat{g}$:

$$\widehat{M}a+\hat{C}v+\hat{g} = \left(\theta_1+\theta_2 q_2^2+2\theta_3 q_2\right)a_1 + \left((\theta_2 q_2+\theta_3)\dot{q}_2\right)v_1 + \left((\theta_2 q_2+\theta_3)\dot{q}_1\right)v_2$$

$$+\theta_4 g\cos q_1+\theta_3 g\cos q_1+\theta_2 q_2 g\cos q_1$$

Second row of $\widehat{M}a+\hat{C}v+\hat{g}$:

$$\widehat{M}a+\hat{C}v+\hat{g} = \theta_2 a_1 + \left((-\theta_2 q_2-\theta_3)\dot{q}_1\right)v_1+\theta_2 g\sin q_1$$

Factoring out θ from the first row of $\widehat{M}a+\hat{C}v+\hat{g}$:

$$\widehat{M}a+\hat{C}v+\hat{g} = \theta_1(a_1)+\theta_2\left(q_2^2 a_1+q_2\dot{q}_2 v_1+q_2\dot{q}_1 v_2+q_2 g\cos q_1\right)$$

$$+\theta_3\left(2q_2 a_1+\dot{q}_2 v_1+\dot{q}_1 v_2+g\cos q_1\right)+\theta_4(g\cos q_1)$$

Factoring out θ from the second row of $\widehat{M}a+\hat{C}v+\hat{g}$:

$$\widehat{M}a+\hat{C}v+\hat{g} = \theta_1(0)+\theta_2\left(a_2-q_2\dot{q}_1 v_1+g\sin q_1\right)-\theta_3\left(\dot{q}_1 v_1\right)+\theta_4(0)$$

The regressor $Y(a,v,q,\dot{q})$ corresponding to the term $\widehat{M}a+\hat{C}v+\hat{g}$:

$$Y(a,v,q,\dot{q}) = \begin{bmatrix} a_1 & q_2^2 a_1+q_2\dot{q}_2 v_1+q_2\dot{q}_1 v_2+g q_2\cos q_1 & \dot{q}_2 v_1+\dot{q}_1 v_2+2q_2 a_1+g\cos q_1 & g\cos q_1 \\ 0 & a_2-q_2\dot{q}_1 v_1+g\sin q_1 & -\dot{q}_1 v_1 & 0 \end{bmatrix}$$

M-file was set up to generate the nominal parameter vector θ_o and a “true” set of parameters.

```
m1=10;
m2=8;
a1=0.6;
Izz_1=26;
Iyy_2=12;
g=9.8;

TH1_0=(1/4)*m1*a1^2+Izz_1+(1/4)*m2*a1^2+Iyy_2;
TH2_0=m2;
TH3_0=(1/2)*m2*a1;
TH4_0=(1/2)*m1*a1;

TH_0=[TH1_0;TH2_0;TH3_0;TH4_0];

level=0.1;    %           10% uncertainty level
%level=0.25;  %           25% uncertainty level
%level=0.8;   %           80% uncertainty level

m1_max=(1+level)*m1;
m2_max=(1+level)*m2;
a1_max=(1+level)*a1;
Izz_1_max=(1+level)*Izz_1;
Iyy_2_max=(1+level)*Iyy_2;

TH1_max=(1/4)*m1_max*a1_max^2+Izz_1_max+(1/4)*m2_max*a1_max^2+Iyy_2_max;
TH2_max=m2_max;
TH3_max=(1/2)*m2_max*a1_max;
TH4_max=(1/2)*m1_max*a1_max;

DTH=[TH1_max-TH1_0;TH2_max-TH2_0;TH3_max-TH3_0;TH4_max-TH4_0];

rho=norm(DTH);

% Perturbations for use in plant
m1=m1+m1*(1-2*rand)*level
m2=m2+m2*(1-2*rand)*level
a1=a1+a1*(1-2*rand)*level
Izz_1=Izz_1+Izz_1*(1-2*rand)*level
Iyy_2=Iyy_2+Iyy_2*(1-2*rand)*level
```

The maximum error bound for the parameter vector is:

for $p = 10\%$ is $\rho = 4.4826$
for $p = 25\%$ is $\rho = 11.4299$
for $p = 80\%$ is $\rho = 39.7037$

The nominal parameters vector is TH_0=[39.62; 8; 2.4; 3]^T

Part #3: Control Design

Robust Passivity-Based Control

M-file was generated for use in the plant (10% uncertainty):

```
function zdot=plant_10_uncert(t,z,u)

m1=10.5721;
m2=7.7704;
a1=0.6216;
g=9.8;
Izz_1=23.6075;
Iyy_2=11.4561;

n=length(z);
z_1=z(1:n/2);
z_2=z(n/2+1:n);

D(1,1)=1/4*m1*a1^2+m2*z_1(2)^2+m2*z_1(2)*a1+1/4*m2*a1^2+Izz_1+Iyy_2;
D(1,2)=0;
D(2,1)=D(1,2);
D(2,2)=m2;

C(1,1)=(m2*z_1(2)+1/2*m2*a1)*(z_2(2));
C(1,2)=(m2*z_1(2)+1/2*m2*a1)*(z_2(1));
C(2,1)=(-m2*z_1(2)-1/2*m2*a1)*(z_2(1));
C(2,2)=0;

g_z(1,1)=1/2*g*m1*a1*cos(z_1(1))+m2*g*(1/2*a1+z_1(2))*cos(z_1(1));
g_z(2,1)=m2*g*sin(z_1(1));

zdot=[z_2;inv(D)*(u-C*z_2-g_z)];
```

M-file was generated for use in the control law:

```
function u=robust_passivity_10_uncert(t,z,zd)

g=9.8;
q1=z(1);
q2=z(2);
q1dot=z(3);
q2dot=z(4);

q1d=zd(1);
q2d=zd(2);
q1ddot=zd(3);
q2ddot=zd(4);
q1dddot=zd(5);
q2dddot=zd(6);
```

```
qtilde=[q1-q1d;q2-q2d];
qtildedot=[q1dot-q1ddot;q2dot-q2ddot];

%Gains
L=diag([100 100]);
K=diag([100 100]);

v=[q1ddot;q2ddot]-L*qtilde;
a=[q1dddot;q2dddot]-L*qtildedot;
r=qtildedot+L*qtilde;

%Regressor
Y(1,1)=a(1);
Y(1,2)=q2^2*a(1)+q2*q2dot*v(1)+q2*q1dot*v(2)+q2*g*cos(q1);
Y(1,3)=2*q2*a(1)+q2dot*v(1)+q1dot*v(2)+g*cos(q1);
Y(1,4)=g*cos(q1);

Y(2,1)=0;
Y(2,2)=a(2)-q2*q1dot*v(1)+g*sin(q1);
Y(2,3)=-q1dot*v(1);
Y(2,4)=0;

TH_0=[39.62;8;2.4;3];    % Nominal parameters

deadzone=0.1;
rho=4.4826;
s=Y'*r;
if norm(s)>deadzone,
    dTh=-rho*s/norm(s);
else
    dTh=[0;0;0;0];
end
Th_hat=TH_0+dTh;

u=Y*Th_hat-K*r;
```

Simulink was used to set up a simulation of the robust passivity-based control for use in RP planar arm.

Figure 3 shows the Simulink diagram for a RP planar arm with the robust passivity-based control.

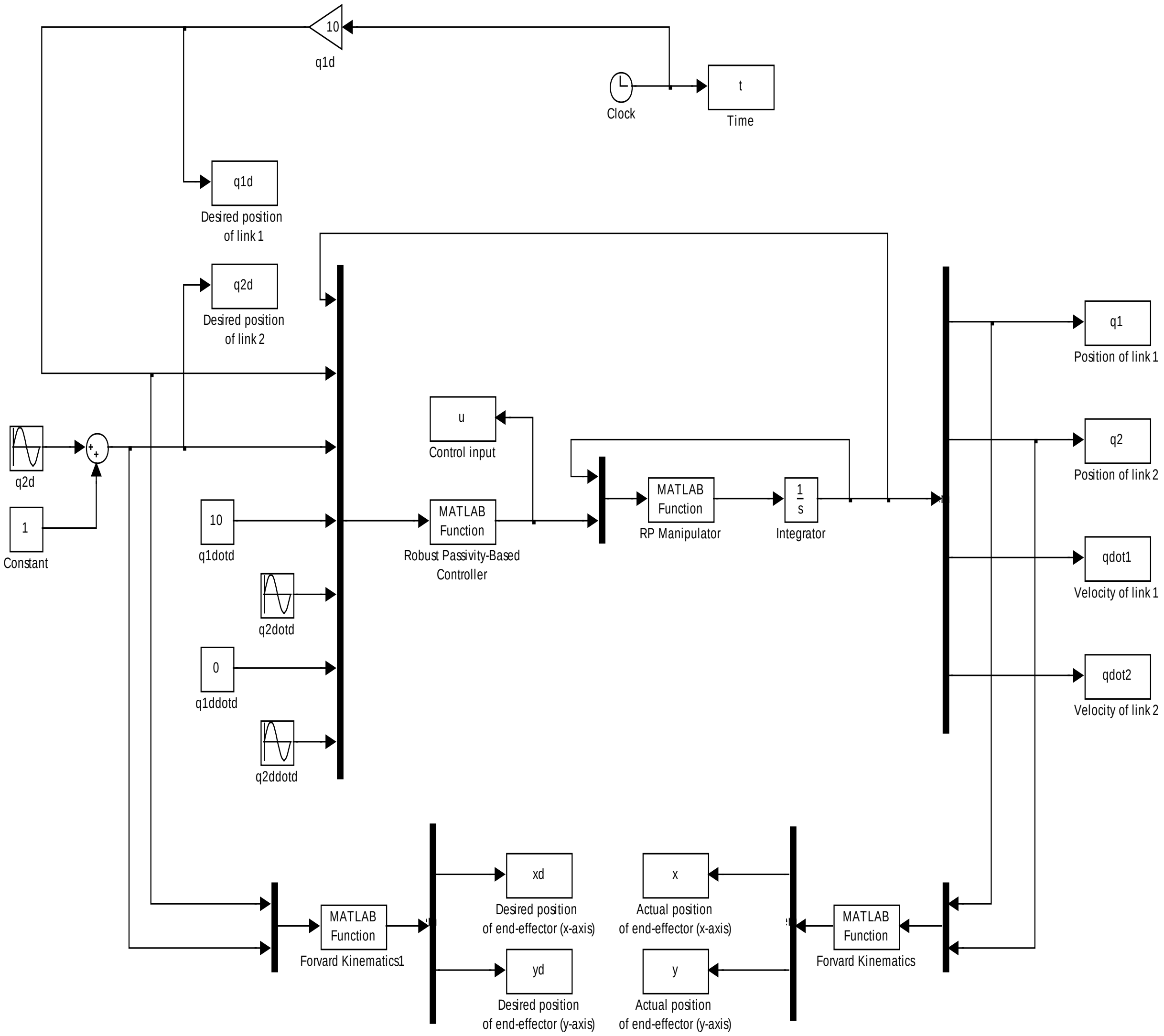


Figure 3. The Simulink diagram for a RP Planar Arm with the robust passivity-based control

Figure 4 shows a position error $\tilde{q}_1$ for link 1 (between actual position and desired)

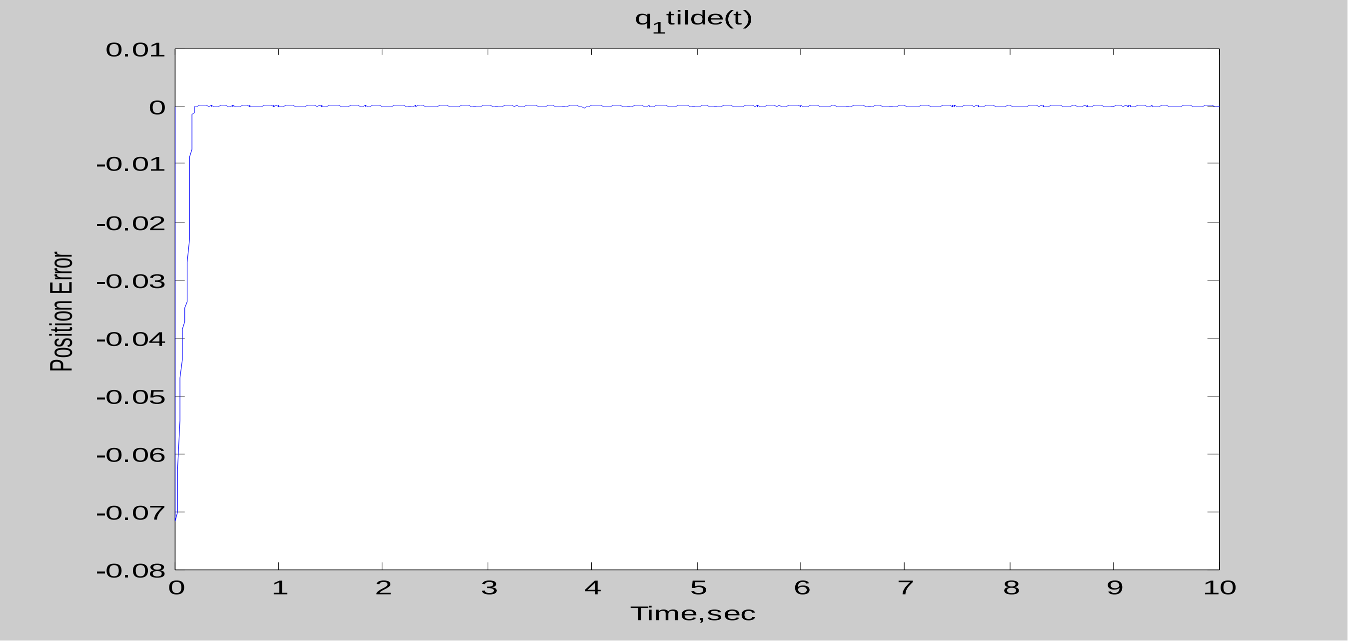


Figure 4. Position Error of Link 1

Figure 5 shows a position error $\tilde{q}_2$ of link 2.

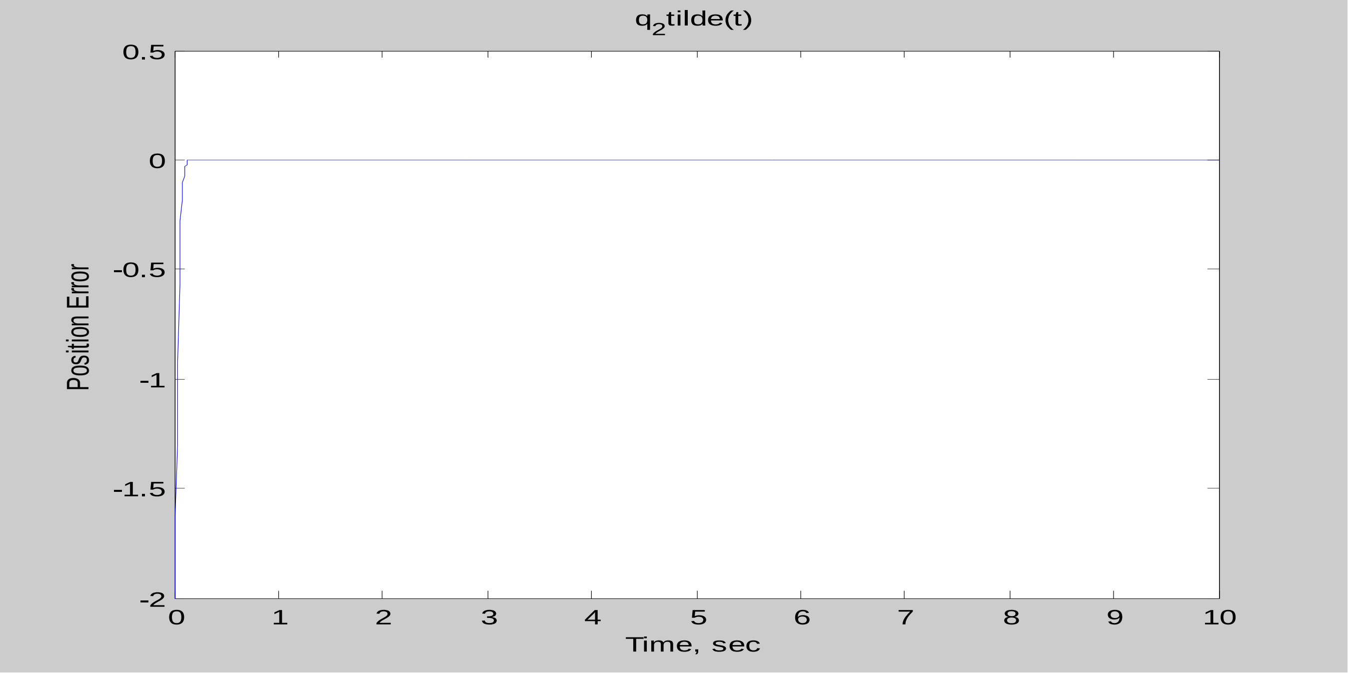


Figure 5. Position of link 2

Figure 6 shows plots of desired and actual positions of the end-effector of the RP planar arm using robust passivity-based control (10 % uncertainty).

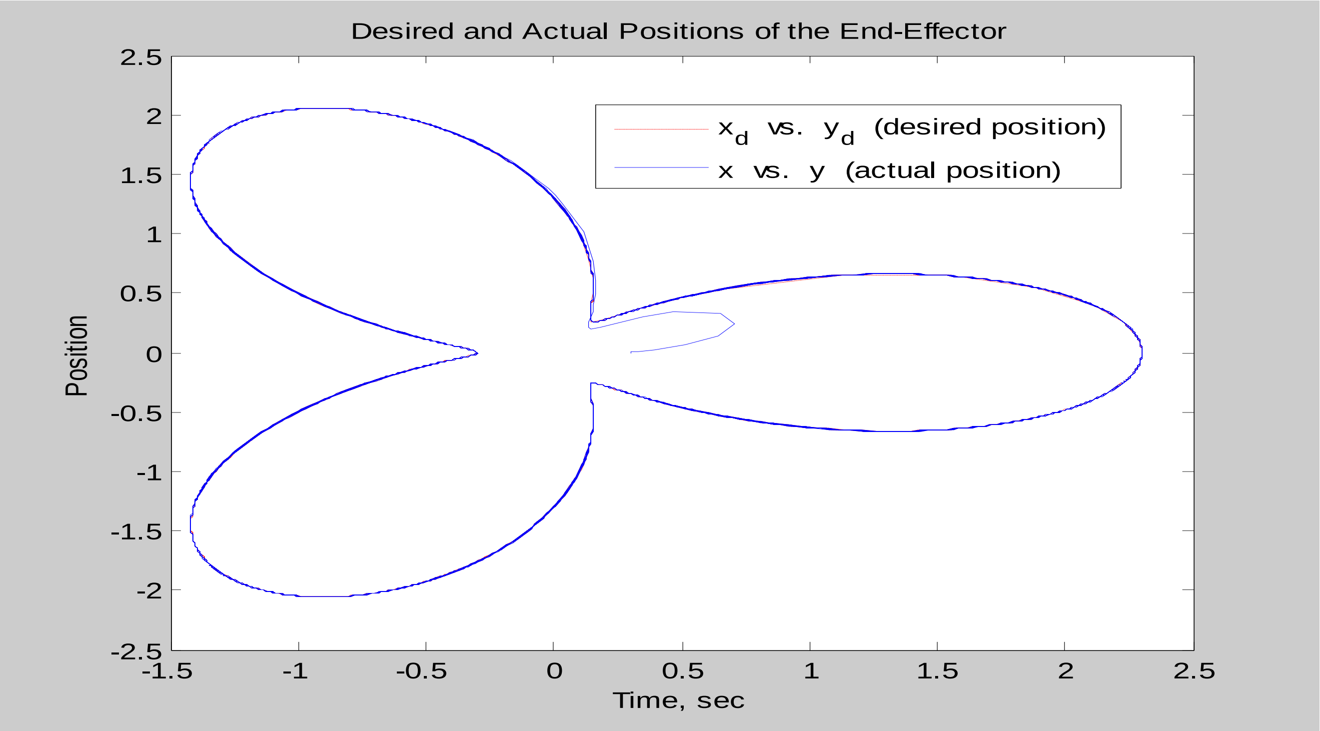


Figure 6. Desired and Actual Positions of the End-Effector Using Robust Passivity-Based Control

Figure 7 shows the control input for link 1.

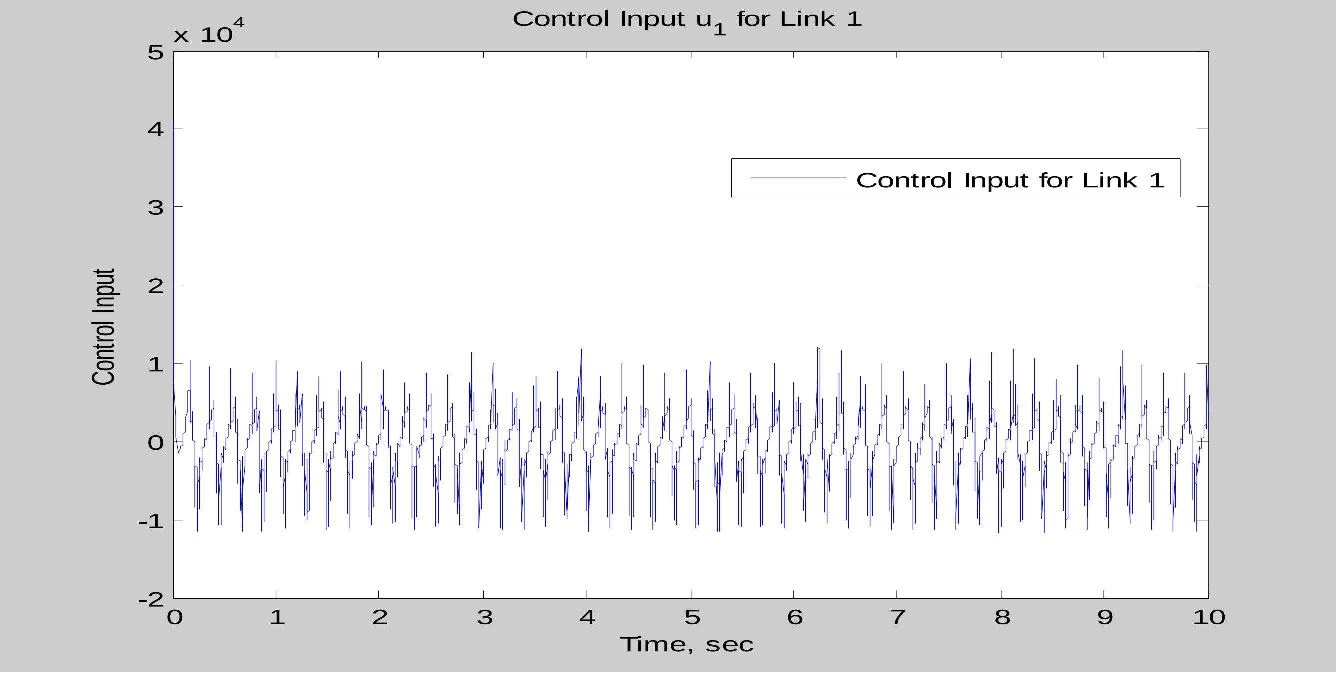


Figure 7. Control Input for Link 1

Figure 8 shows the control input for link 2.

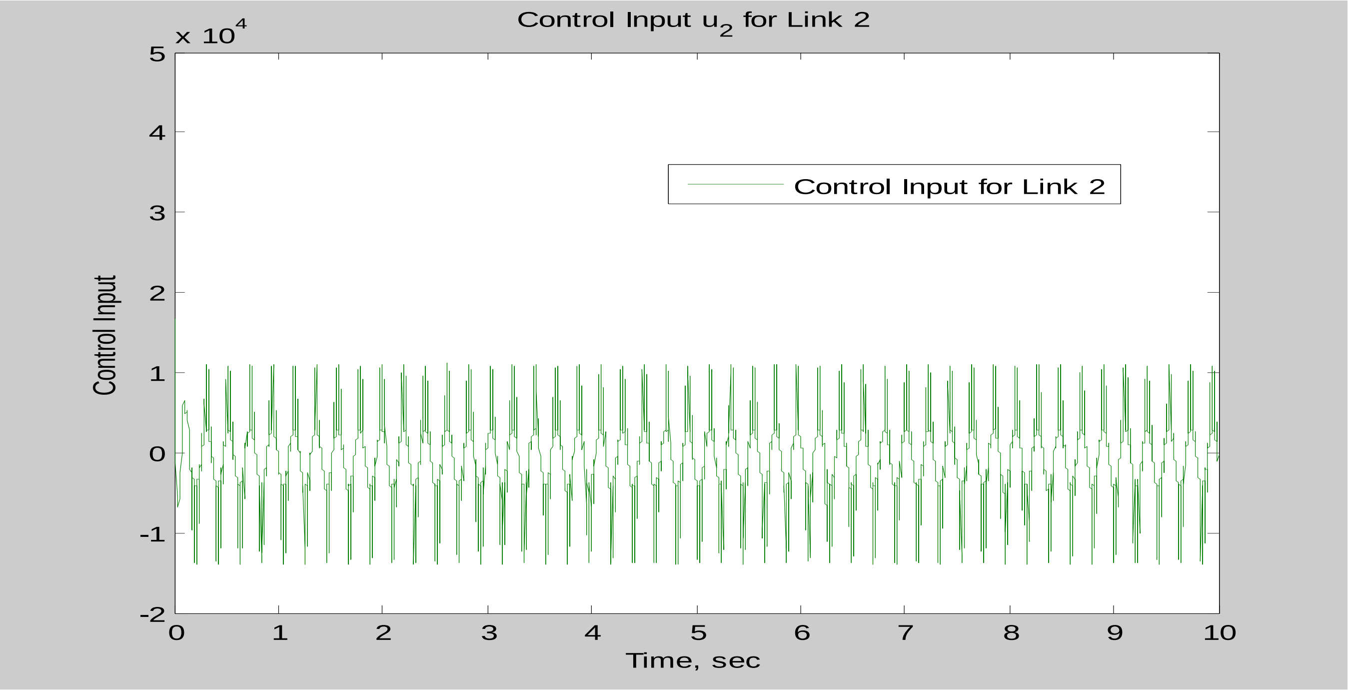


Figure 8. Control Input for Link 2

M-file was generated for use in the plant (25% uncertainty):

```
function qdot=plant_25_uncert(t,q,u)

m1=11.4643;
m2=7.5712;
a1=0.5610;
g=9.8;
Izz_1=27.6838;
Iyy_2=11.5491;

n=length(q);
q_1=q(1:n/2);
q_2=q(n/2+1:n);

D(1,1)=1/4*m1*a1^2+m2*q_1(2)^2+m2*q_1(2)*a1+1/4*m2*a1^2+Izz_1+Iyy_2;
D(1,2)=0;
D(2,1)=D(1,2);
D(2,2)=m2;

C(1,1)=(m2*q_1(2)+1/2*m2*a1)*(q_2(2));
C(1,2)=(m2*q_1(2)+1/2*m2*a1)*(q_2(1));
C(2,1)=(-m2*q_1(2)-1/2*m2*a1)*(q_2(1));
C(2,2)=0;
```

```
g_q(1,1)=1/2*g*m1*a1*cos(q_1(1))+m2*g*(1/2*a1+q_1(2))*cos(q_1(1));
g_q(2,1)=m2*g*sin(q_1(1));

qdot=[q_2;inv(D)*(u-C*q_2-g_q)];
```

M-file was generated for use in the control law:

```
function u=robust_passivity_25_uncert(t,z,zd)

g=9.8;
q1=z(1);
q2=z(2);
q1dot=z(3);
q2dot=z(4);

q1d=zd(1);
q2d=zd(2);
q1ddot=zd(3);
q2ddot=zd(4);
q1dddot=zd(5);
q2dddot=zd(6);

qtilde=[q1-q1d;q2-q2d];
qtildedot=[q1dot-q1ddot;q2dot-q2ddot];

%Gains
L=diag([100 100]);
K=diag([100 100]);

v=[q1ddot;q2ddot]-L*qtilde;
a=[q1dddot;q2dddot]-L*qtildedot;
r=qtildedot+L*qtilde;

%Regressor
Y(1,1)=a(1);
Y(1,2)=q2^2*a(1)+q2*q2dot*v(1)+q2*q1dot*v(2)+q2*g*cos(q1);
Y(1,3)=2*q2*a(1)+q2dot*v(1)+q1dot*v(2)+g*cos(q1);
Y(1,4)=g*cos(q1);
Y(2,1)=0;
Y(2,2)=a(2)-q2*q1dot*v(1)+g*sin(q1);
Y(2,3)=-q1dot*v(1);
Y(2,4)=0;

TH_0=[39.62;8;2.4;3];      % Nominal parameters

deadzone=0.1;
rho=11.4299;
s=Y'*r;
if norm(s)>deadzone,
    dTh=-rho*s/norm(s);
else
    dTh=[0;0;0;0];
end
```

```
Th_hat=TH_0+dTh;
u=Y*Th_hat-K*r;
```

Figure 9 shows a position error $\tilde{q}_1$ for link 1 (between actual position and desired)

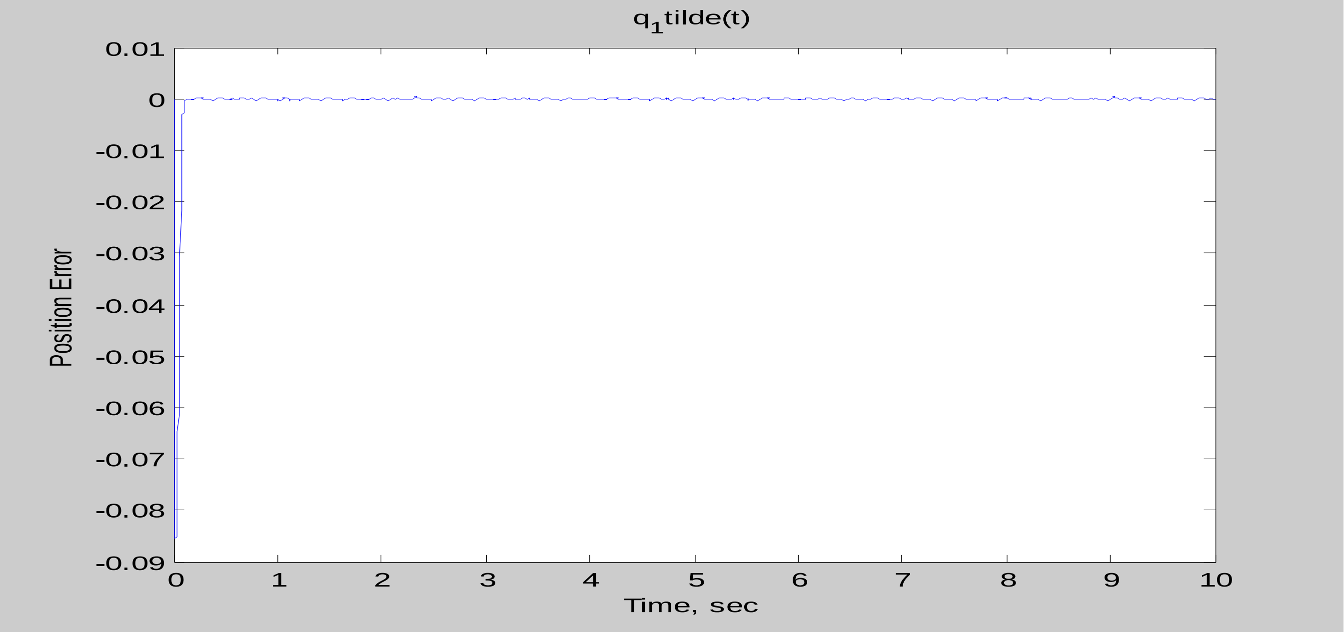


Figure 9. Position Error of Link 1

Figure 10 shows a position error $\tilde{q}_2$ for link 2 (between actual position and desired)

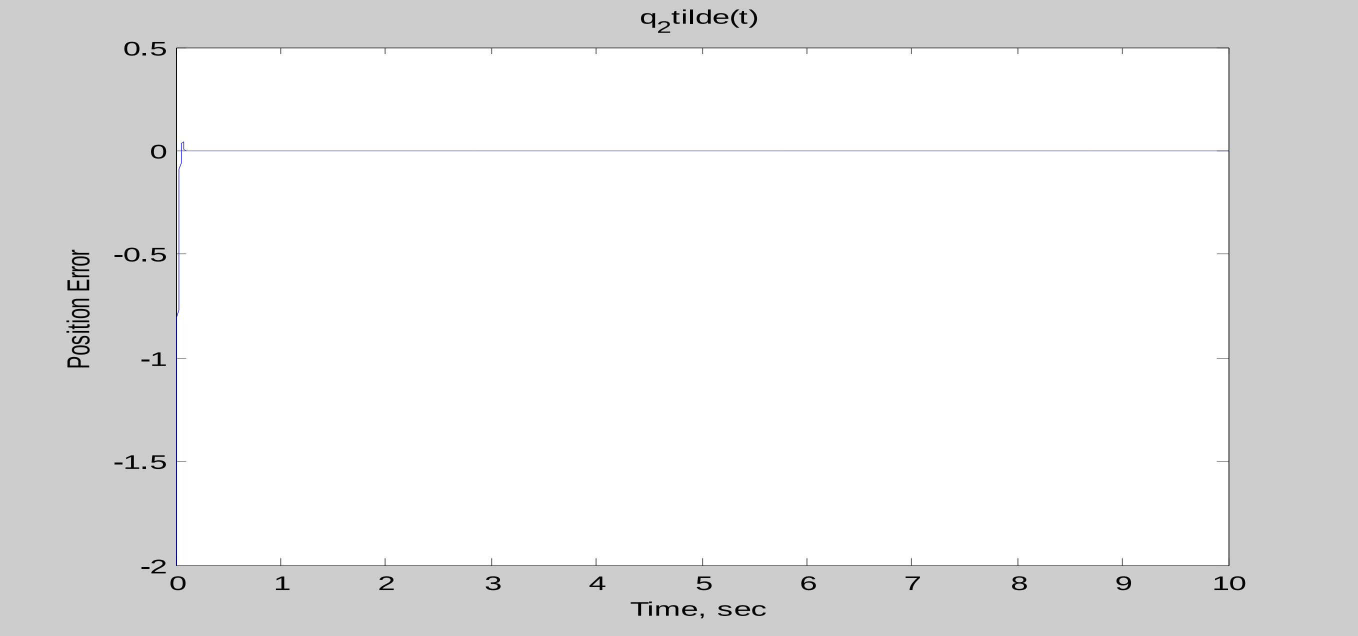


Figure 10. Position Error of Link 2

Figure 11 shows plots of desired and actual positions of the end-effector of the RP planar arm using the robust passivity-based control (25% uncertainty).

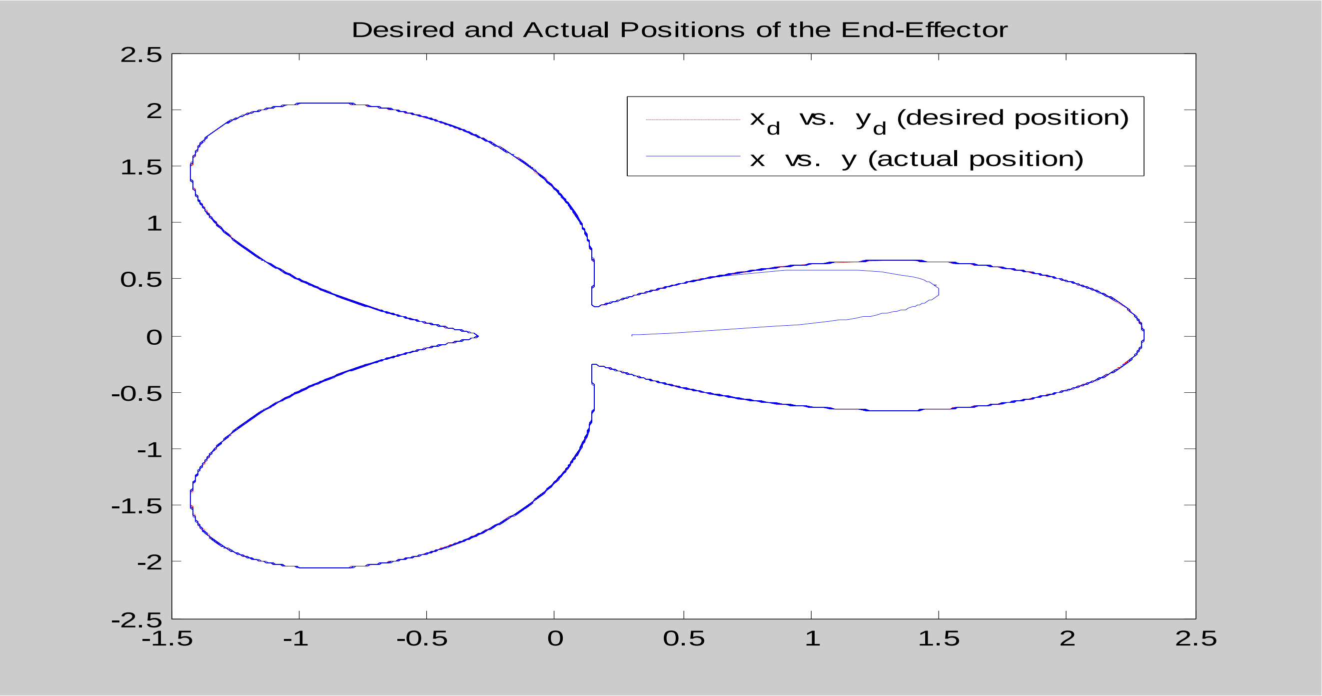


Figure 11. Desired and Actual Positions of the End-Effector Using Robust Passivity-Based Control

Figure 12 shows the control input for link 1.

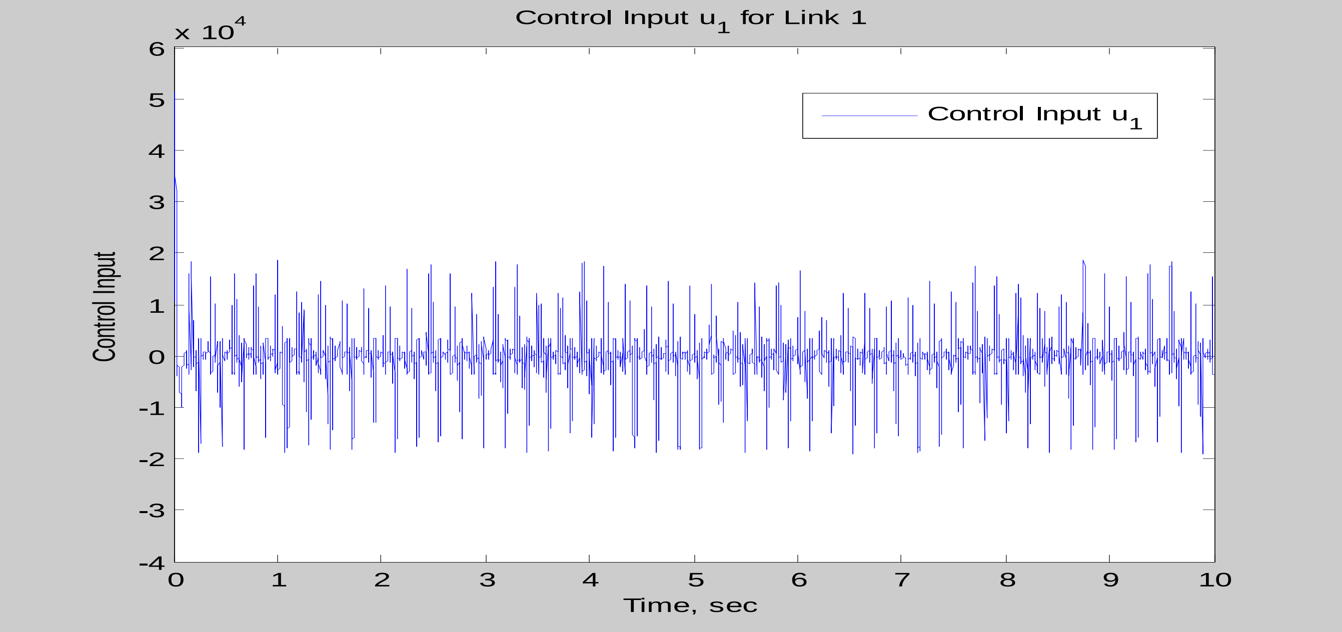


Figure 12. The Control Input for Link 1

Figure 13 shows the control input for link 2.

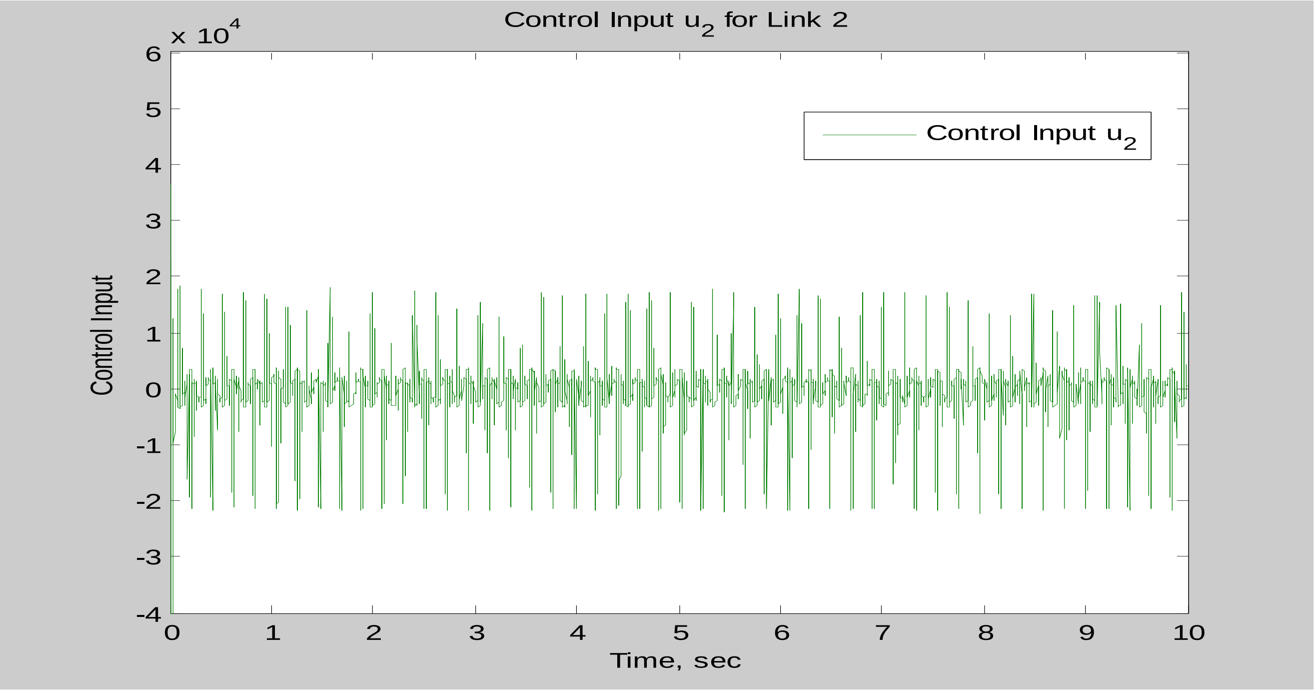


Figure 13. The Control Input for Link 2

Adaptive Passivity-Based Control

M-file was generated for use in the plant (80% uncertainty):

```
function qdot=plant_80_uncert(t,q,u)

m1=8.2634;
m2=14.1983;
a1=1.0643;
g=9.8;
Izz_1=38.8929;
Iyy_2=10.3312;

n=length(q);
q_1=q(1:n/2);
q_2=q(n/2+1:n);

D(1,1)=1/4*m1*a1^2+m2*q_1(2)^2+m2*q_1(2)*a1+1/4*m2*a1^2+Izz_1+Iyy_2;
D(1,2)=0;
D(2,1)=D(1,2);
D(2,2)=m2;
```

```
C(1,1)=(m2*q_1(2)+1/2*m2*a1)*(q_2(2));
C(1,2)=(m2*q_1(2)+1/2*m2*a1)*(q_2(1));
C(2,1)=(-m2*q_1(2)-1/2*m2*a1)*(q_2(1));
C(2,2)=0;

g_q(1,1)=1/2*g*m1*a1*cos(q_1(1))+m2*g*(1/2*a1+q_1(2))*cos(q_1(1));
g_q(2,1)=m2*g*sin(q_1(1));

qdot=[q_2;inv(D)*(u-C*q_2-g_q)];
```

M-file was generated for use in the control law:

```
function u=adaptive_passivity(t,z,zd,Th_hat)

g=9.8;
q1=z(1);
q2=z(2);
q1dot=z(3);
q2dot=z(4);

q1d=zd(1);
q2d=zd(2);
q1ddot=zd(3);
q2ddot=zd(4);
q1dddot=zd(5);
q2dddot=zd(6);

qtilde=[q1-q1d;q2-q2d];
qtildedot=[q1dot-q1ddot;q2dot-q2ddot];

%Gains
L=diag([500 500]);
K=diag([500 500]);

v=[q1ddot;q2ddot]-L*qtilde;
a=[q1dddot;q2dddot]-L*qtildedot;
r=qtildedot+L*qtilde;

%Regressor
Y(1,1)=a(1);
Y(1,2)=q2^2*a(1)+q2*q2dot*v(1)+q2*q1dot*v(2)+q2*g*cos(q1);
Y(1,3)=2*q2*a(1)+q2dot*v(1)+q1dot*v(2)+g*cos(q1);
Y(1,4)=g*cos(q1);
Y(2,1)=0;
Y(2,2)=a(2)-q2*q1dot*v(1)+g*sin(q1);
Y(2,3)=-q1dot*v(1);
Y(2,4)=0;

u=Y*Th_hat-K*r;
```

```
function th_hat_dot=parameter_adaptive(t,z,zd)

g=9.8;
q1=z(1);
q2=z(2);
q1dot=z(3);
q2dot=z(4);

q1d=zd(1);
q2d=zd(2);
q1ddot=zd(3);
q2ddot=zd(4);
q1dddot=zd(5);
q2dddot=zd(6);

qtilde=[q1-q1d;q2-q2d];
qtildedot=[q1dot-q1ddot;q2dot-q2ddot];

%Gains
L=diag([500 500]);
G=eye(4);

v=[q1ddot;q2ddot]-L*qtilde;
a=[q1dddot;q2dddot]-L*qtildedot;
r=qtildedot+L*qtilde;

% Regressor
Y(1,1)=a(1);
Y(1,2)=q2^2*a(1)+q2*q2dot*v(1)+q2*q1dot*v(2)+q2*g*cos(q1);
Y(1,3)=2*q2*a(1)+q2dot*v(1)+q1dot*v(2)+g*cos(q1);
Y(1,4)=g*cos(q1);

Y(2,1)=0;
Y(2,2)=a(2)-q2*q1dot*v(1)+g*sin(q1);
Y(2,3)=-q1dot*v(1);
Y(2,4)=0;

th_hat_dot=-G*Y'*r;
```

Simulink was used to set up a simulation of the adaptive passivity-based control for use in RP planar arm.

Figure 14 shows the Simulink diagram for a RP Planar Arm with the adaptive passivity-based control.

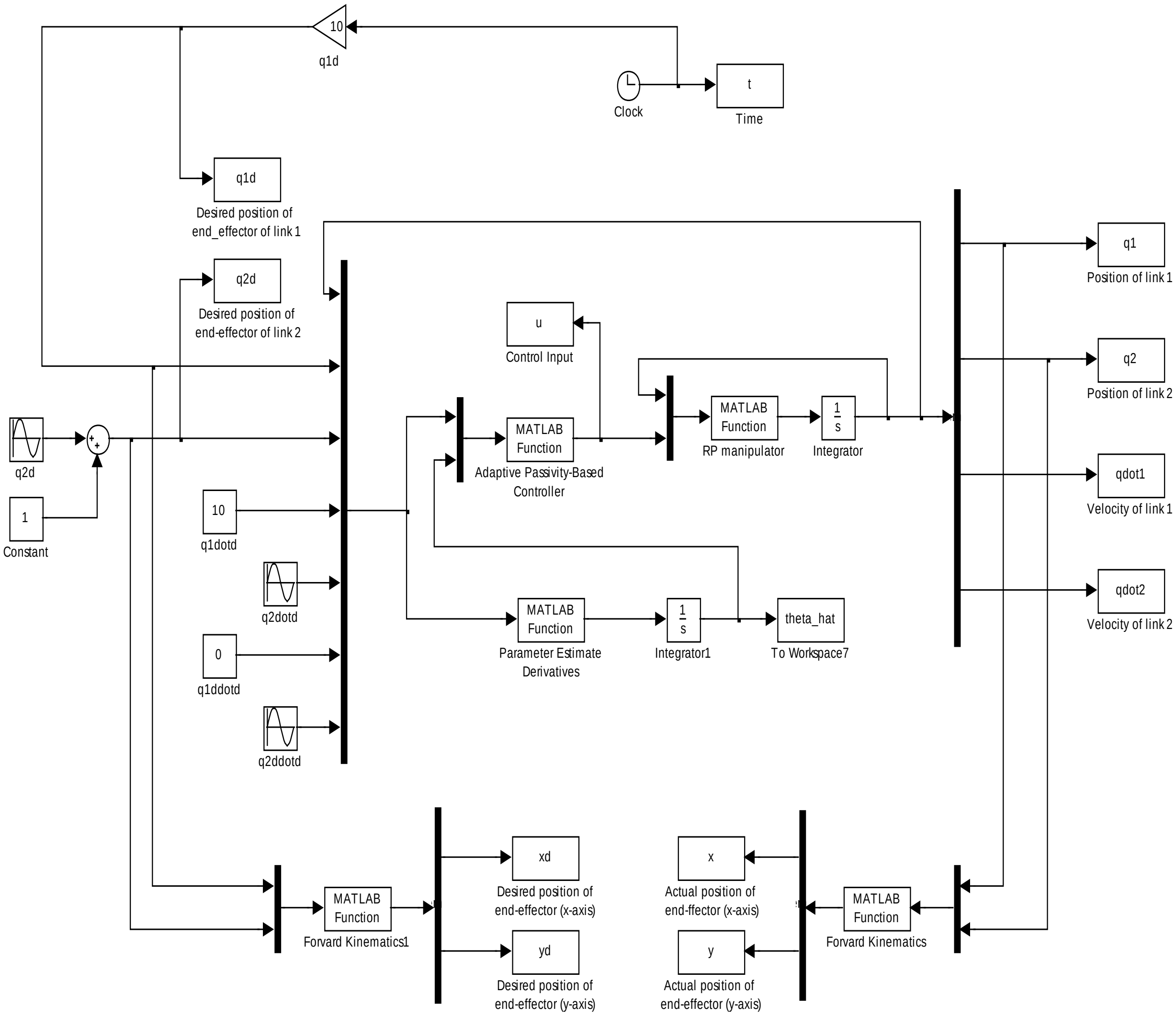


Figure 14. The Simulink diagram for a RP Planar Arm with the adaptive passivity-based control

Figure 15 shows plot of adapted parameter $\hat{\theta}_1$ versus time.

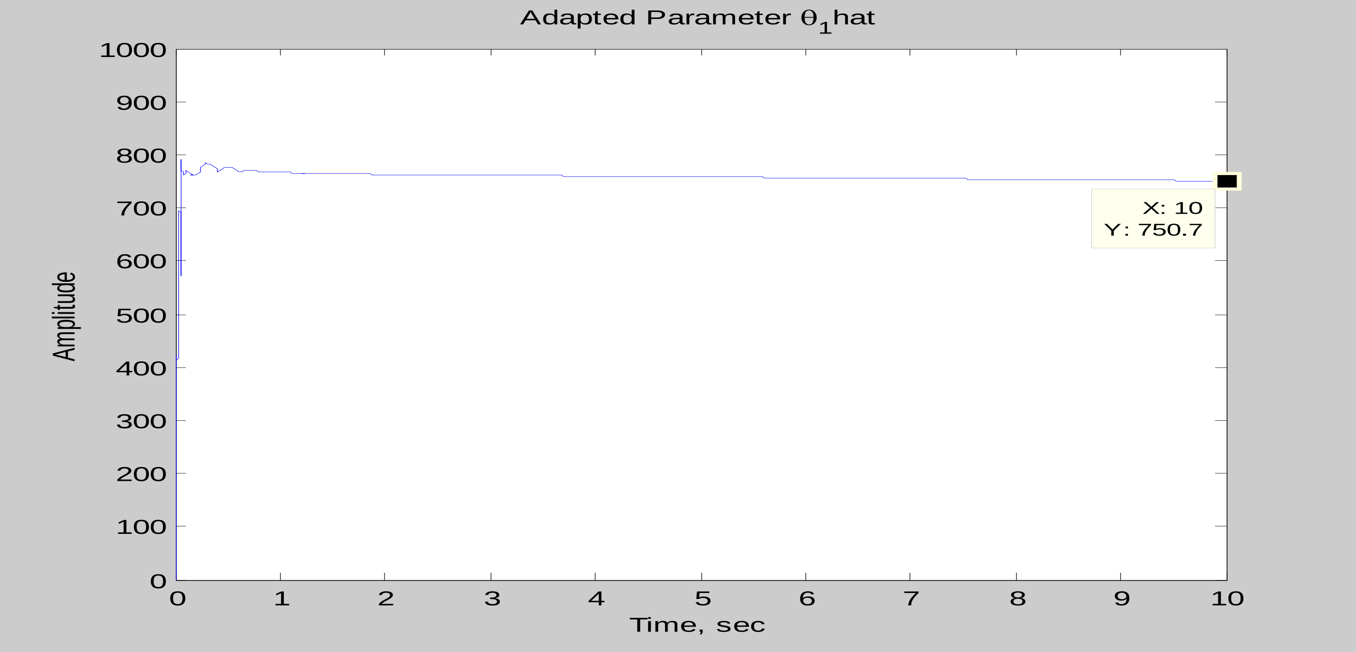


Figure 15. Adapted parameter $\hat{\theta}_1$

Figure 16 shows plot of adapted parameter $\hat{\theta}_2$versus time.

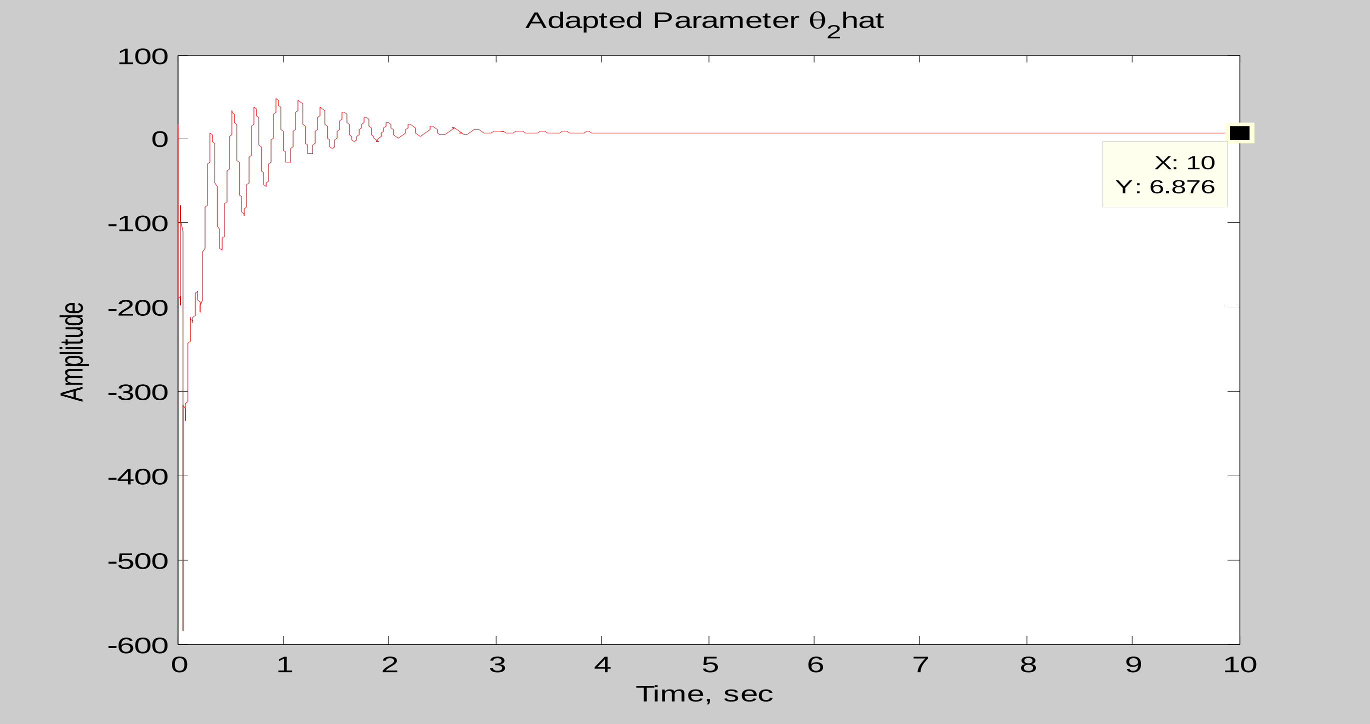


Figure 16. Adapted parameter $\hat{\theta}_2$

Figure 17 shows plot of adapted parameter $\widehat{\theta}_3$versus time.

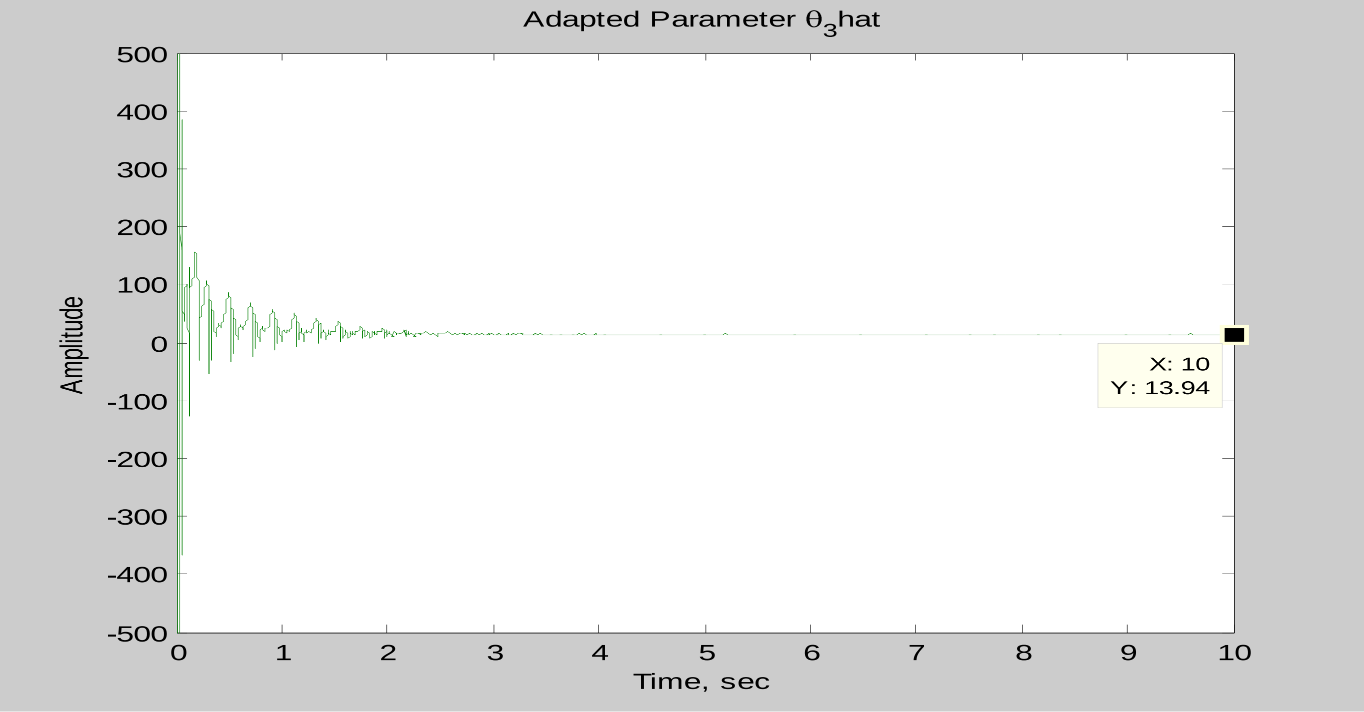


Figure 17. Adapted parameter $\widehat{\theta}_3$

Figure 18 shows plot of adapted parameter $\widehat{\theta}_4$versus time.

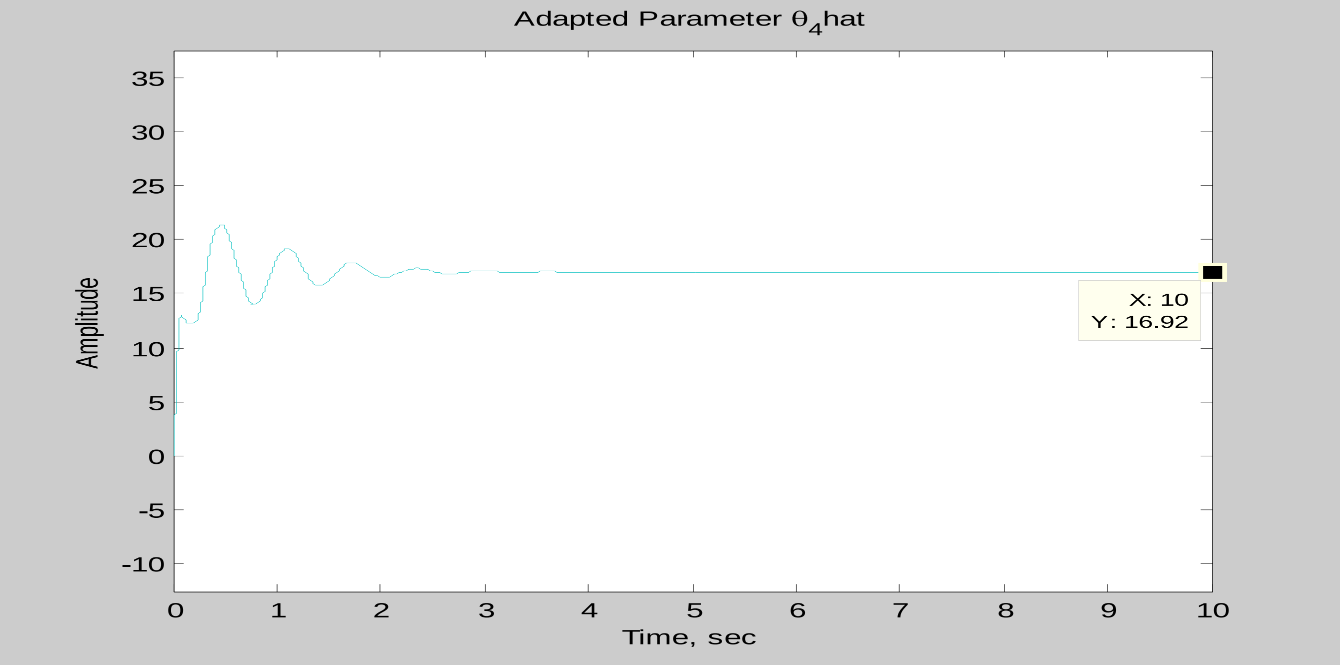


Figure 18. Adapted parameter $\widehat{\theta}_4$

Figure 19 shows a position error $\tilde{q}_1$ for link 1 (between actual position and desired)

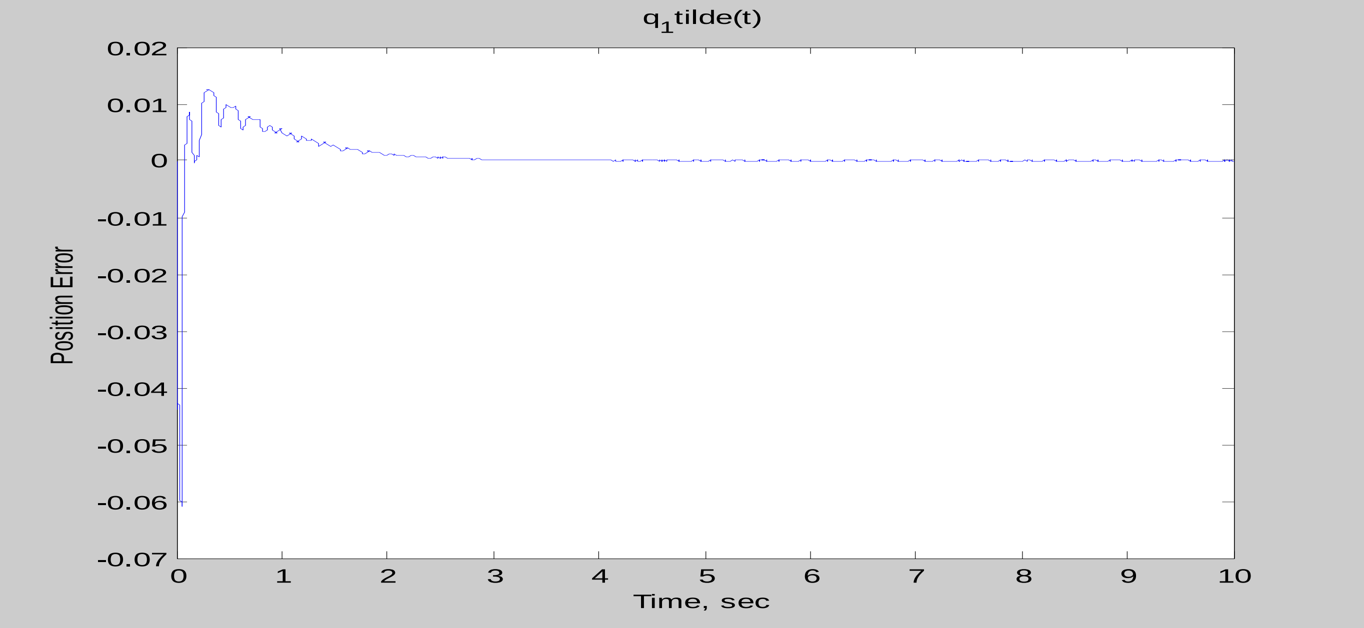


Figure 19. Position Error of Link 1

Figure 20 shows a position error $\tilde{q}_2$ for link 2 (between actual position and desired)

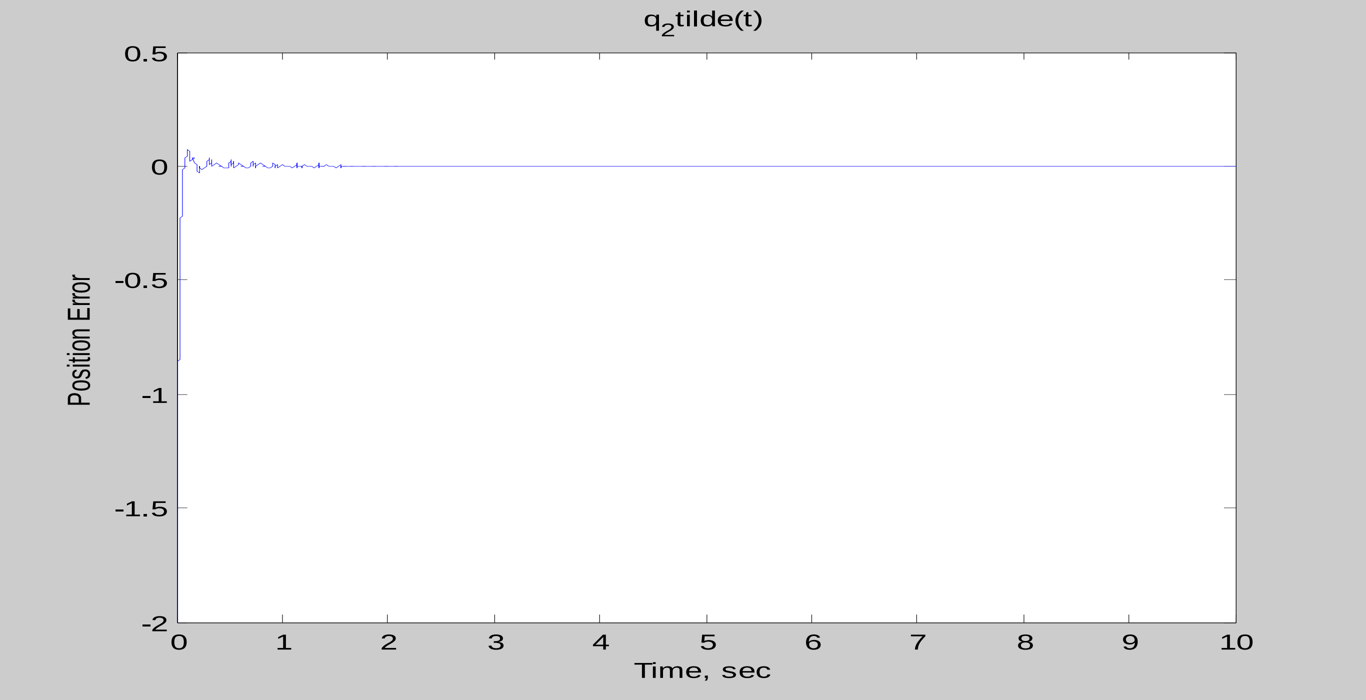


Figure 20. Position Error of Link 2

Figure 21 shows plots of desired and actual positions of the end-effector of the RP planar arm using the adaptive passivity-based control.

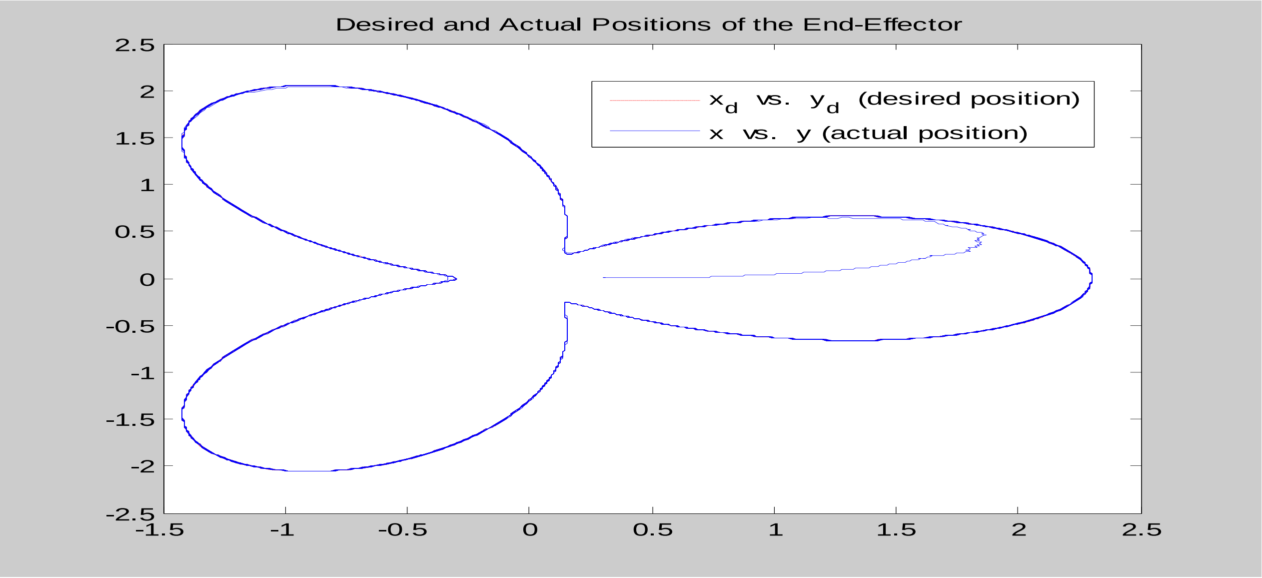


Figure 21. Desired and Actual Positions of the End-Effector Using the Adaptive Passivity-Based Control

Figure 22 shows the control input for link 1.

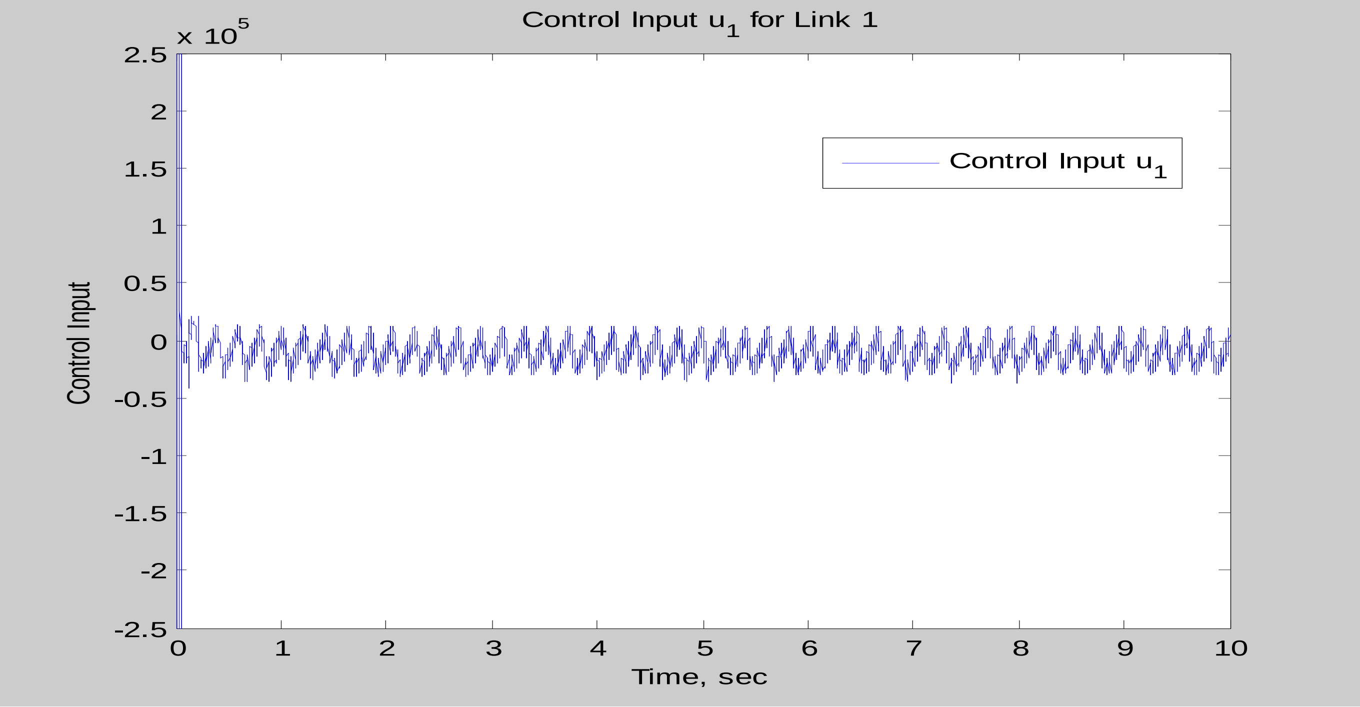


Figure 22. The control input for link 1

Figure 23 shows the control input for link 2.

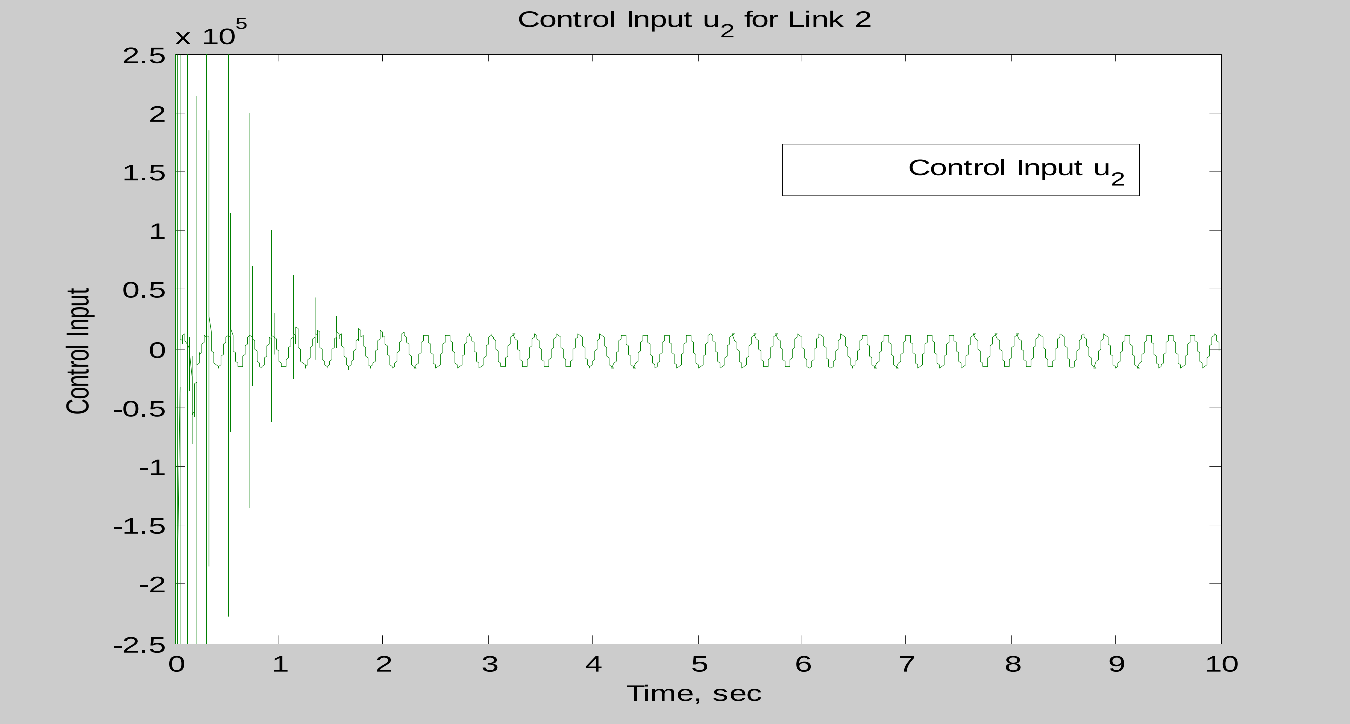


Figure 23. The control input for link 2

The nominal parameters vector is TH_0=$[39.62; 8; 2.4; 3]^T$. Using figures 15-18 we can conclude that the parameters did not converge to their true values.