پروژه پایانی درس کنترل مدرن

کنترل موقعیت ربات از راه دور

استاد درس

دکتر ایمان شریفی

تهیه کنندگان :

یاسین دهفولی 9623048

عارفه کوهی 9623095

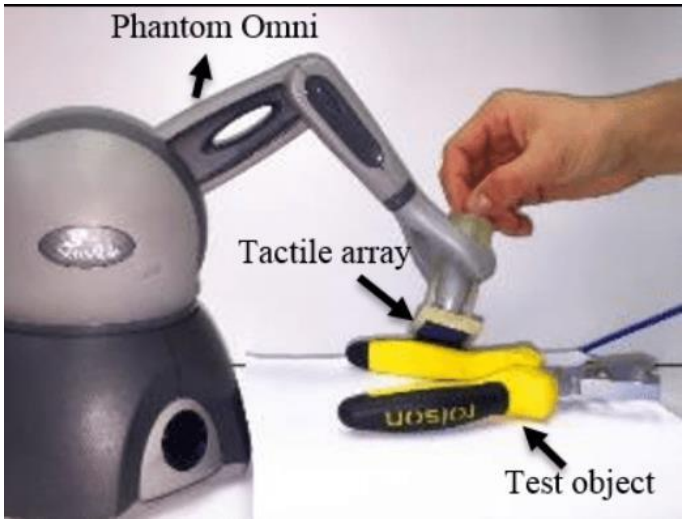
درسا نظری 9623113

سپهر قمری 9623090

بخش اول

الف)معرفی سیستم،عملگر ها و سنسور های آن

سنسور های مورد استفاده در این ربات سنسور tactileرا میتوانیم به ان اضافه کنیم.در واقع کاربرد این سنسور به این شکل میباشد که در مصارف گوناگون صنعتی وپزشکی و دندان پزشکی برای اندازه گیری فشار ونیرو قابل استفاده هستند و در رنج ادراک یک تا 100هزار نقطه را شامل میشود



معرفی سیستم:

سیستم مورد بحث در این پروژه،از دو ربات **PHANTOM omni**تشکیل شده است.کاربر با حرکت دادن بازوی ربات ، **Master**فرمان خود را برای ربات **Slave**صادر کرده و این فرمان(نیروی وارد شده به بازوی مصنوعی ربات فرمانده) از طریق کانال ارتباطی به **slave**میرسد.

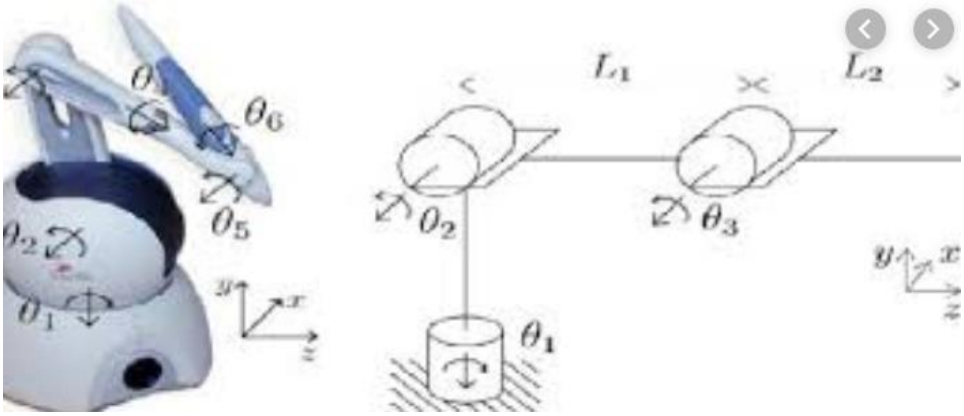
آ--2بیان معادله غیرخطی سیستم و پارامتر های آن

مدل دینامیکی ربات **phantom omni**که بصورت غیر خطی تعریف شده به شکل زیر است:

$$M(q)\ddot{q}+C(q,\dot{q})\dot{q}+G(q)+d=\tau$$

که در این معادله **q**یک ماتریس **R*1**که موقعیت زاویه ای و ماتریس M(q) یک ماتریس **N*N** این ماترسی یک ماتریس **positive definite**میباشد.

ماتریس **c**ماتریس گریز از مرکز است و یک ماتریس **N*1**است و ماتریس **taw**که گشتاور است و یک ماتریس **N*1**است.



L1=L2=135mm

معادلات بالا را میتوان ساده تر کرد و به شکل زیر نوشت:

$V(q,\dot{q})=C(q,\dot{q})\dot{q}+G(q)$ که چنین نتیجه میدهد:

$$M(q)\ddot{q}+V+d=\tau$$

معادلات دینامیکی سیستم **master , slave**میتوان به شکل زیر نوشت:

$$M_m(q_m)\ddot{q}_m+V_m(q_m,\dot{q}_m)=\tau_m+f_m+d_m$$
$$M_s(q_s)\ddot{q}_s+V_s(q_s,\dot{q}_s)=\tau_s-f_s+d_s$$

که در ان **fm**نیرویی است که اپراتور به ربات **master**وارد میکند و **fs**نیرویی است که محیط به ربات **slave** وارد میکند.و ماتریس **vm , vs**که بیانگر نیروی گریز از مرکز نیروی گرانش است و همچنین **dm , ds**بیانگر **disturbance** است.

معادله

$$V(q,\dot{q}) = [V_1, V_2]^T = C(q,\dot{q})\dot{q} + G(q)$$

برای ساده سازی سیمولینک درجه ازادی دوم ربات های **master , slave**حذف شده است و با جایگذاری درعبارت بالا حاصل میشود.

$$V_1 = -2a_1\dot{q}_1\dot{q}_3\sin(2q3) + 2a_3\dot{q}_1\dot{q}_3\cos(2q3) + a_4\dot{q}_1\dot{q}_3\cos(q3) - a_5\dot{q}_1\dot{q}_3\sin(q3)$$

$$V_2 = 2a_2\dot{q}_1^2\cos(q3)\sin(q3) - a_3\dot{q}_1^2\cos(2q3) - \frac{1}{2}a_4\dot{q}_1^2\cos(q3) + \frac{1}{2}a_5\dot{q}_1^2\sin(q3) + a_7\sin(q3) + a_8\cos(q3)$$

ماتریس **M(q)** به شکل زیر است:

$$M(q) = \begin{bmatrix} a_1 + a_2c_{23} + a_3s_{23} + a_4c_3 + a_5s_3 & 0 \\ 0 & a_6 \end{bmatrix}$$

$$c_i = \cos(q_i), s_i = \sin(q_i),$$

$$c_{2i} = \cos(2q_i), s_{2i} = \sin(2q_i), \quad i = 1,2,3\ldots$$

جدول پارامترهای ربات به شکل زیر تعریف شده است:

parameters	value	parameters	value
a_1	$6.11\times10^{-3}\pm0.9\times10^{-3}$	a_2	$-2.89\times10^{-3}\pm0.43\times10^{-3}$
a_3	$-4.24\times10^{-3}\pm1.01\times10^{-3}$	a_4	$3.01\times10^{-3}\pm0.52\times10^{-3}$
a_5	$2.05\times10^{-3}\pm0.15\times10^{-3}$	a_6	$1.92\times10^{-3}\pm0.23\times10^{-3}$
a_7	$1.60\times10^{-3}\pm0.05\times10^{-3}$	a_8	$-8.32\times10^{-3}\pm2.78\times10^{-3}$

در این پروژه دو کنترلر استفاده میشود**pid , backstopping controller** . کنترلر **pid** جهت کنترل نیروی اعمالی برای ربات ، **master** کنترلر **backstopping** برای کنترل موقعیت یا **slave** برای ربات **position tracking** طراحی میشود.

کنترلر pid

کنترلرهای pidبطور وسیعی در سیستم های پیچیده force trackingاستفاده میشوندبه همین منظور از این این کنترلر برای کنترل positionرباتmasterاستفاده میشود.

همان طور که میدانید کنترلر pidبه شکل زیر تعریف میشود

$$\tau_m = k_p e_f + k_i \int e_f dt + k_d \dot{e}_f$$

که در انerror :

$$e_f = f_m - f_s$$

و به محض ایجاد **error**کنترلر به گونه ای رفتار میکند که ان را کاهش دهد.

توجه کنید که ضرایب **kp , ki , kd** همگی ثابت هستند.

کنترلر backstopping

در واقع نظریه این کنترلر تجزیه ی سیستم های غیر خطی پیچیده به یک سری زیر سیستم تبدیل میکند که درجه ان از حدود سیستم تجاوز نمیکند.

کنترلر **backstopping** اجازه میدهد که سیستم شرایط پایداری تابع لیاپانوف را ارضا کند برای همین است که از این کنترلر استفاده میکنند.

تعریف میشود که:

$$x_1 = q_s = \begin{bmatrix} q_{s1} \\ q_{s3} \end{bmatrix} \quad x_2 = \dot{q}_s = \begin{bmatrix} \dot{q}_{s1} \\ \dot{q}_{s3} \end{bmatrix}$$

که در انها **q1 , q2**و **q1-dot , q2-dot**سرعت زاویه ای انها میباشد.

نیرویی که محیط به **slave**وارد میکند به شکل زیر است که در ان **bw**ضریب میرایی و **cw**ضریب الاستیسیته است.

$$f_s = b_w \frac{dx_s}{dt} + c_w x_s$$

متغیر های تعریف شده بالا را درمعادلات دینامیکی جاگذاری کرده

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = f(x,t) + b(x,t)u$$
$$b(x,t) = M_s^{-1}(x_1)$$
$$f(x,t) = -M_s^{-1}(x_1)V_s - M_s^{-1}(x_1)f_s + M_s^{-1}d_s$$

هدف کنترلی ربات phantom omni

یکی از زیر شاخه های کاربردی علم رباتیک و مبحث کنترل از راه دور ربات ها در یک محیط غیر قابل دسترس می باشد.اپراتور انسانی با تاثیر گذاری بر ربات راهبر توانایی انتقال کنش خود را به محیط مورد نظر توسط ربات رهرو بدست می اورد . بصورت کلی چالش های اصلی این سیستم عبارتند از تاخیر شبکه مخابراتی بین ربات راهبر و رهرو و هم چنین انتقال حس لامسه از طریق کنترل نیرو و یا امپدانس ربات ها که باعث افزایش دقت و صحت عملیات انجام شده از راه دور اشاره کرد.شایان ذکر است که مبحث خمش بازو هنگام ظریف سازی ابزار در اعمال جراحی رباتیکی میباشد که در طی چند سال اخیر توجه بسیاری از محققین این حوزه را به خود جلب کرده است.

خطی سازی و بدست آوردن نقطه تعادل

برای خطی سازی سیستم فوق باید یک نقطه تعادل برای سیستم بیابیم بنابراین کد های زیر را پیاده سازی کردیم:

بدست اوردن نقطه تعادل:

درواقع برای بدست اوردن نقطه تعادل سیستم لازم است که مشتق حالت ها را مساوی صفر بگذاریم که انجام این کار تنها به یک نقطه میرسیم که همان صفر است.

```
%Equilibirum
%%x1_dot = [q1_dot;q3_dot]  q1_dot = n3  q3_dot = n4
x1_dot = [n3 ; n4];
%%x2_dot = [q1_ddot;q3_ddot]
x2_dot = -M_inv*V - M_inv*f + M_inv*u + M_inv*d;

n_dot = [x1_dot ; x2_dot];
```

خطی سازی:

```
%Linearization:
SOLV = solve(n_dot == 0 ,[n1 n2 n3 n4 u1 u2 d1 d2]);
%Equilibirum
EQ = [SOLV.p1  SOLV.p2  SOLV.p3  SOLV.p4 SOLV.u1 SOLV.u2 SOLV.d1 SOLV.d2];
%linearization
var = [n1 n2 n3 n4 u1 u2 d1 d2];
JAC_A = jacobian(n_dot,[n1 n2 n3 n4]);
A = subs(JAC_A,var,EQ);
JAC_B = jacobian(n_dot,u);
B = subs(JAC_B, var,EQ);
B = vpa(B,3);
JAC_Bd = jacobian(n_dot,d);
Bd = subs(JAC_Bd,var,EQ);
```

بخش دوم پروژه درس کنترل مدرن – کنترل موقعیت ربات از راه دور

1) پس از خطی کردن سیستم ، زیر سیستم مینیمال(کاهش ناپذیر) محاسبه شد.

```
A =

   1.0e+05 *

         0         0    0.0000         0
         0         0         0    0.0000
   -2.4077         0   -0.0482         0
         0   -7.8125         0   -0.1563

B =

         0         0
         0         0
  160.5136         0
         0  520.8333

C =

     1     0     0     0
     0     1     0     0
```

<<<<<<<<

```
>> rank(Controllability)
ans =
     4

>> rank(Observability)
ans =
     4
```

2) قطب های مطلوب و طراحی فیدبک حالت

با در نظر گرفتن شرایط زیر :

3= 0.5

$T_s = 1\ s$

قطب های مطلوب سیستم به صورت زیر به دست آمد:

```
pc = [-4+0.86j -4-0.86j -80 -90];
```

وهمچنین :

```
k = place(A,B,pc);
```

<<<<<<<<

k =

1.0e+03 *

-1.4980 -0.0004 -0.0295 -0.0000

0.0001 -1.4993 0.0000 -0.0298

امتیازی :

سعی کردیم برای یک سیستم siso دلخواه با روش بس و گیورا و اکرمن k رو محاسبه کنیم

```
tic
disp(' place in MATLAB')
pc=[-1,-2,-3];
K0 = place(A, b, pc)
toc
```

<<<<<<<<

= K0

0.5789 1.0526 1.2632

```
tic
disp(' Bass-Gura Formula')
alpha = [1 6 11 6];
a = denOL;
Omega = [1 a(2) a(3);0 1 a(2);0 0 1];
K1 = (alpha(2:4)-a(2:4))*inv(Omega)*inv(Wc)
toc
```

<<<<<<<<

= K1

0.5789 1.0526 1.2632

```
tic
disp(' Ackermann Formula')
alpha = [1 6 11 6];
alpha_of_A = zeros(3,3);
for i=1:4
alpha_of_A = alpha_of_A + alpha(i)*A^(4-i);
end
K2 = [0 0 1]*inv(Wc)*alpha_of_A
toc
```

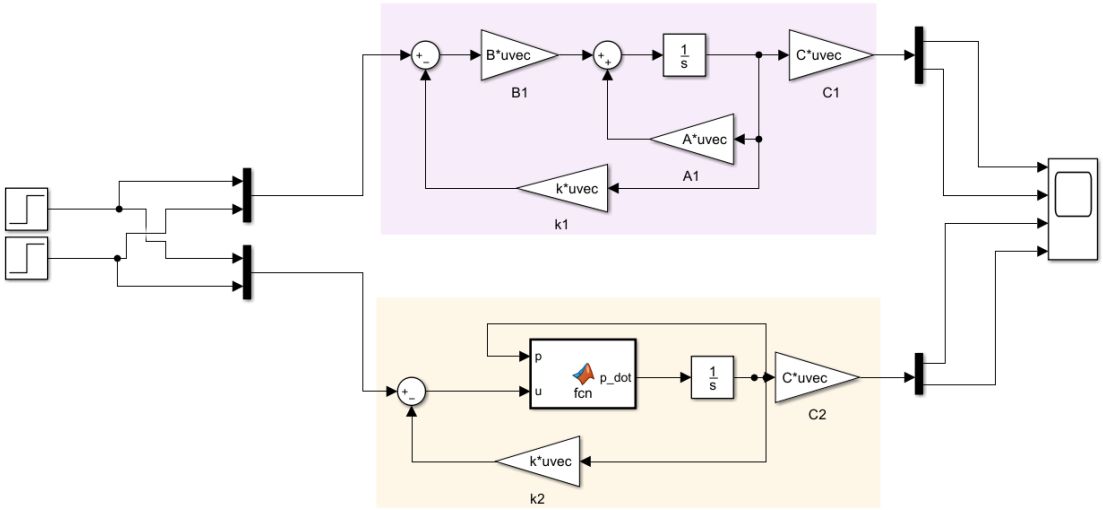
<<<<<<<<

= K2

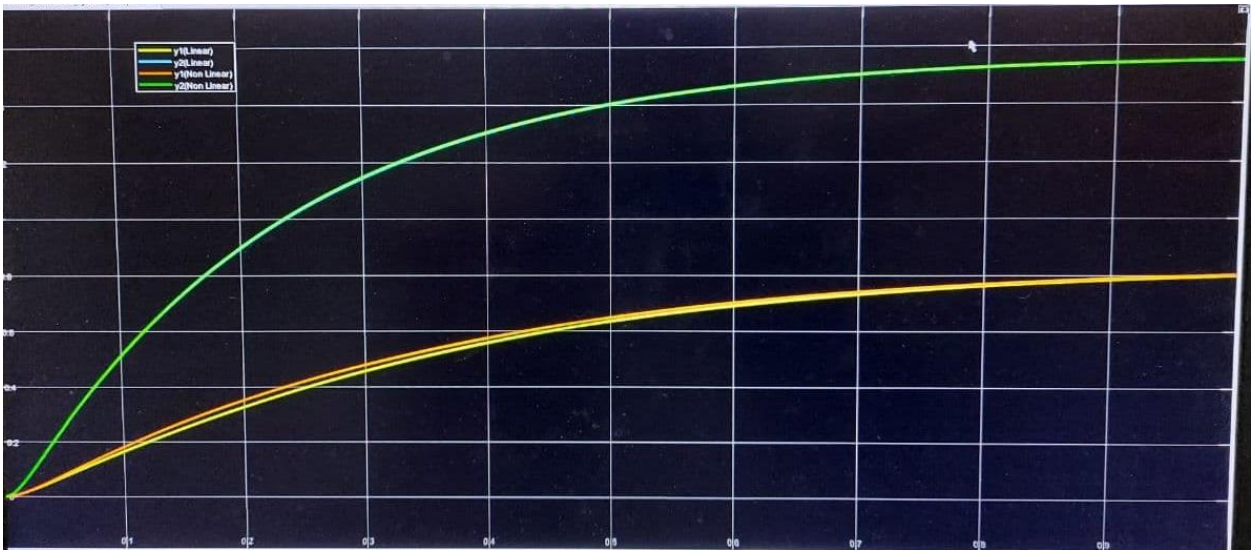
0.5789 1.0526 1.2632

مشاهده می شود که k به دست آمده از هر سه روش یکسان است !

سپس پیاده سازی سیستم کنترلر فیدبک حالت در محیط سیمولینک به شکل زیر صورت گرفت:



3) در نهایت نتیجه شبیه سازی :



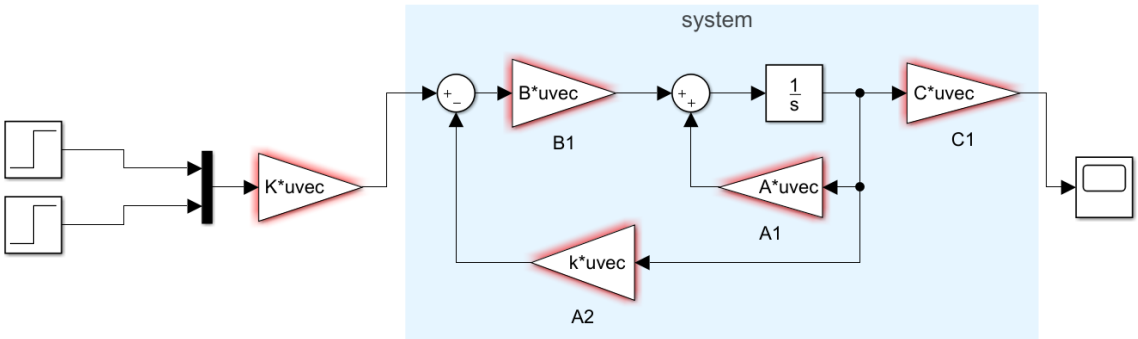
در این پروژه سیستم دارای 2 خروجی می باشد . در تصویر بالا مشاهده می شود که خروجی های مربوط به هر دو حالت سیستم غیر خطی و سیستم خطی شده ، با قرار دادن فیدبک حالت پایدار شده اند و بر یکدیگر منطبق هستند و این انطباق در حالتی صورت گرفته که سیستم بالازدگی برابر صفر دارد . می توان نتیجه گرفت که هم عملیات خطی سازی به درستی

صورت گرفته و هم این که فیدبک حالت طراحی شده توانایی این را دارد که هم سیستم غیر خطی و هم سیستم خطی شده را پایدار بدون بالازدگکی پایدار کند .

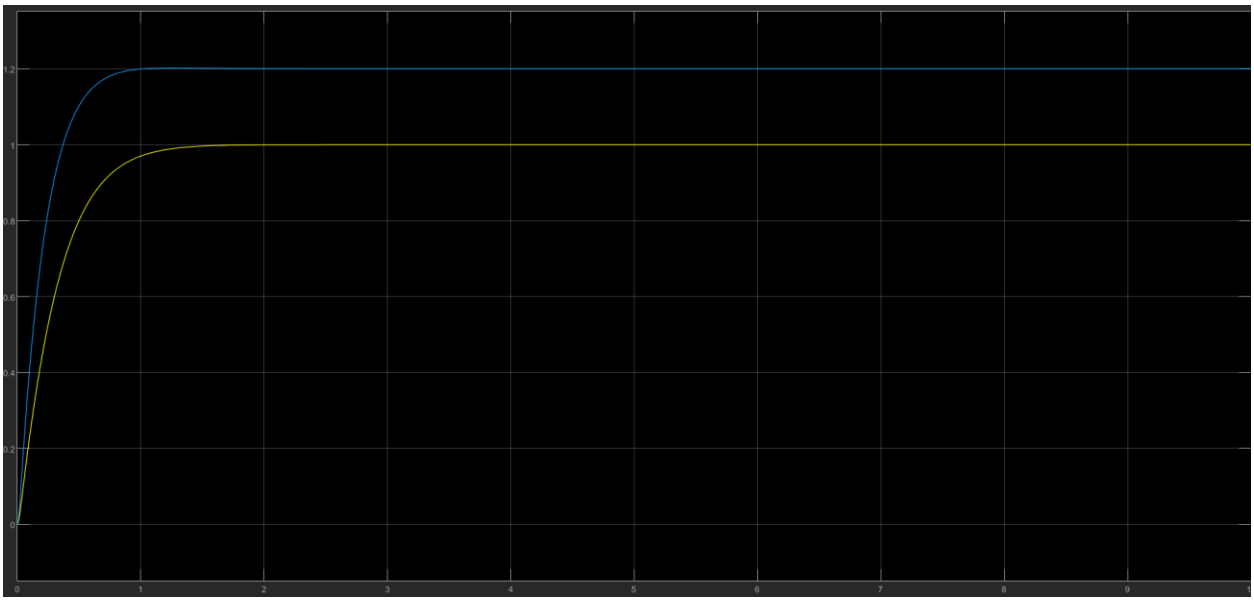
4) سیستم طراحی شده با فیدبک حالت در حالت غیر خطی بودن کاملا پایدار است .

5) پیش جبران ساز استاتیکی برای سیستم طراحی شد و سیستم در حالت های مختلف مورد بررسی قرار گرفت :

الف- پیش جبرانساز استاتیکی بدون اغتشاش

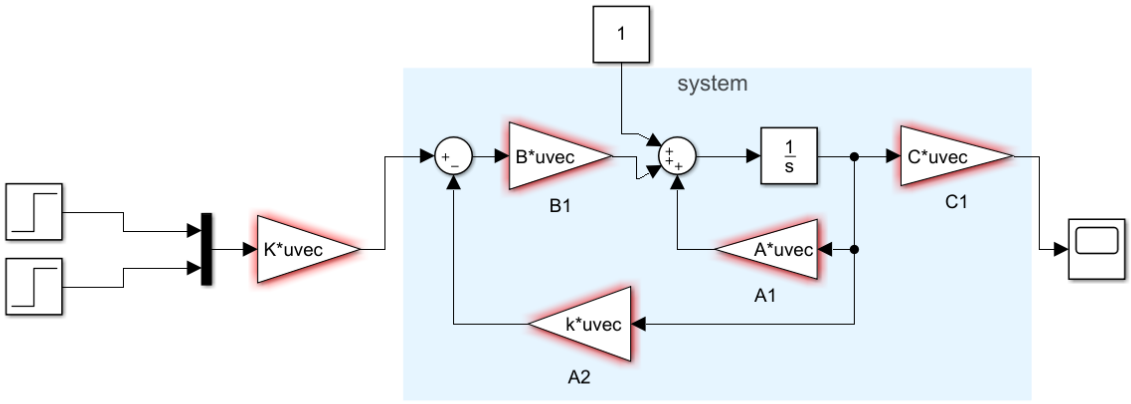


نتیجه شبیه سازی :

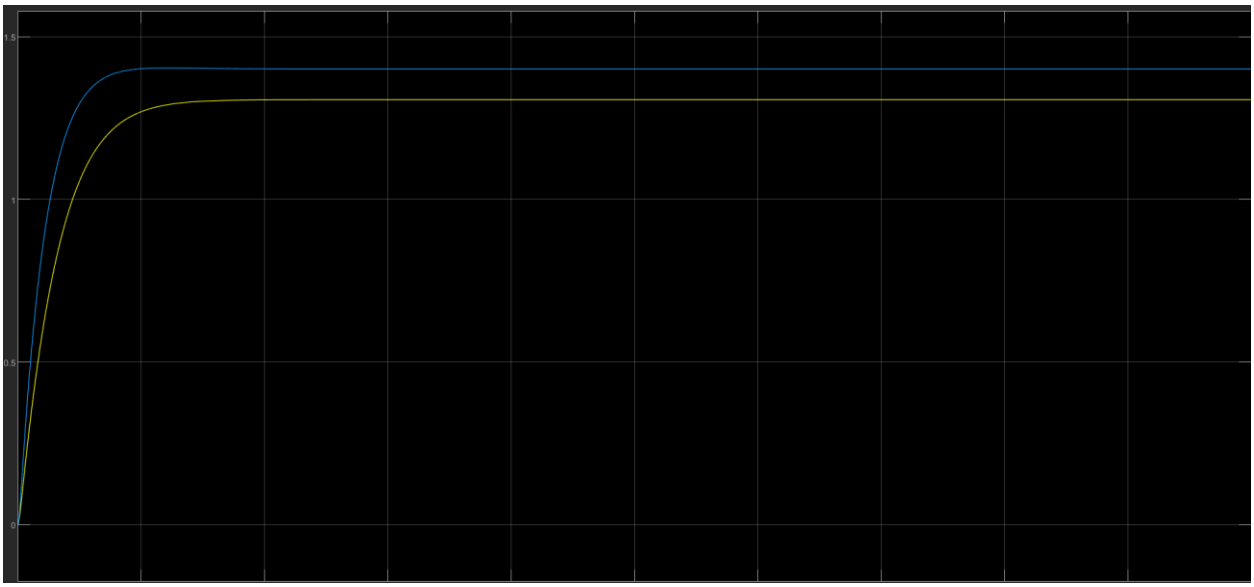


مشاهده میگردد که پیش جبران ساز طراحی شده در نبود اغتشاش به خوبی خطای حالت دائم را صفر کرده است.

ب-- پیش جبرانساز استاتیکی در حضور اغتشاش

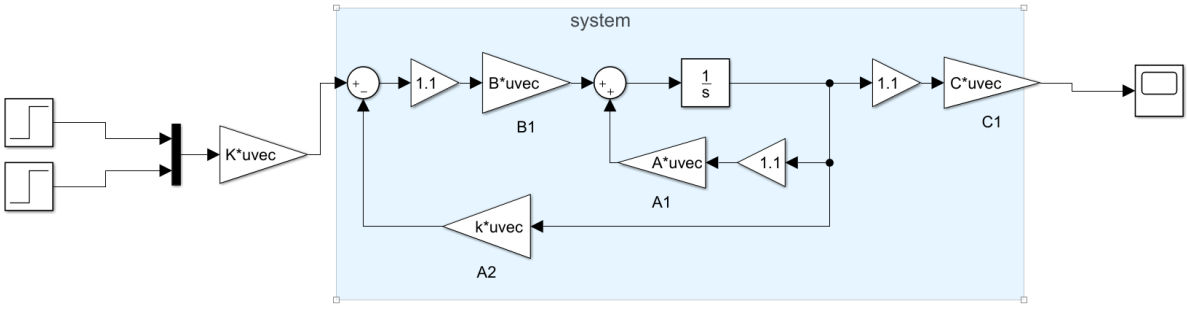


نتایج شبیه سازی :



مشاهده می شود که این پیش جبران ساز در حضور اغتشاش عملکرد مد نظر را نداشته است و تنها خطایحالت دائم یکی از خروجی ها را توانسته صفر کند . این عملکرد در برابر اغتشاش با توجه به ماهیت پیش جبرانساز استاتیکی کاملا قابل پیش بینی بود .

پ- پیش جبرانساز استاتیکی با aging

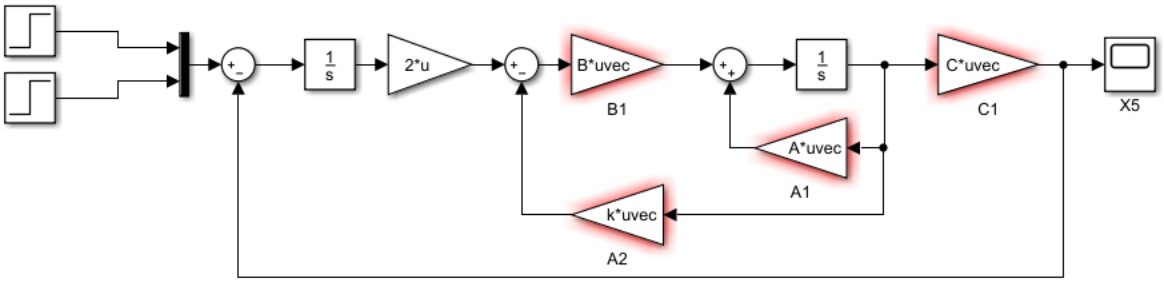


نتایج شبیه سازی :

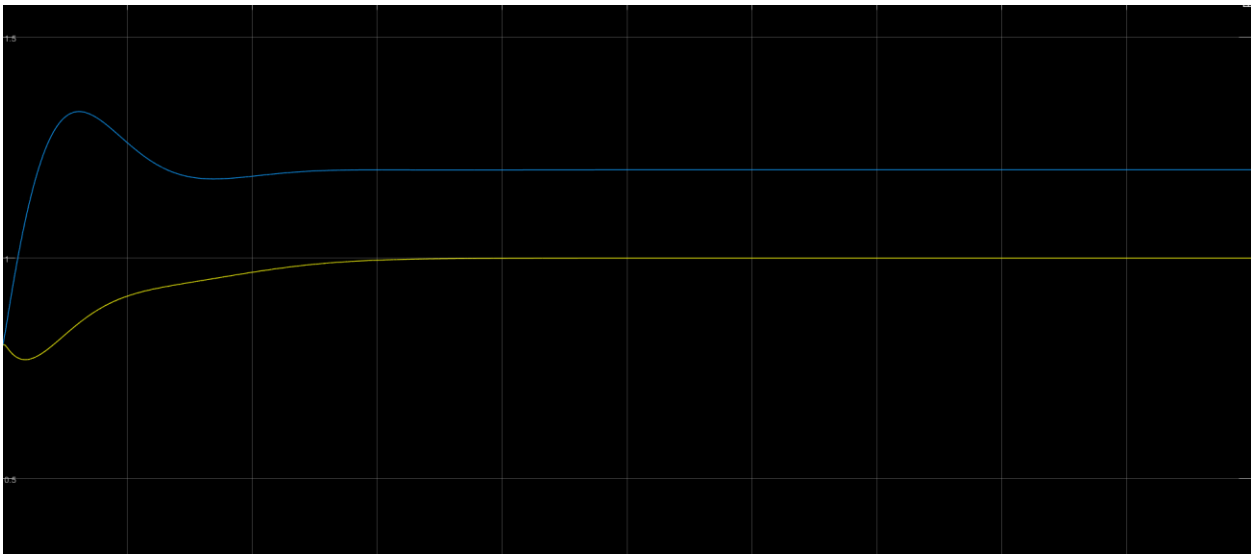
مشاهده میگردد که پیش جبرانساز استاتیکی در مقابل aging هم عملکرد مناسبی ندارد .

در مرحله بعدی برای سیستم پیش جبرانساز دینامیکی طراحی شد و وضعیت سیستم در حالات مختلف شبیه سازی شد :

الف- پیش جبرانساز دینامیکی بدون اغتشاش

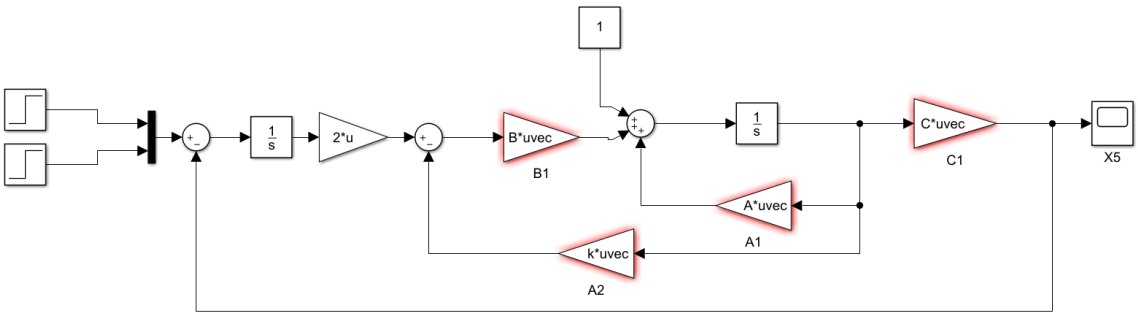


نتایج شبیه سازی :

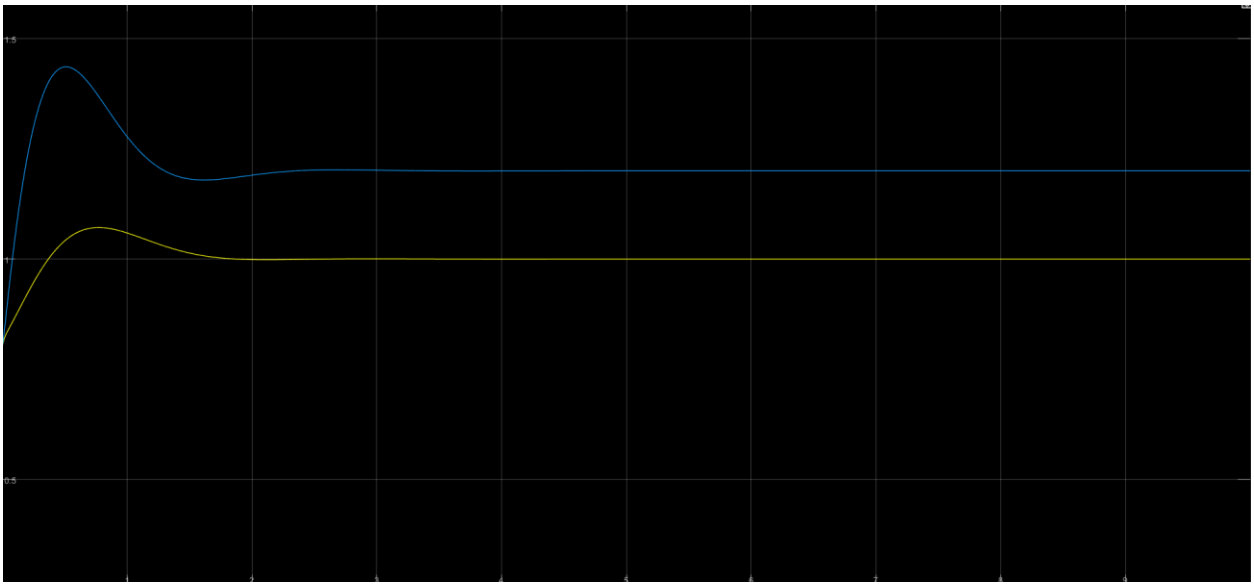


مشاهده میگردد که سیستم با وجود بالازدگی در حد قابل قبول ، خطای حالت دائم برای هر دو خروجی را صفر کرده است.

ب-پیش جبرانساز دینامیکی در حضور اغتشاش

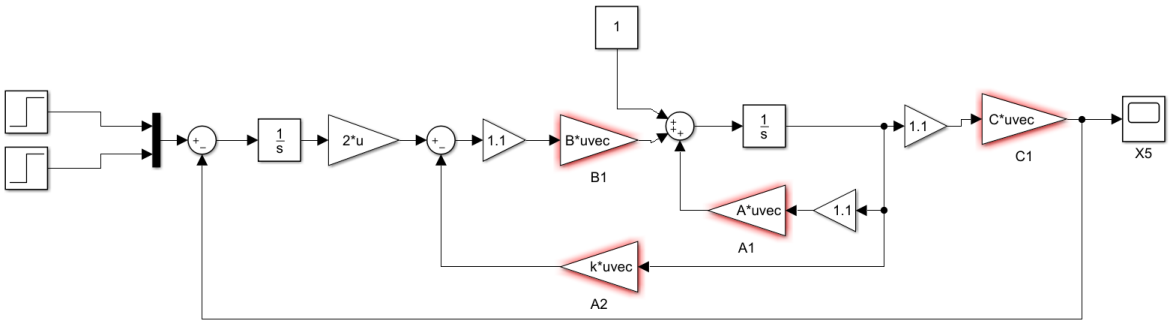


نتایج شبیه سازی:

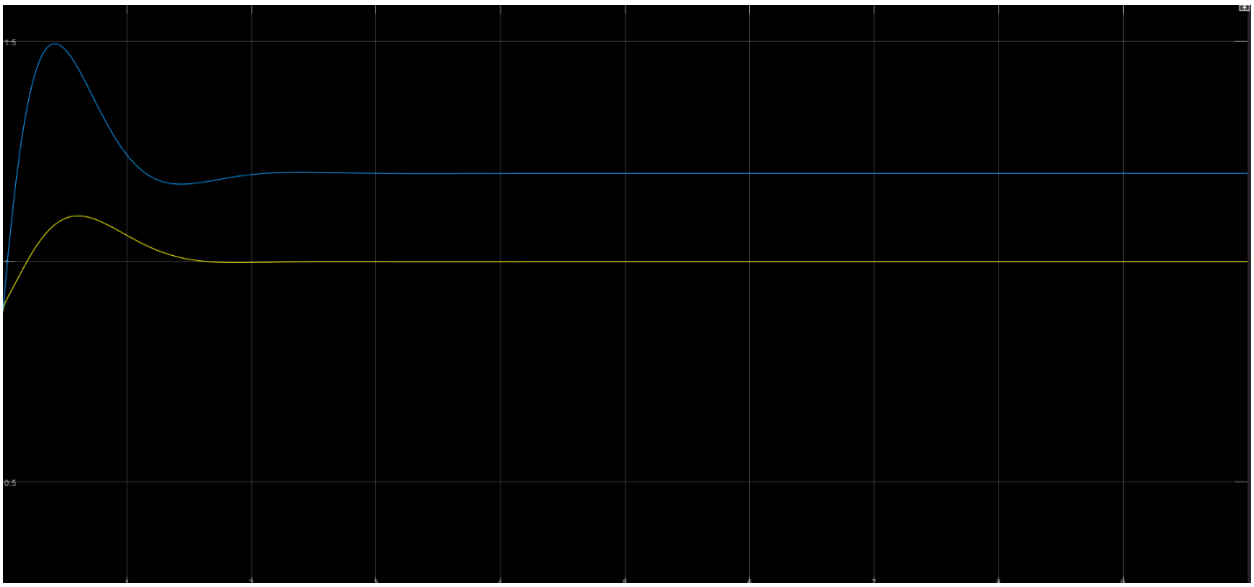


مشاهده میگردد سیستم در حضور پیش جبران ساز دینامیکی ، حتی در حضور اغتشاش هم خطای حالت دائم برای هر دو وردی را صفر کرده است.

پ- پیش جبرانساز دینامیکی با aging



نتایج شبیه سازی :



مشاهده میشود با وجود این که مقداری بالازدگی افزایش پیدا کرده است ، اما بازهم پیش جبرانساز دینامیکی طراحی شده با پاسخ مطلوب برای هر دو ورودی همراه بوده است.

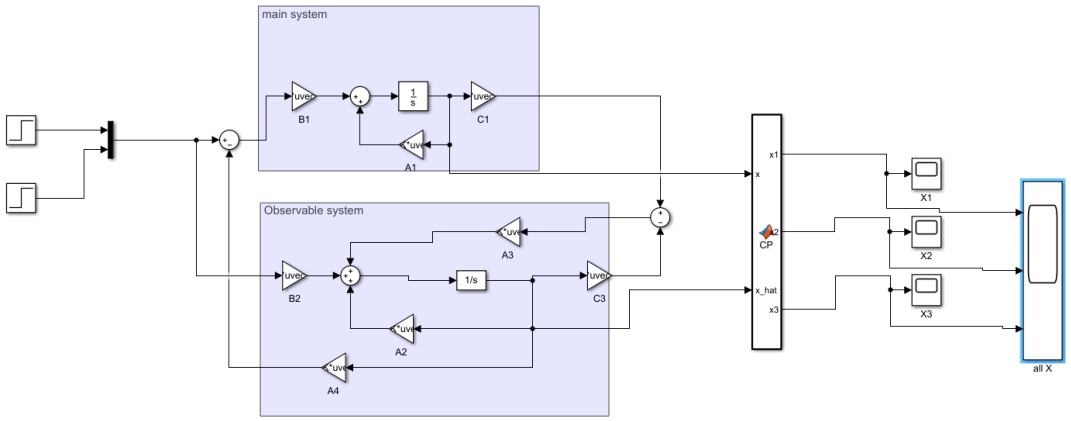
بخش نهایی پروژه درس کنترل مدرن – کنترل موقعیت ربات از راه دور

1) الف) در این قسمت برای سیستم مورد نظر رویتگر مرتبه کامل طراحی می کنیم.

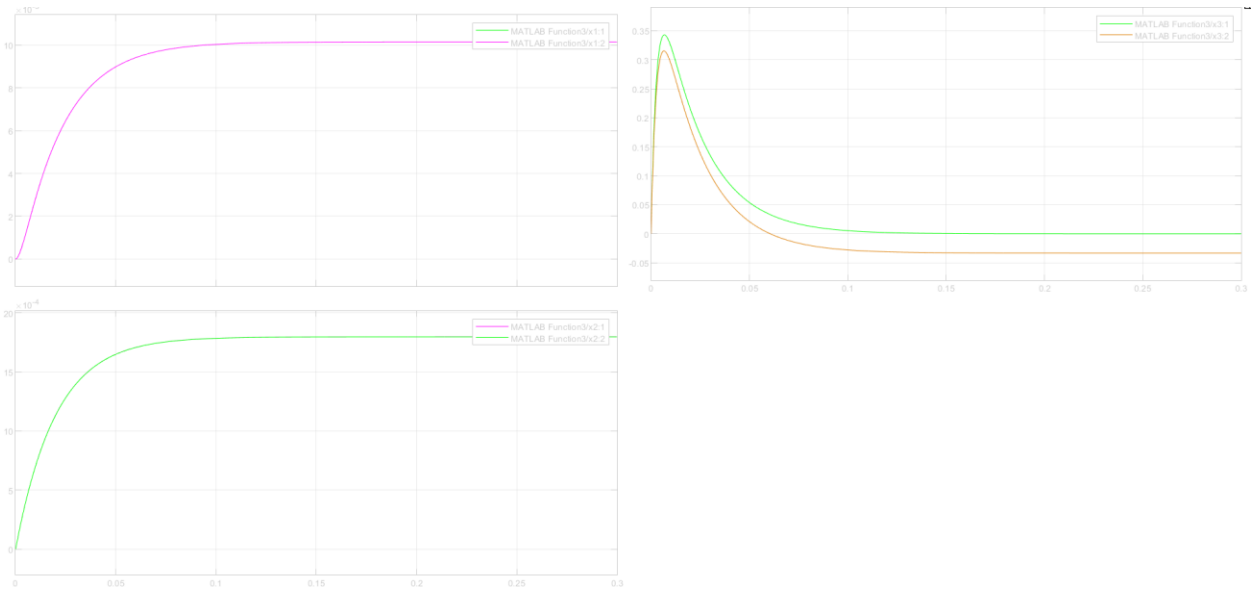
با توجه به این که سیستم چند ورودی چند خروجی است ، با مشورت تدریس یار زیر سسیتم از سیستم مورد نظر را در نظر گرفتیم.

با توجه به این که سیستم مینیمال است و در نتیجه کاهش ناپذیر می باشد ، لذا روند طراحی رویتگر dual طراحی فیدبک حالت خواهد بود.

بلوک زیر برای رویتگر در محیط سیمولینک طراحی شد:



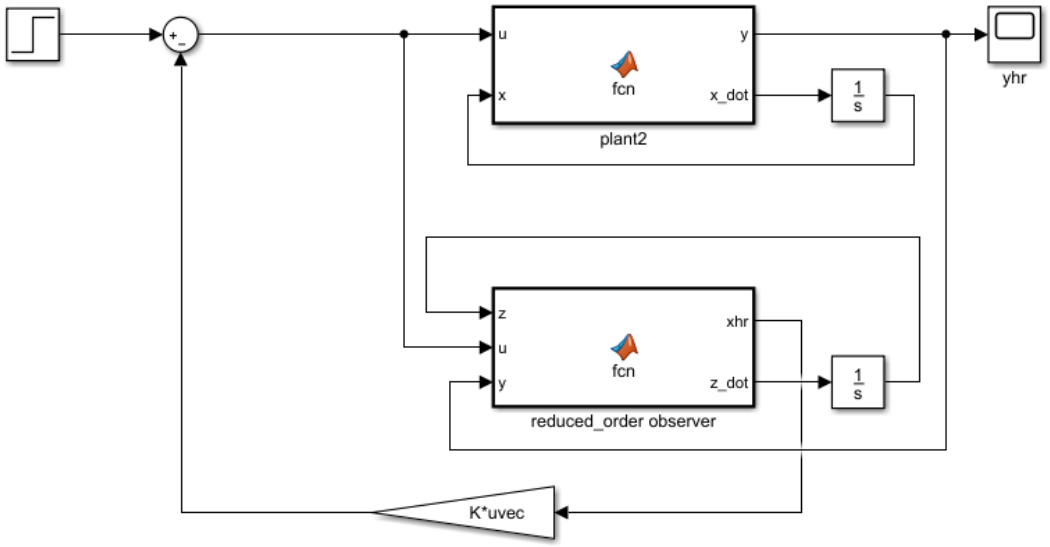
و خروجی سیستم به صورت زیر شد :



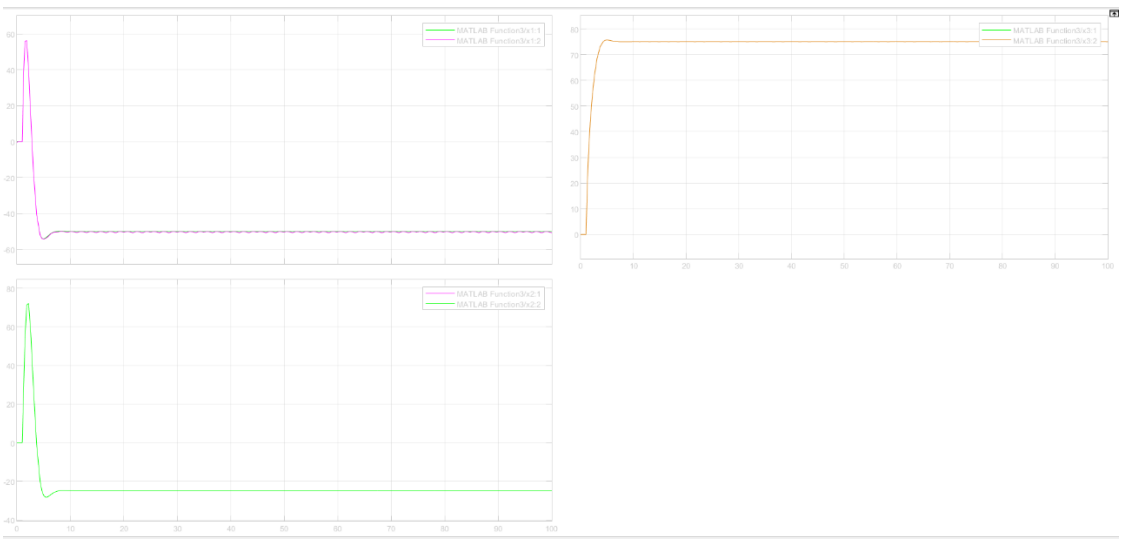
مشاهده می کنیم که رویتگر برا ی دوتا از حالت ها رویتگر به درستی توانسته حالت سیستم را پیش بینی کند اما برای دوتا از حالت ها مقداری خطای حالت دائم داریم که طبیعی به نظر می رسد.

ب) در این قسمت برای سیستم مورد نظر رویتگر مرتبه کاهش یافته طراحی می کنیم.

بلوک زیر برای رویتگر در محیط سیمولینک طراحی شد:



و خروجی سیستم به صورت زیر شد :



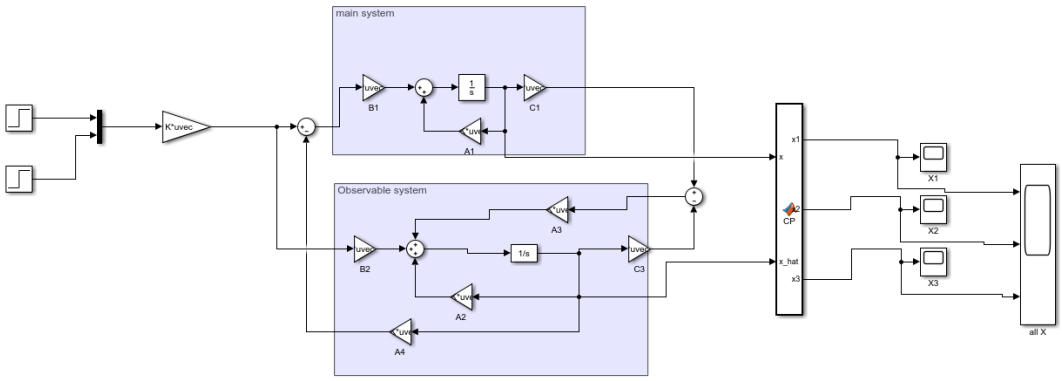
مشاهده می شود که برای دوتا از حالت ها تخمین به درستی صورت گرفته و حالت سوم هم دارای اندکی نوسان است.

2) طراحی پیش جبرانساز برای سیتم با رویتگر

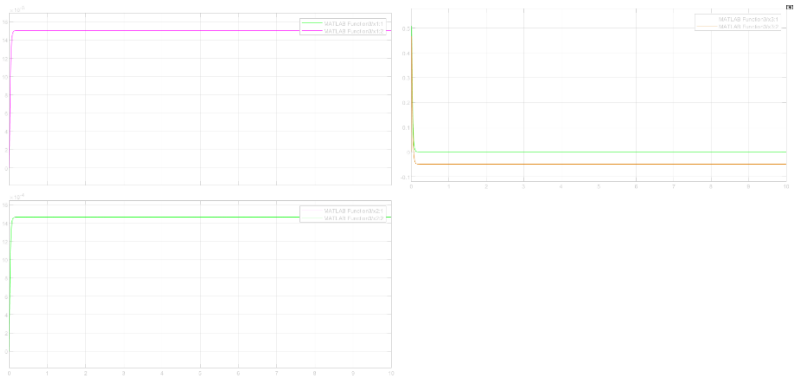
الف) طراحی پیش جبرانساز استاتیکی برای سیتم با رویتگر مرتبه کامل

اثبات شد که پیش جبرانساز استاتیکی برای سیستم در وضعیتی که فیدبک حالت مستقیما از حالت ها گرفته شده نسبت به حالتی که از رویتگر برای پیش بینی حالت ها استفاده می کنیم ، پیش جبرانساز استاتیکی کاملا مطابق است. بنابراین از همان پیش جبرانساز حالت قبل استفاده می کنیم.

بلوک زیر برای رویتگر در محیط سیمولینک طراحی شد:



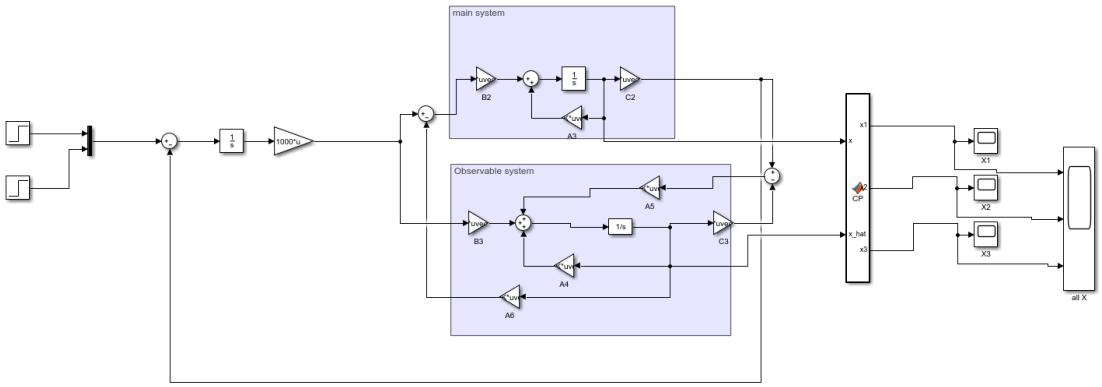
و خروجی سیستم به صورت زیر شد :



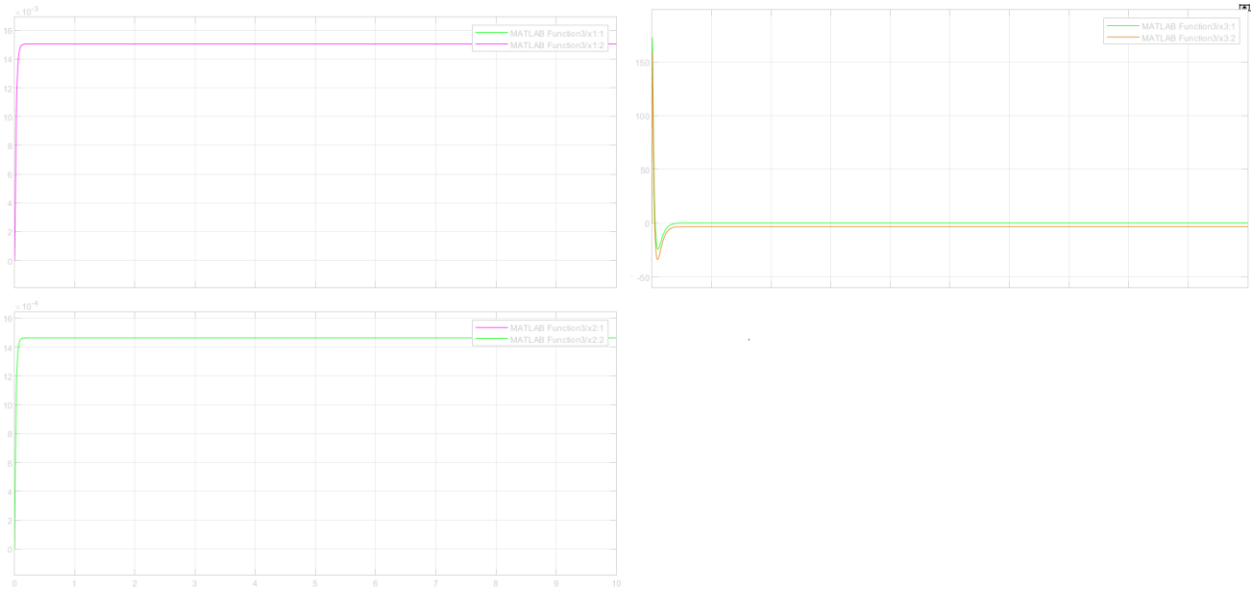
مشاهده می شود نسبت حالت قبل گرچه پاسخ بهبود یافته اما باز هم می توان آنرا بهبود داد.

ب) طراحی پیش جبرانساز دینامیکی برای سیستم با رویتگر مرتبه کامل

بلوک زیر برای رویتگر در محیط سیمولینک طراحی شد:



و خروجی سیستم به صورت زیر شد :



مشاهده می شود که پیش جبرانساز دینامیکی به خوبی توانسته پاسخ را بهبود بخشد.

3) بررسی پایداری لیاپانوف

الف)روش اول

برای بررسی روش اول لیاپانوف ، کد زیر در محیط متلب زده شد.

```
%Lyapanov Stability first method
stability_check=eig(A)
```

مقادیر ویژه به صورت زیر محاسبه شد:

stability_check

4x1 double

	1	2
1	-50.5302	
2	-4.7649e+03	
3	-50.1611	
4	-1.5575e+04	

از آن جایی که تمام مقادیر ویژه منفی هستند ، بنابراین سیتم به روش اول لیاپانوف پایدار است.

ب)روش دوم لیاپانوف

در ابتدا با آزمون و خطا تابع مناسب لیاپانوف محاسبه شد.

$$V_1 = \frac{1}{2} z_1^{\,2}$$

$$V_2 = V_1 + \frac{1}{2} z_2^{\,2}$$

که طبق محاسبات انجام شده ، در نقطه تعادل صدق میکند ، خود تابع مثبت است و دارای مشتق منفی می باشد.بنابراین سیستم بنابه روش دوم لیاپانوف پایدار است.

ج)بررسی پایداری ورودی به خروجی

پس ازمحاسبه تابع تبدیل قطب های سیستم قابل مشاهده است.با توجه به اینکه همه ی قطب ها دارای مقادیر منفی

هستند لذا میتوان گفت سیستم پایدار ورودی –خروجی یا BIBOمیباشد.

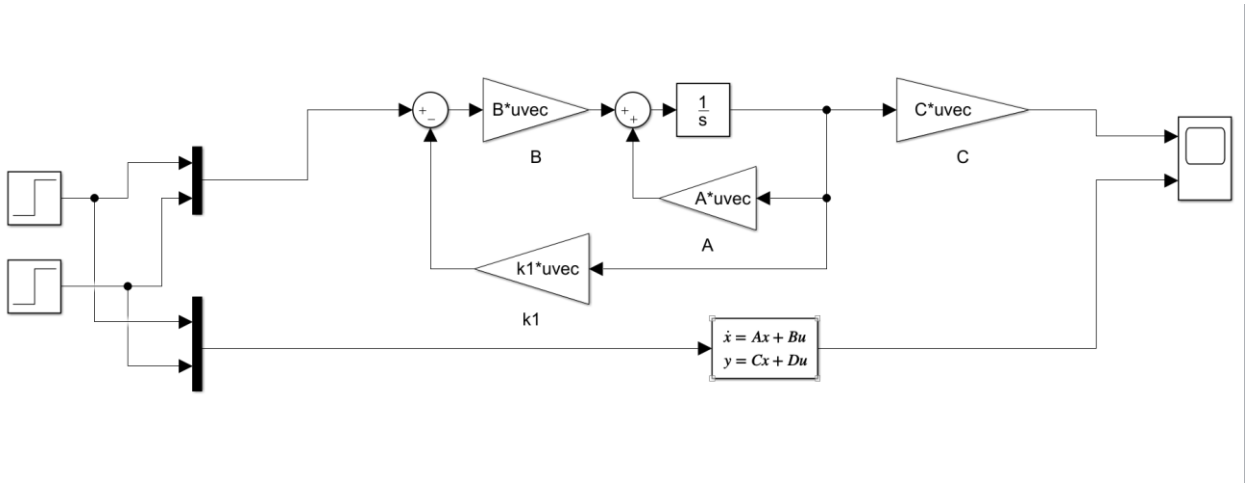
از طرفی چون هیچ مقدارویژه ی صفری نداریم لذا سیستم پایدار مرزی نمی باشد.

4) بهینه سازی

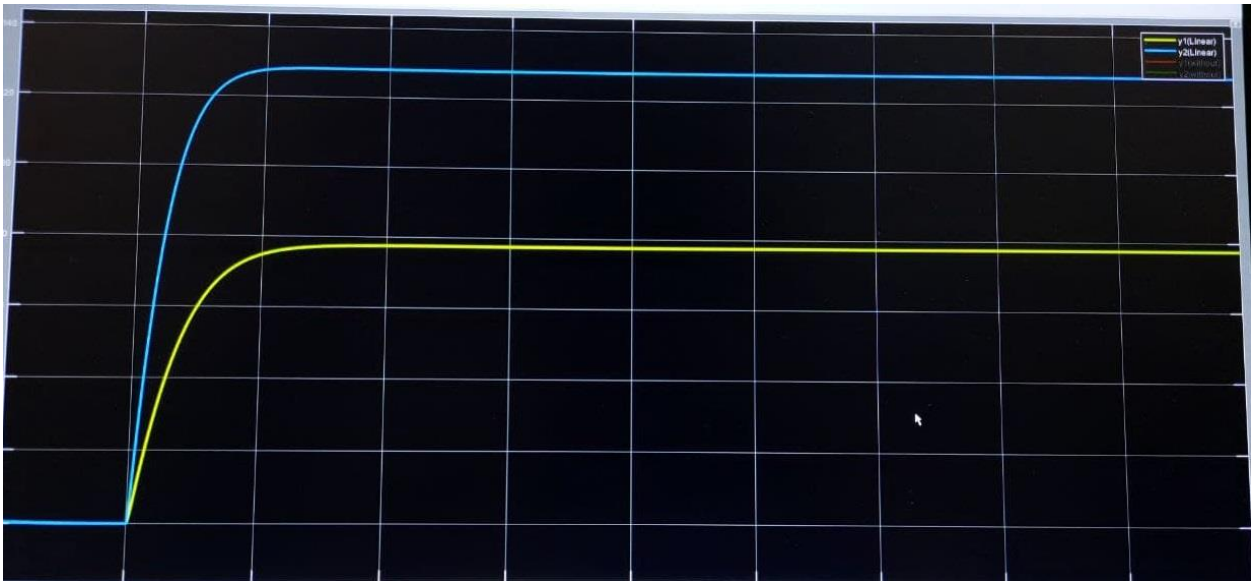
جهت بهینه سازی سیستم کد زیر در متلب زده شد

```
%COST_FUNCTION
syms p11 p12 p13 p14 p21 p22 p23 p24 p31 p32 p33 p34 p41 p42 p43 p44
R = eye(2);
Q = 1000*[1 0 0 0;0 0 0 0;0 0 0 0;0 0 0 0];
P = [p11 p12 p13 p14 ;
    p21 p22 p23 p24;
    p31 p32 p33 p34;
    p41 p42 p43 p44];
E = A'*P + P*A - P*B*inv(R)*B'*P == -Q ;
sP = solve(E , P);
i = 1;
P_ = [sP.p11(i) sP.p12(i) sP.p13(i) sP.p14(i) ;
      sP.p21(i) sP.p22(i) sP.p23(i) sP.p24(i);
      sP.p31(i) sP.p32(i) sP.p33(i) sP.p34(i);
      sP.p41(i) sP.p42(i) sP.p43(i) sP.p44(i)];
P_ = vpa(P_,3);
k1= inv(R)*B'*P_;
k1 = vpa(k,3);
k1 = double(k1)
```

و سپس شبیه سازی زیر در محیط سیمولینک صورت گرفت :



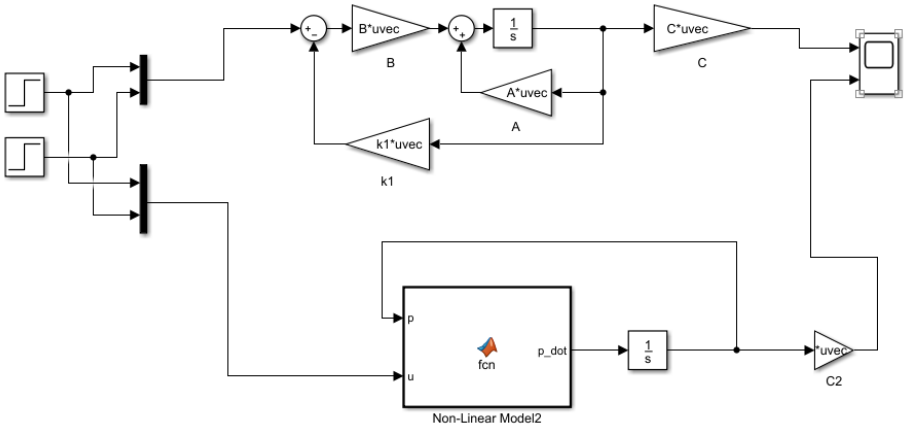
خروجی سیستم به صورت زیر درآمد:



مشاهده می کنیم به خوبی توانسته است دنبال کند.

5) بهینه سازی برای سیستم غیر خطی

شبیه سازی مربوط به بخش غیر خطی انجام شد :



که پاسخ مناسبی برای سیستم غیر خطی به دست نیامد. اما با توجه به این که سیستم خطی پاسخ خوبی به بهینه سازی داد ، در همین حد اکتفا می کنیم.