دانشگاه صنعتی امیرکبیر
Amirkabir University
of Technology

مقطع

کارشناسی

عنوان

Trajectory tracking of a Segway model

درس

سیستمهای کنترل خطی

نگارش

مارال مرداد   ۹۷۲۳۱۴۸

سحرالسادات اسلامی   ۹۷۲۳۰۰۵

مهشاد علیان نصرابادی   ۹۷۲۳۰۵۹

نگار سلطان محمدی   ۹۷۲۳۰۳۸

نام استاد:

جناب دکتر طالبی

بهمن ۱۳۹۹

چکیده:

در این پروژه با توجه به معادلات مکانیکی، ولتاژ به عنوان ورودی در نظر گرفته شده و دو متغیر سرعت خطی و زاویه پاندول با محور عمودی به عنوان خروجی در نظر گرفته شده‌اند.

حالت مطلوب، دنبال کردن ورودی توسط سیستم می‌باشد. برای مثال با دادن ورودی پله انتظار داریم زاویه و سرعت ما ورودی را دنبال کند.

با توجه به وضعیت مکان هندسی توابع تبدیل و وجود یک شاخه در سمت راست برای تمامی kهای صفر تا بی نهایت، چالش اصلی در این پروژه پایدار کردن توابع تبدیل می باشد.

برای کنترل زاویه کنترلر PID و برای کنترل سرعت state feedback بکار گرفته شده است.

فهرست مطالب

۱– مقدمه

ساختن وسایل به منظور جابه‌جایی انسان‌ها از زمانهای قدیم وجود داشته است. اسکوتر، دوچرخه و در سال‌های اخیر هم Segway (سگوی) که در واقع می‌توان از آن به عنوان پاندول معکوس هم نام برد از مسائل کلاسیک غیرخطی در سیستم‌های کنترل هستند.

این دستگاه وسیله‌ای است برای جابه‌جایی یک فرد در فواصل نسبتاً کوتاه که با حرکت شخص به جلو و عقب کنترل می‌شود.

ایده‌ی ساخت این سیستم در سال ۱۹۹۴ شکل گرفت و ۱۹۹۷ اولین نمونه از آن ساخته شد.

این دستگاه‌ها برای کاربردهای مختلفی از جمله گردشگری، توریستی و در جامعه پزشکی مورد استفاده قرار می‌گیرند.

شکل ۱-۱ : نمونه امروزی از Segway

۲ – سیستم مورد مطالعه

سگوی را می‌توان با یک پاندول معکوس ساده مدل کرد که به یک ارابه متصل شده است و توسط دو موتور که در دو چرخ آن هستند در راستای محور افقی حرکت می‌کند یعنی حرکت رو به جلو و رو به عقب دارد.

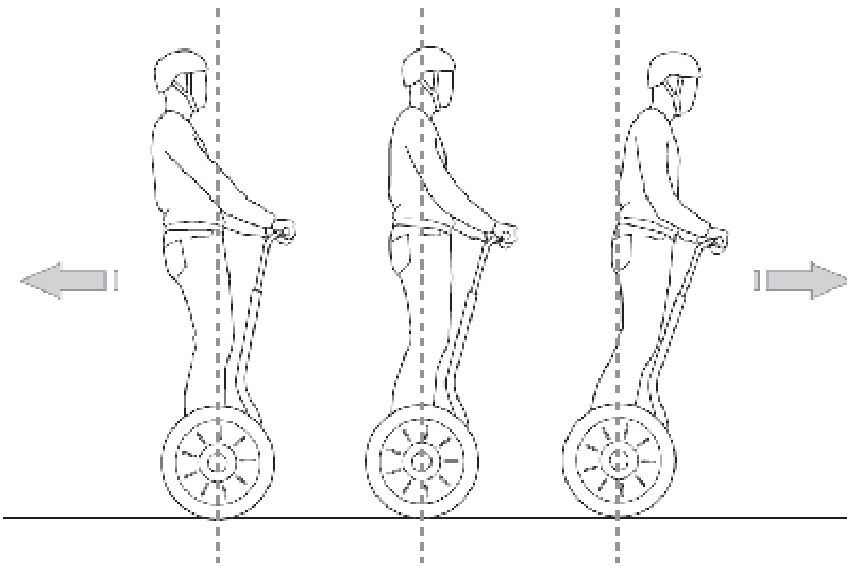
شخص روی سگوی مانند پاندول معکوس عمل می‌کند. سنسورهایی جهت اندازه‌گیری زاویه انحراف پاندول و سرعت و موقعیت سگوی در سیستم قرار دارد.

مدل خطی سگوی، ولتاژ وارد شده به چرخ‌ها را به عنوان ورودی می‌گیرد. ما باید کنترلری طراحی کنیم که ولتاژ ورودی را تنظیم کند.

۳– تعیین ورودی و خروجی‌های سیستم

این سیستم در واقع یک سیستم یک ورودی- دو خروجی می‌باشد و ما باید بتوانیم با یک سیگنال کنترلی که به موتور می‌دهیم، به طور همزمان سرعت خطی سیستم و زاویه پاندول را کنترل کنیم.

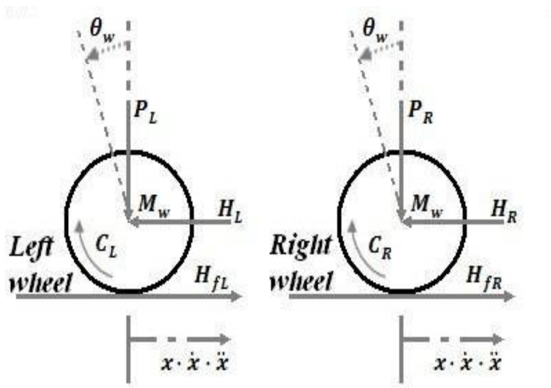
هدف اصلی ما در این پروژه این است که فرد بتواند تعادل خود را زمانی که روی دستگاه ایستاده حفظ کند یعنی وضعیت پاندول از یک حالت نامتعادل به نقطه تعادل برسد و بتواند این وضعیت را حفظ کند.



شکل۱-۳: نحوه کنترل و جهت‌دهی به segway

۴– سیستم غیر خطی

ما مدل کردن سیستم را با بررسی کردن رفتار دینامیکی سیستم آغاز می‌کنیم. ابتدا گشتاور ورودی از موتور به سیستم را بررسی می‌کنیم. فرمول‌ها در [۳] وجود دارد و برای تکرار و بیان راحت‌تر در ادامه آورده شده است.



شکل ۱-۴ : گشتاور ورودی به چرخ‌ها [۳]

چون نیروهای وارد شده به چرخ‌ها یکسان است در به دست آوردن معادلات تنها چرخ راست را بررسی می‌کنیم.

با استفاده از قانون نیوتن داریم:

$$\sum F_x = Ma \Rightarrow M_w \ddot{x} = H_{fR} - H_R \qquad (۱)$$

جمع گشتاور در یک چرخ:

$$\sum M_o = I\alpha \Rightarrow I_w \ddot{\theta}_w = C_R - H_{fR} r \qquad (۲)$$

گشتاور موتور:

$$\tau_m = I_R \frac{d\omega}{dt} + \tau_a \qquad (۳)$$

با استفاده از پارامترهای موتور دی‌سی:

$$C = I_R \frac{d\omega}{dt} = -\frac{k_m k_e}{R}\dot{\theta}_\omega + \frac{k_m}{R} V_a \qquad (۴)$$

معادله (۲) را می‌توان به شکل زیر نوشت:

$$I_w \ddot{\theta}_w = -\frac{\kappa_m \kappa_e}{R}\dot{\theta}_\omega + \frac{\kappa_m}{R} V_a - H_{fR} r \qquad (۵)$$

(۶)

از روابط (۱) و (۶) معادلات چرخ‌ها به دست می‌آید:

$$H_{fR} = -\frac{k_m k_e}{Rr}\dot{\theta}_{\omega} + \frac{k_m}{Rr}V_a - \frac{I_w}{r}\ddot{\theta}_w$$

چرخ راست:

$$M_w \ddot{x} = -\frac{k_m k_e}{Rr}\dot{\theta}_{\omega} + \frac{k_m}{Rr}V_a - \frac{I_w}{r}\ddot{\theta}_w - H_R \qquad (۷)$$

چرخ چپ:

$$M_w \ddot{x} = -\frac{k_m k_e}{Rr}\dot{\theta}_{\omega} + \frac{k_m}{Rr}V_a - \frac{I_w}{r}\ddot{\theta}_w - H_L \qquad (۸)$$

۵– خطی‌سازی معادلات سیستم

معادله حرکت زاویه‌ای را به یک معادله خطی تبدیل می‌کنیم.

$$\ddot{\theta}_w r = \ddot{x} \Rightarrow \ddot{\theta}_w = \frac{\ddot{x}}{r} \qquad (۹)$$

$$\dot{\theta}_w r = \dot{x} \Rightarrow \dot{\theta}_w = \frac{\dot{x}}{r} \qquad (۱۰)$$

در معادلات (۷) و (۸) جاگذاری می‌کنیم:

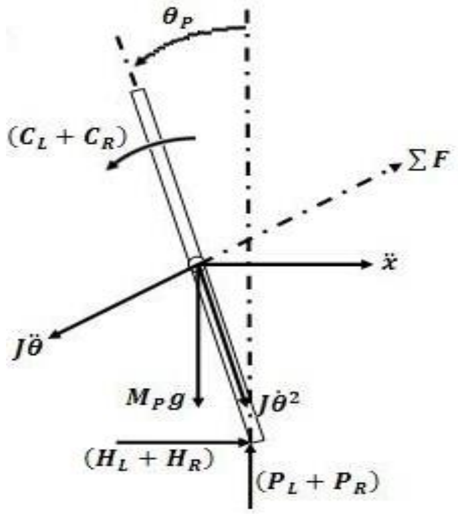
$$M_w \ddot{x} = -\frac{k_m k_e}{Rr^2}\dot{x} + \frac{k_m}{Rr}V_a - \frac{I_w}{r^2}\ddot{x} - H_R \qquad (۱۱)$$

$$M_w \ddot{x} = -\frac{k_m k_e}{Rr^2}\dot{x} + \frac{k_m}{Rr}V_a - \frac{I_w}{r^2}\ddot{x} - H_L \qquad (۱۲)$$

از (۱۱) و (۱۲) داریم:

$$2\left(M_w + \frac{I_w}{r^2}\right)\ddot{x} = -\frac{2k_m k_e}{Rr^2}\dot{x} + \frac{2k_m}{Rr}V_a - (H_L + H_R) \qquad (۱۳)$$

فرد ایستاده روی Segway و دسته‌ی آن را به عنوان یک جسم صلب در نظر می‌گیریم و نیروهای وارد به آنها را بررسی می‌کنیم.



شکل ۱-۵: نیروهای وارد شده به شخص [۳]

قانون نیوتن را روی محور افقی اعمال می‌کنیم:

$$\sum F_x = M_p \ddot{x}$$
$$\Rightarrow (H_L + H_R) - M_p l \ddot{\theta}_p \cos\theta_p + M_p l \dot{\theta}_p^2 \sin\theta_p = M_p \ddot{x} \qquad (۱۴)$$

$$(H_L + H_R) = M_p l \ddot{\theta}_p \cos\theta_p + M_p l \dot{\theta}_p^2 \sin\theta_p + M_p \ddot{x} \qquad (۱۵)$$

مجموع تمام نیروهای وارد شده:

$$\sum F_{xp} = M_p \ddot{x} \cos\theta_p \qquad (۱۶)$$
$$\Rightarrow (H_L + H_R)\cos\theta_p + (P_L + P_R)\sin\theta_p - M_p l \ddot{\theta}_p - M_p g \sin\theta_p = M_p \ddot{x} \cos\theta_p$$

مجموع تمام گشتاورها:

$$\sum M_o = I\alpha$$
$$\Rightarrow -(H_L + H_R) l \cos\theta_p + (P_L + P_R) l \sin\theta_p + (C_L + C_R) = I_p \ddot{\theta}_p \qquad (۱۷)$$

با استفاده از روابط (۱۳) و (۱۱) و (۹)، گشتاور ورودی به سیستم بر اساس پارامترهای موتور را به دست می‌آوریم:

$$(C_L + C_R) = -\frac{2k_m k_e}{Rr}\dot{x} + \frac{2k_m}{R} V_a \qquad (۱۸)$$

(۱۸) را در (۱۷) جاگذاری می‌کنیم:

$$-(H_L+H_R)l\cos\theta_p+(P_L+P_R)l\sin\theta_p+\left(-\frac{2k_mk_e}{Rr}\dot{x}+\frac{2k_m}{R}V_a\right)=I_p\ddot{\theta}_p \qquad (۱۹)$$

از (۱۶) و (۱۹) داریم:

$$I_p\ddot{\theta}_p-\frac{2k_mk_e}{Rr}\dot{x}+\frac{2k_m}{R}V_a+M_pl^2\ddot{\theta}_p+M_pgl\sin\theta_p=-M_pl\ddot{x}\cos\theta_p \qquad (۲۰)$$

(۱۳) را در (۱۵) جاگذاری می‌کنیم:

$$2\left(M_w+\frac{I_w}{r^2}\right)\ddot{x}=-\frac{2k_mk_e}{Rr^2}\dot{x}+\frac{2k_m}{Rr}V_a-M_pl\ddot{\theta}_p\cos\theta_p+M_pl\dot{\theta}_p^2\sin\theta_p-M_p\ddot{x} \qquad (۲۱)$$

معادلات حرکت غیرخطی سیستم که از روابط (۲۱) و (۲۲) به دست می‌آیند:

$$(M_pl^2+I_p)\ddot{\theta}_p-\frac{2k_mk_e}{Rr}\dot{x}+\frac{2k_m}{R}V_a+M_pgl\sin\theta_p=-M_pl\ddot{x}\cos\theta_p \qquad (۲۲)$$

$$\frac{2k_m}{Rr}V_a=\left(2M_w+\frac{2I_w}{r^2}+M_p\right)\ddot{x}+\frac{2k_mk_e}{Rr^2}\dot{x}+M_pl\ddot{\theta}_p\cos\theta_p-M_pl\dot{\theta}_p^2\sin\theta_p \qquad (۲۳)$$

۶– ساده سازی معادلات و ماتریس های حالت

برای به دست آوردن مدل فضای حالت، باید معادلات بالا را خطی سازی کنیم. فرض می‌کنیم Φ خیلی کوچک باشد.

$$\theta_P=\Phi+\pi$$

$$\cos\theta_p=-1,\quad \sin\theta_p=-\phi,\quad \left(\frac{d\theta_p}{dt}\right)^2=0 \qquad (۲۴)$$

معادلات حالت سیستم به دست می‌آید:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{2k_m k_e \left(M_p lr - I_p - M_p l^2\right)}{Rr^2\alpha} & \frac{M_p^2 g l^2}{\alpha} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{2k_m k_e \left(r\beta - M_p l\right)}{Rr^2\alpha} & \frac{M_p g l \beta}{\alpha} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{2k_m \left(-M_p lr + I_p + M_p l^2\right)}{Rr\alpha} \\ 0 \\ \frac{2k_m \left(-r\beta + M_p l\right)}{Rr\alpha} \end{bmatrix} V_a$$

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad (۲۵)$$

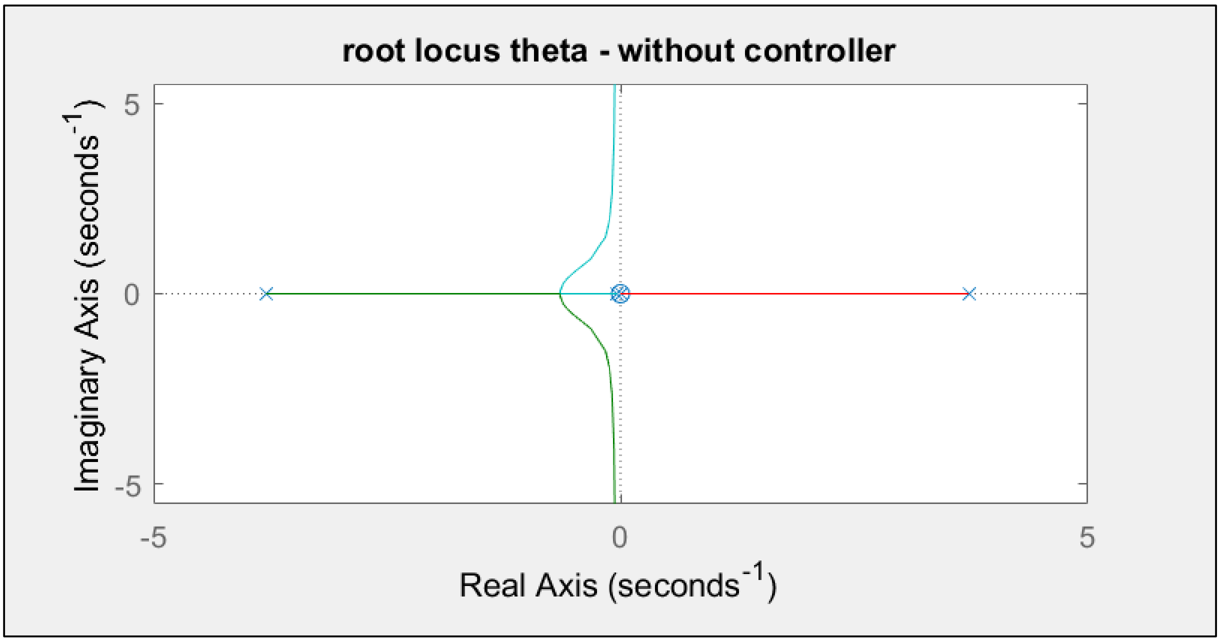
$$\beta = \left(2M_w + \frac{2I_w}{r^2} + M_p\right) \qquad (۲۶)$$

$$\alpha = \left[I_p \beta + 2M_p l^2 \left(M_w + \frac{I_w}{r^2}\right)\right]$$

۷- طراحی جبران ساز

کنترلر زاویه:

در ابتدا برای کنترل زاویه مکان هندسی تابع تبدیل زاویه رسم شده است:



شکل ۷-۱ : مکان هندسی تابع تبدیل زاویه

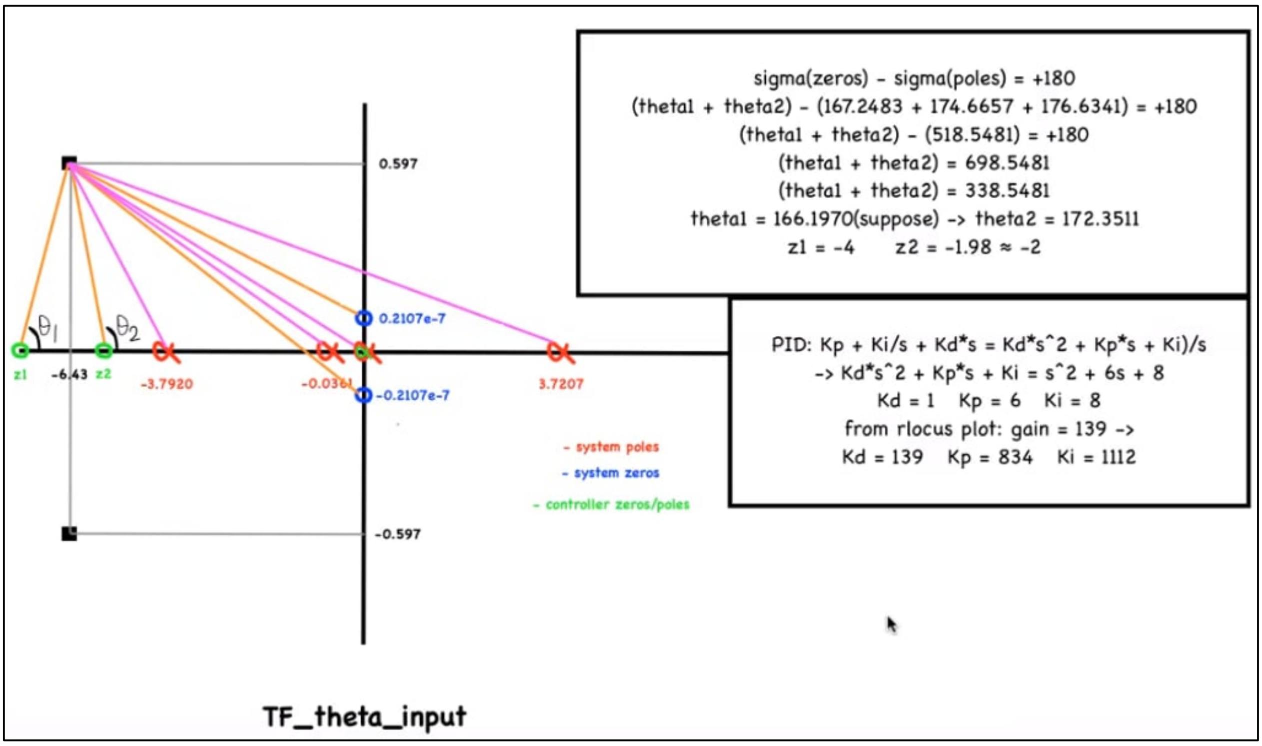
همانطور که مشاهده می‌شود یک شاخه‌ی سمت راست به ازای تمامی مقادیر k بین ۰ تا بی نهایت دیده می‌شود که این به معنای ناپایداری است. هدف ما در کنترل زاویه رسیدن به overshoot و settling time مطلوب می‌باشد.

برای این کار overshoot=۱۰% و settling time=۱s در نظر گرفته شده است و از روی این overshoot و settling time با تقریب این سیستم با سیستم مرتبه دو استاندارد، قطب های مطلوب بدست می‌آید.

(۶,۴۳-۰,۵۹۷j , -۶,۴۳+۰,۵۹۷j-)

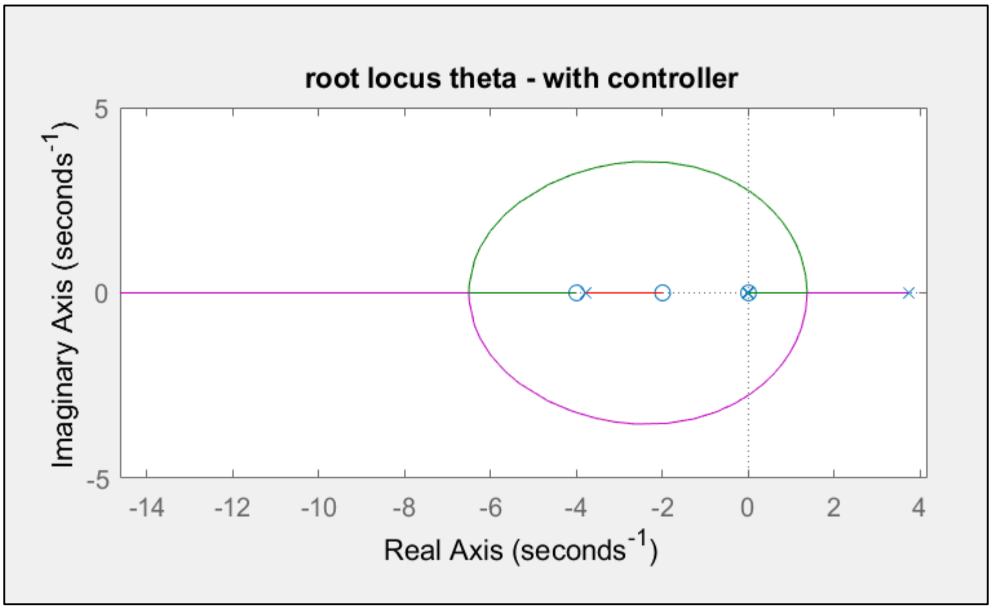
حال می‌خواهیم برای سیستم کنترلر PID ای طراحی کنیم که در نهایت قطب‌های غالب سیستم حلقه بسته‌ی ما قطب های مطلوب شوند. همانطور که می‌دانیم PID شامل یک قطب در مبداء و دو صفر می‌باشد و مکان این صفرها را با اعمال شرط زاویه مانند شکل پیدا می‌کنیم. همانطور که در شکل مشخص است دو

مجهول تتا۱ و تتا۲ در این معادله داریم که با توجه به اینکه صفرها در root locus مکان را به سمت خود جذب می‌کنند، یک ۰ را در نقطه ی ۴- قرار می‌دهیم. با اعمال شرط زاویه مکان صفر دیگر در نقطه‌ی ۱,۹۸- می‌باشد که آن را با ۲- تقریب می‌زنیم. با توجه به تابع تبدیل کنترلر PID بدست آمده ضرایب Kp,Ki,Kd را همانند شکل بدست می‌آوریم.



شکل ۷-۲ : به دست آوردن مکان صفرها به صورت تئوری

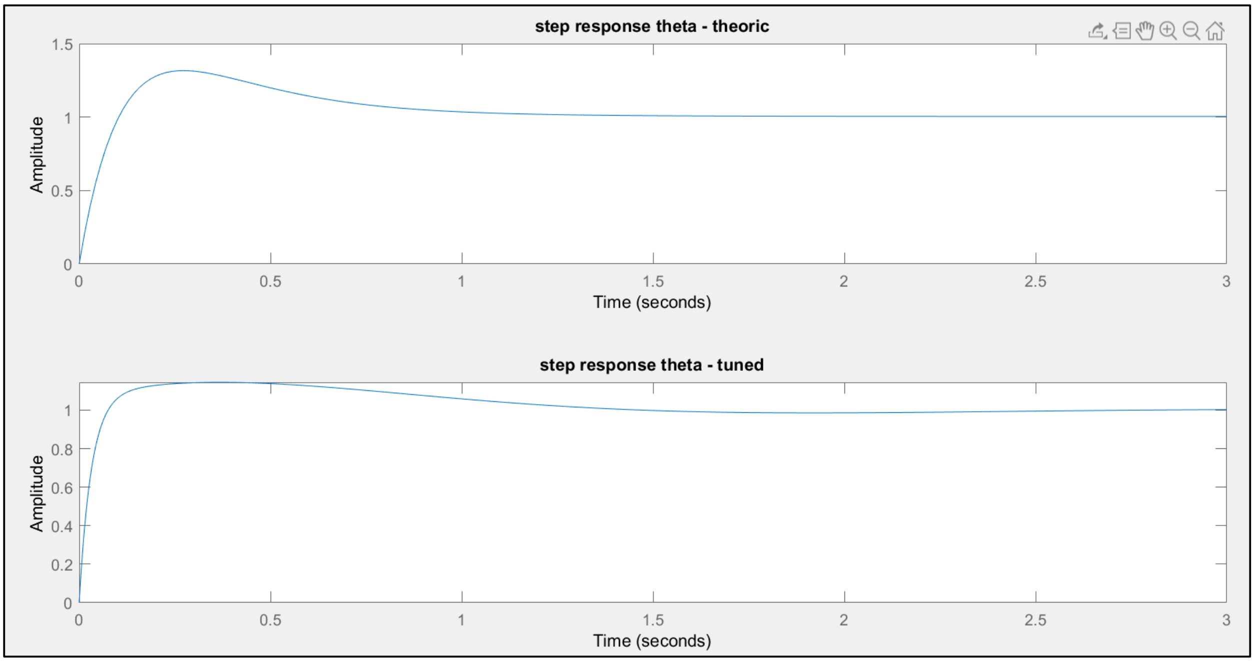
با اعمال این کنترلر به سیستم root locus به شکل زیر در می‌آید:



شکل ۷-۳ : مکان هندسی تابع تبدیل زاویه بعد از اعمال کنترلر

همانطور که دیده می‌شود دیگر به ازای تمامی مقادیر k شاخه در سمت راست نخواهیم داشت. حال سیستم حلقه بسته با فیدبک واحد را تشکیل می‌دهیم و مشاهده می‌کنیم که میزان overshoot و settling time دقیقا مقادیر مطلوب نشدند به این دلیل که تقریب سیستم با سیستم مرتبه ۲ استاندارد خیلی دقیق نبوده به همین دلیل ضرایب PID را خودمان دستی tune می‌کنیم تا به میزان overshoot و settling time مطلوب برسیم.

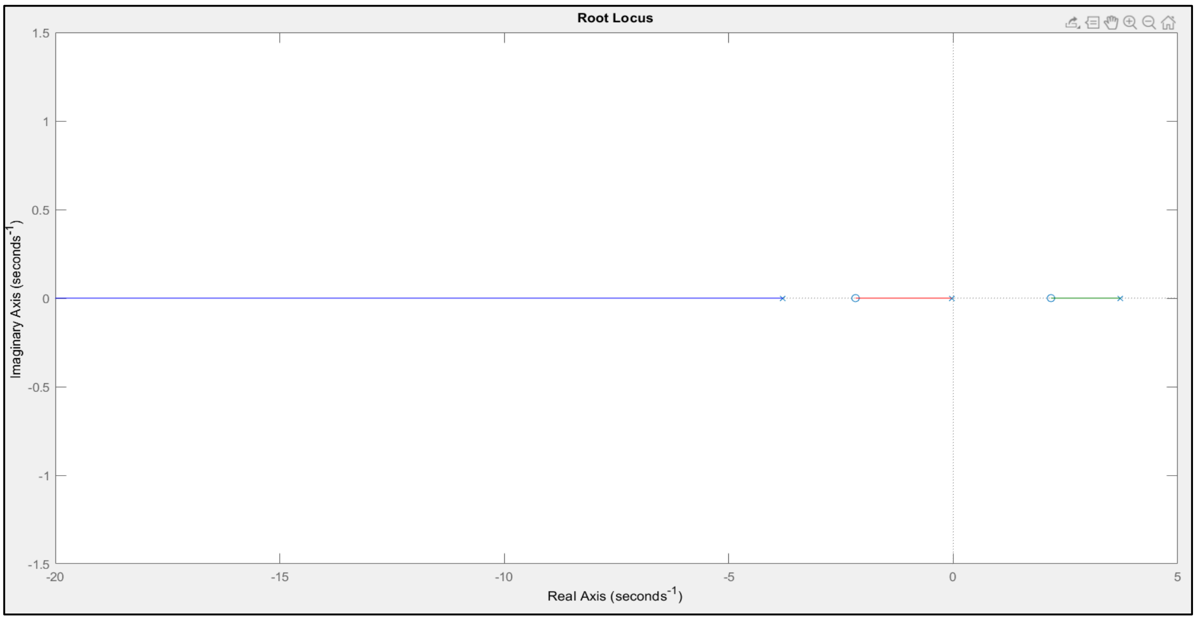
پاسخ به ورودی پله با ضرایب PID تئوری و tune شده به شکل زیر می‌باشد:



شکل ۷-۴ : پاسخ به ورودی پله

کنترلر سرعت:

برای کنترل سرعت نیز همانند کنترل زاویه ابتدا مکان هندسی تابع تبدیل سرعت رسم را رسم می‌کنیم که به شکل زیر می‌باشد:

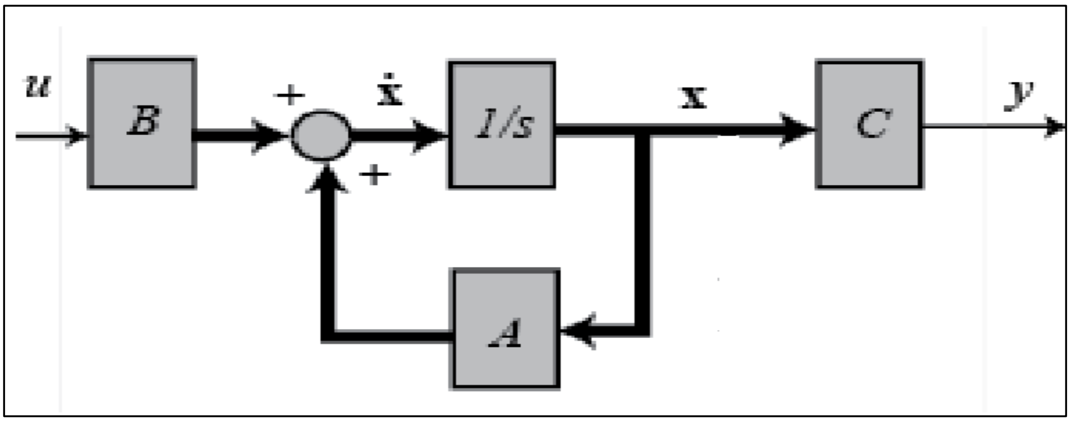


شکل ۷-۵ :مکان هندسی تابع تبدیل سرعت

همانطور که قابل مشاهده است، در این مکان هندسی نیز مانند مکان هندسی تابع تبدیل زاویه یک شاخه‌ی سمت راست به ازای جمیع مقادیر k وجود و علاوه بر آن یک صفر سمت راست نیز دارد و این دو مورد باعث ناپایداری تابع تبدیل ما می‌شود. این تابع تبدیل قابل کنترل فقط با کنترلر PID نمی‌باشد.

در ابتدا برای کنترل این سیستم از این روش استفاده کردیم که ابتدا با اضافه کردن صفر و قطب‌های مناسب، شاخه ی سمت راست تابع تبدیل را طوری تغییر دادیم که از سمت راست شروع شود و در نهایت به سمت چپ ختم شود(که به این روش root locus shaping می‌گویند) و بعد روی سیستم پایدار شده برای رسیدن به overshoot و Ts مطلوبمان از کنترلر PID استفاده کردیم که این روش هم برای پایدار سازی این تابع تبدیل مناسب نبود. پس به دنبال کنترلر دیگری رفتیم و فیدبک حالت را برای کنترل این سیستم مناسب دیدیم.

همانطور که می‌دانیم دیاگرام حالت به شکل زیر می‌باشد:



شکل ۶-۷ : دیاگرام حالت

معادلات آن شکل زیر است:

$\dot{x} = Ax + Bu$

$y = Cx$

و همانطور که می‌دانیم قطب‌های سیستم مقادیر ویژه ماتریس A هستند. در سیستم segway ماتریس A یک ماتریس ۴*۴ می‌باشد پس ۴ قطب داریم.

حال اگر قرار دهیم :

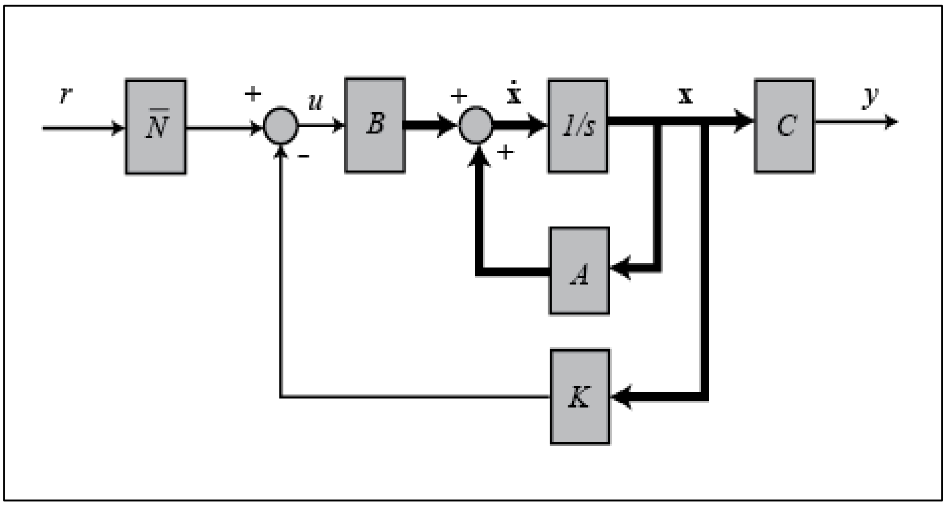
$u = r - \vec{k}\vec{x}$

که K یک بردار می باشد، داریم:

$\dot{x} = (\, A - Bk \,)x + Br$

پس به این صورت ماتریس A تغییر می‌کند و از آنجایی که مقادیر ویژه ماتریس A همان قطب‌های سیستم ما می‌باشند پس قطب‌های سیستم نیز تغییر می‌کنند.

کنترلر فیدبک حالت به شکل زیر عمل می‌کند:



شکل ۷-۶ : کنترل به وسیله فیدبک حالت

واضح است که مقدار ماتریس A عوض شده و مقدار جدید آن A-BK می‌باشد.

$\bar{N}$ هم معکوس مقدار دی سی گین می‌باشد.(دلیل اضافه کردن $\bar{N}$ این است که قبل از اضافه کردن آن یعنی فقط با فیدبک حالت ما می‌توانیم سیستم را پایدار کنیم و مطمئن باشیم این سیستم به ازای هیچ ورودی به بی‌نهایت میل نمی‌کند. حال برای کنترل آن و اطمینان حاصل کردن از اینکه این سیستم به ازای هر ورودی خروجی اش ورودی را دنبال می کند باید gain dc سیستم ۱ شود پس ما برای این کار یک ضریب (یک تقسیم بر گین دی سی تابع تبدیل جدید) به سیستم اضافه می‌کنیم.

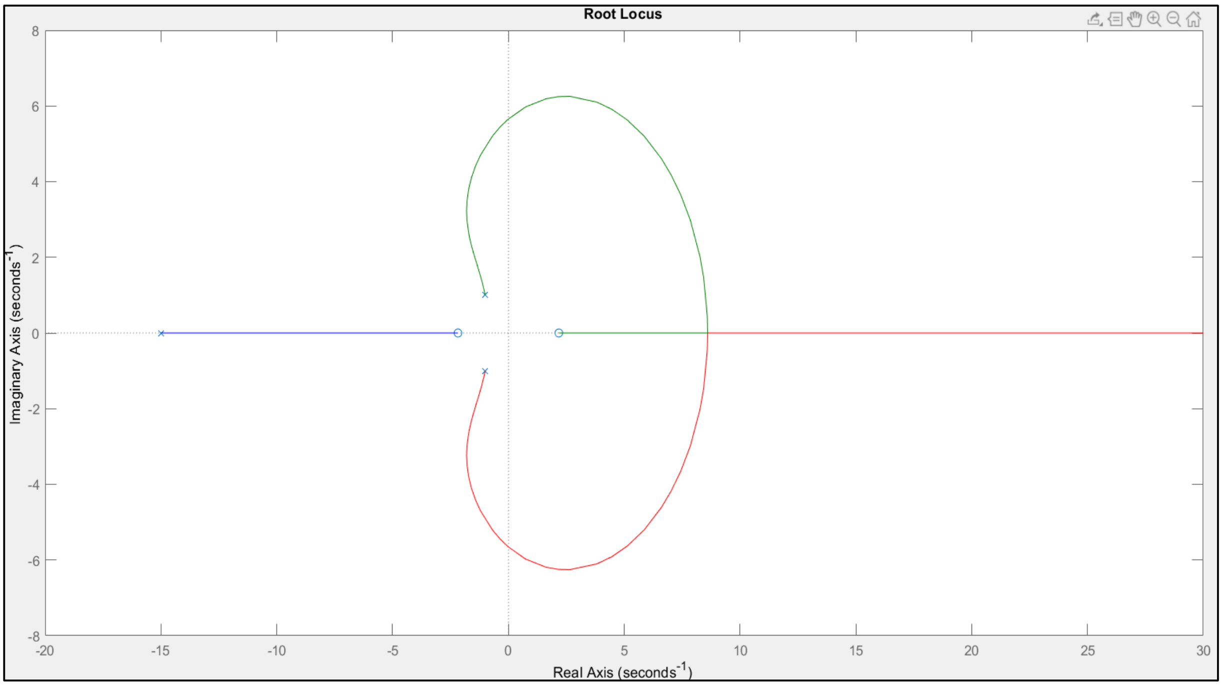
برای محاسبه بردار K از دستور متلب به شکل زیر استفاده می‌کنیم:

([i ۰ -۱۵-K = acher(A,B,[-۱+i -۱

که در ورودی دو ماتریس A و B و یک بردار ۴ تایی که مقادیر آن قطب‌های دلخواه سیستم ما می‌باشند را دریافت می‌کند. در این پروژه با این استدلال مقادیر داده شده که دو قطب اول یعنی ۱-i- و ۱-i قطب‌های

مطلوب سیستم ما برای رسیدن به اورشوت ۴.۴ درصد و Ts=۴s می‌باشد پس برای تقریب تابع تبدیل با تابع تبدیل مرتبه دو استاندارد لازم است که این ها قطب‌های مسلط سیستم باشند و می‌دانیم که قطب مسلط قطبی است که به محور موهومی حداقل ۵ برابر دیگر قطب‌ها نزدیک‌تر باشد. پس قطب دیگرمان را در نقطه‌ی ۱۵- انتخاب می‌کنیم تا قطب غیرمسلط شود. دلیل انتخاب قطب دیگر در صفر، وجود صفر در مبدا می‌باشد که می‌خواستیم اثر آن را خنثی کنیم.

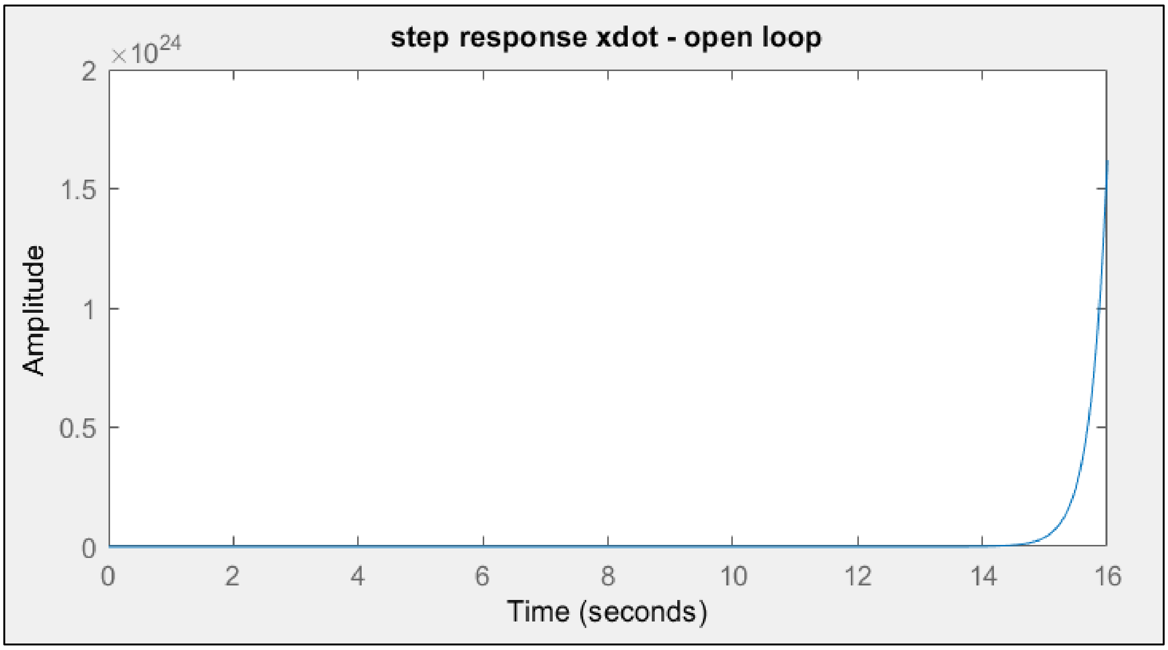
بعد از اعمال کنترل فیدبک حالت مکان هندسی به شکل زیر در می‌آید:



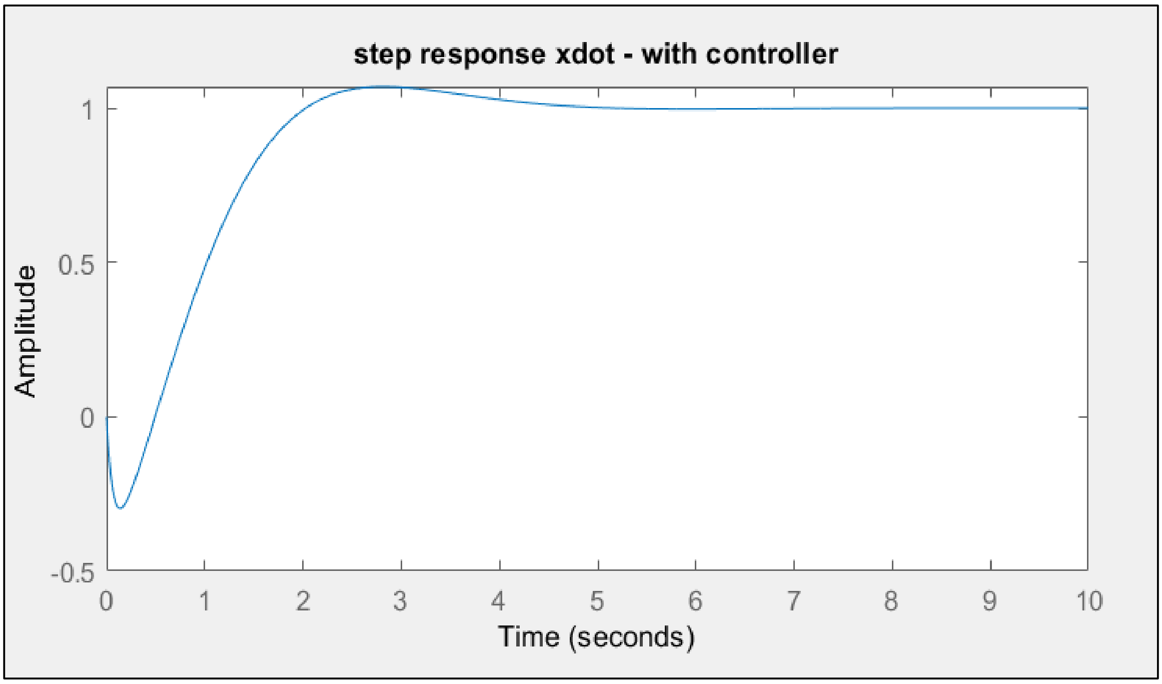
شکل ۷-۷ : مکان هندسی پس از اعمال کنترلر

حال شاخه‌ی سمت راستی ندارد که به ازای تمام مقادیر k در سمت راست باشد.(یا به اصطلاح شاخه‌ی چنبره زده‌ی سمت راست ندارد) و اورشوت آن ۶,۹ درصد و Ts آن ۴ ثانیه می‌باشد که هر دو به مقادیر مطلوب ما بسیار نزدیک است. اما سیستم به میزان ۲۹ درصد Undershoot دارد که دلیل آن وجود صفر سمت راست می‌باشد.

پاسخ پله سیستم قبل و بعد از اضافه کردن کنترل به صورت زیر می‌باشد:



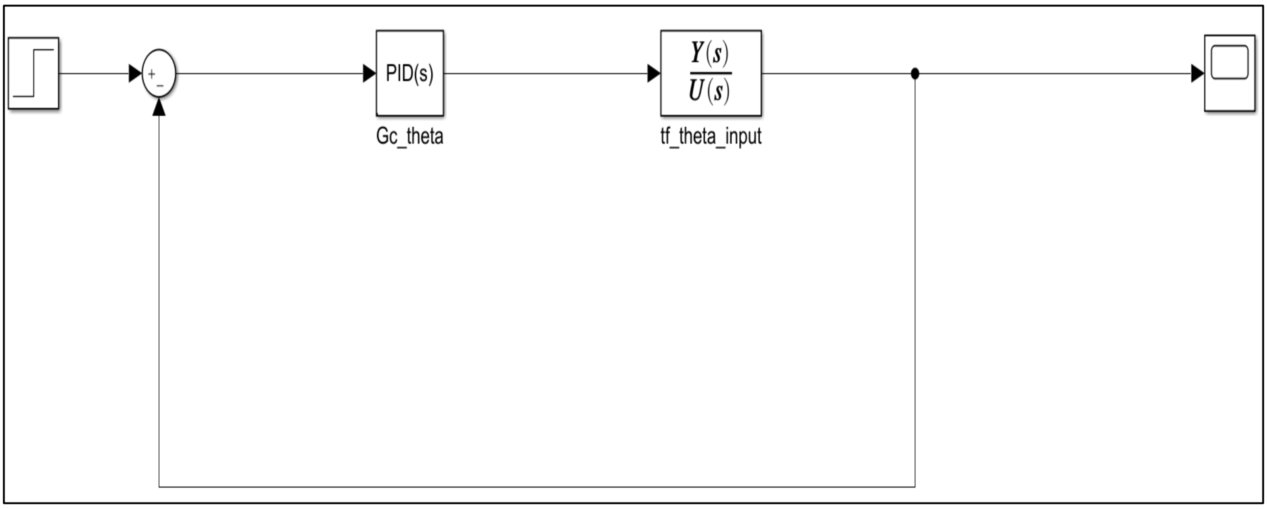
شکل ۷-۸ : پاسخ پله سیستم قبل از اعمال کنترلر



شکل ۷-۹ : پاسخ پله سیستم پس از اعمال کنترلر

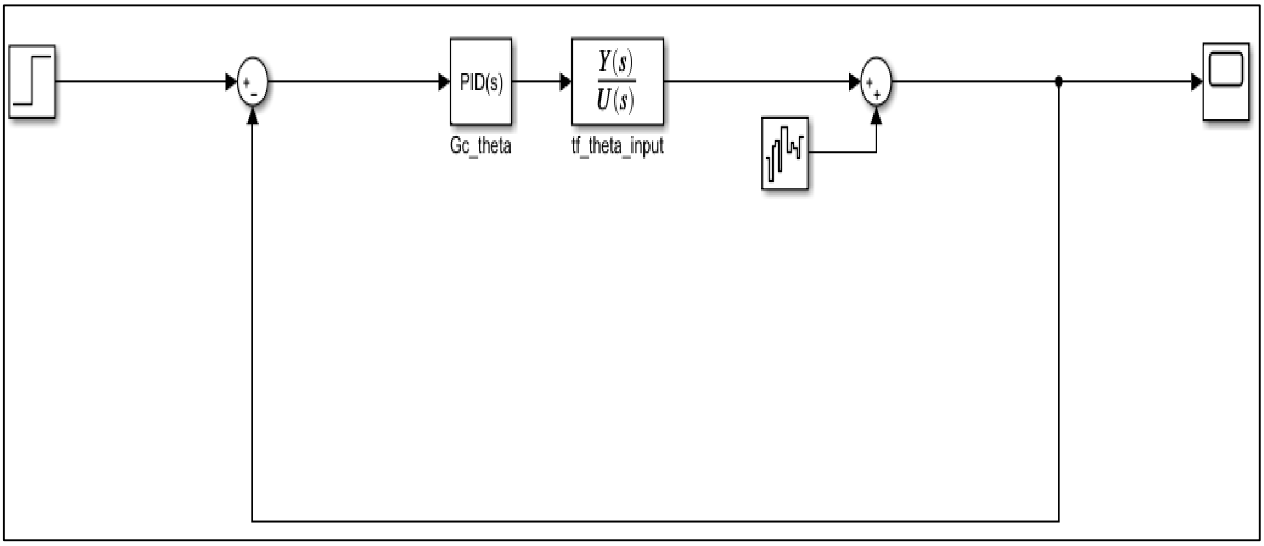
۸– پیاده سازی سیمولینک

در پیاده سازی سیمولینک ابتدا تابع کنترلر زاویه مانند شکل زیر پیاده سازی شده است و پاسخ آن به ورودی پله بررسی شده است.

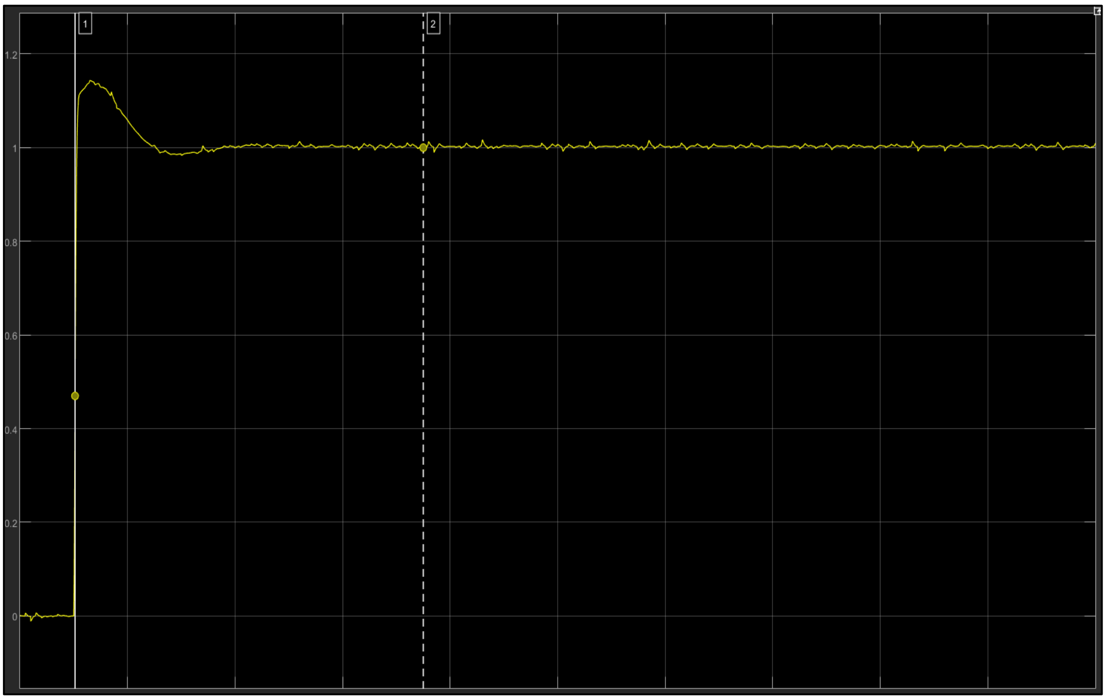


شکل ۸-۱ : system with controller - theta

سپس به این سیستم نویز اضافه شده (شکل۱-۸) و خروجی آن روی اسکوپ نمایش داده شده است.(شکل ۳-۸)

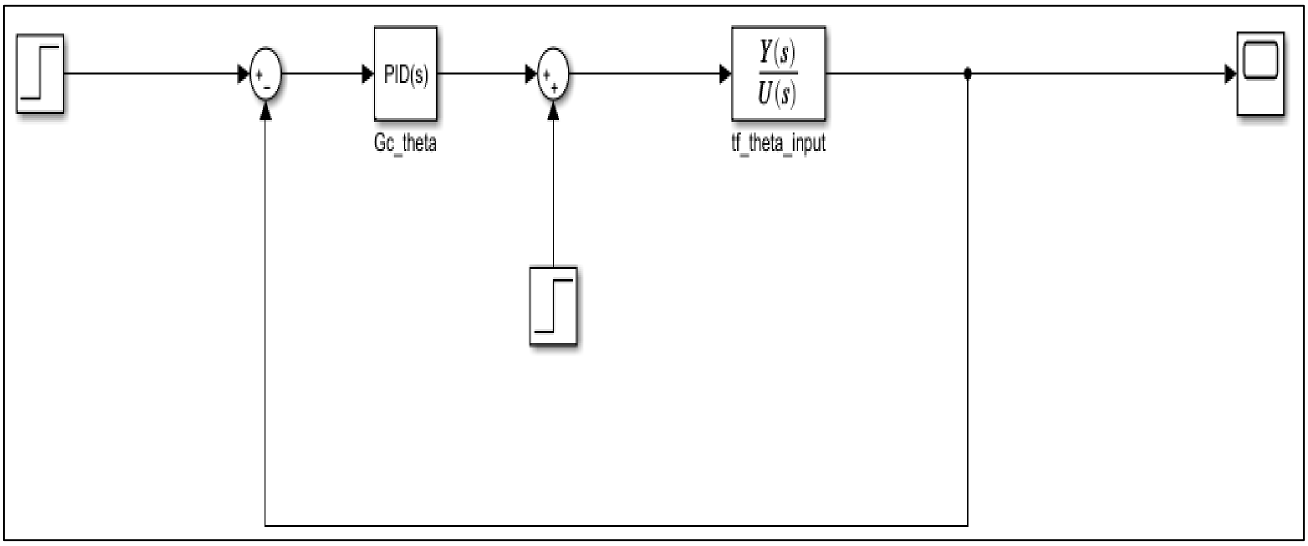


شکل۲-۸ : system with controller - theta- noise

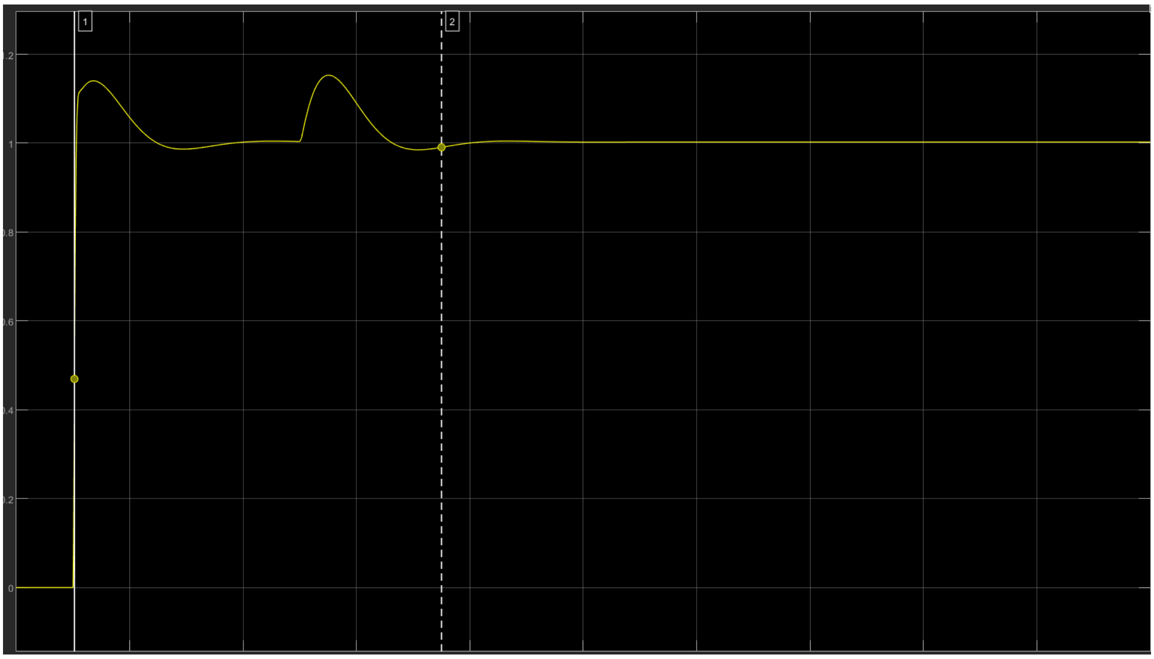


شکل ۸-۳ : نتیجه‌ی اعمال نویز به سیستم

سپس به سیستم اغتشاش اضافه شده(شکل۴-۸) و نتیجه‌ی آن بر روی اسکوپ قابل مشاهده است. (شکل۵-۸)

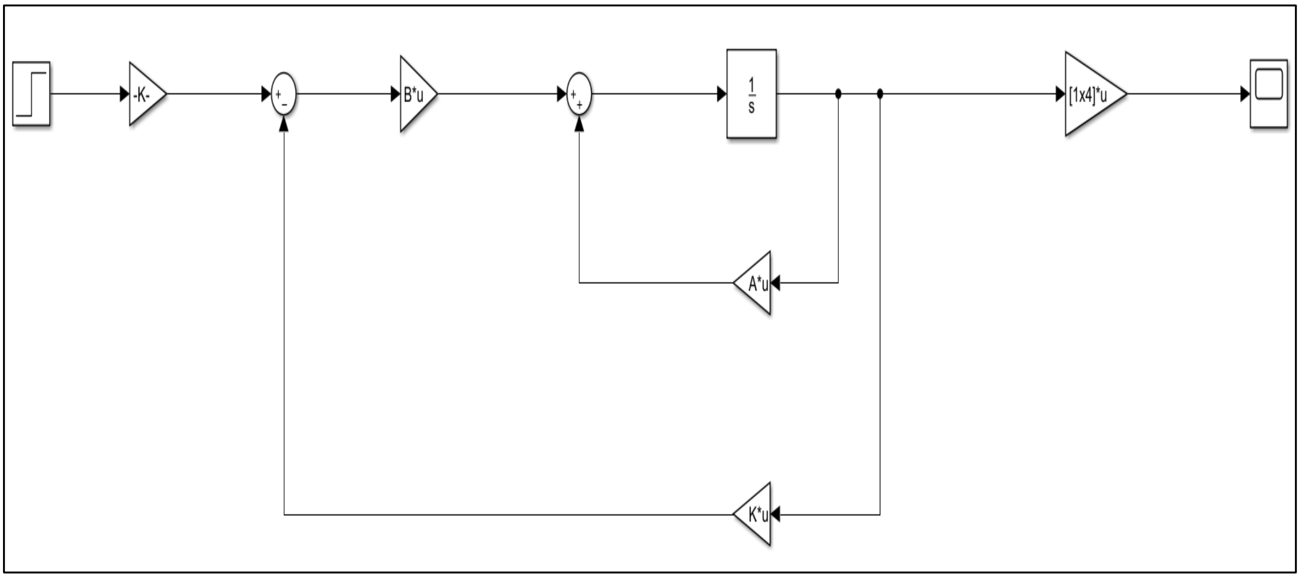


شکل ۸-۴ : system with controller- theta - disturbance



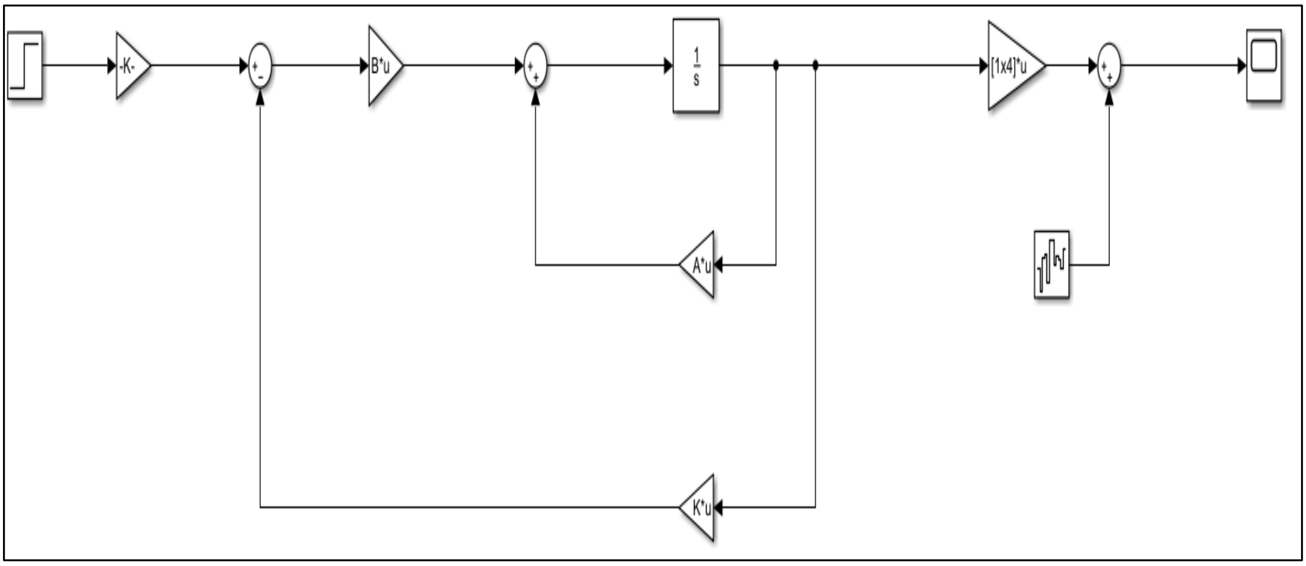
شکل ۸-۵ : نتیجه ی اعمال اغتشاش به سیستم

پس از کنترلر زاویه، کنترلر سرعت در سیمولینک پیاده سازی شده است که به شکل زیر می‌باشد:

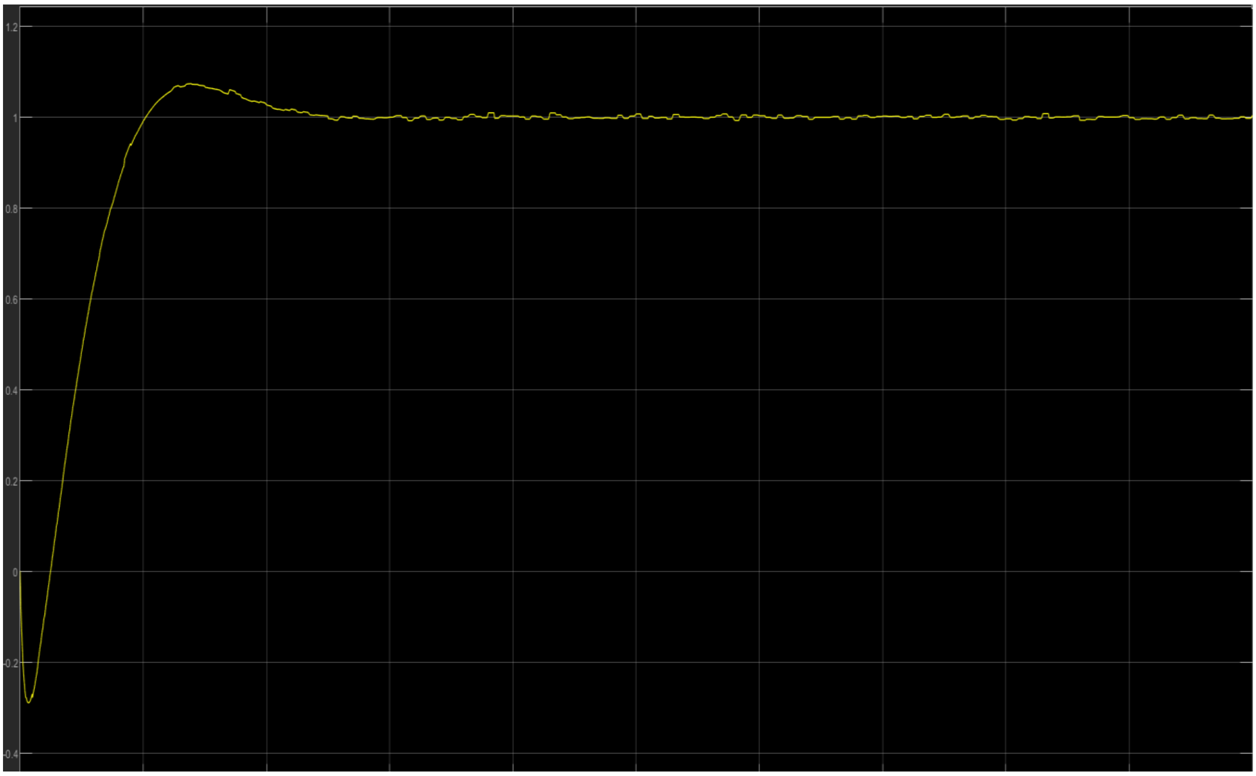


شکل ۸-۶ : system with controller – xdot

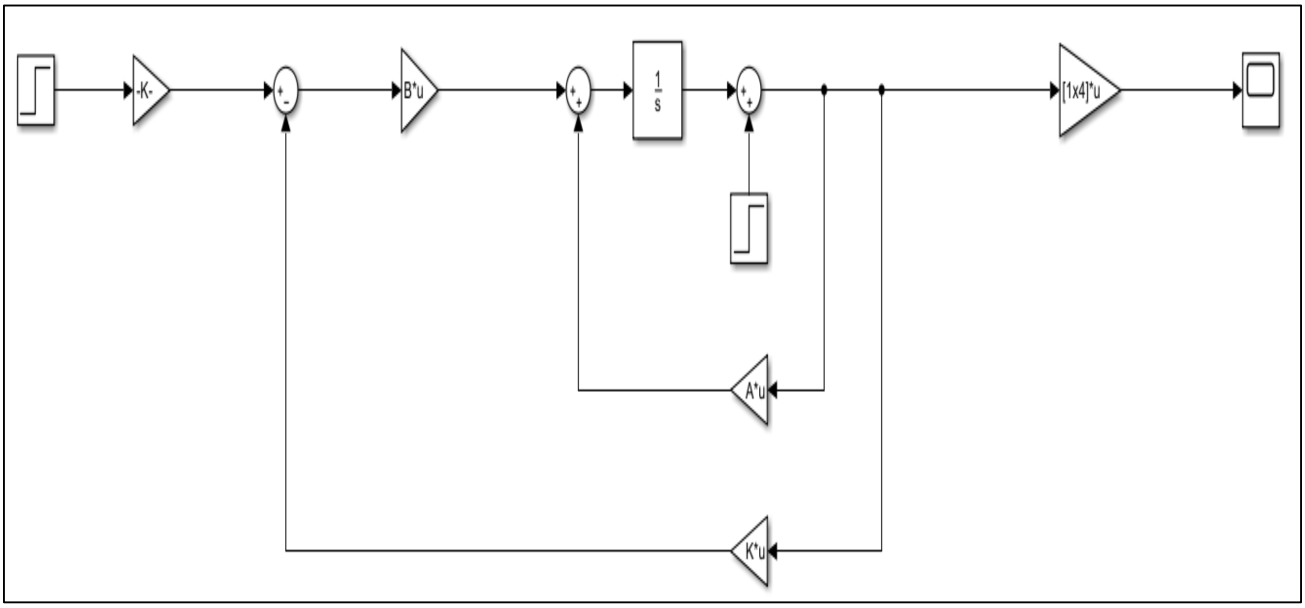
سپس همانند کنترلر قبلی تاثیر نویز(شکل۸-۷) و اغتشاش(۸-۹) روی آن بررسی شده و نتایج حاصل بر روی اسکوپ نمایش داده شده است.(شکل ۸-۸ و ۸-۱۰)



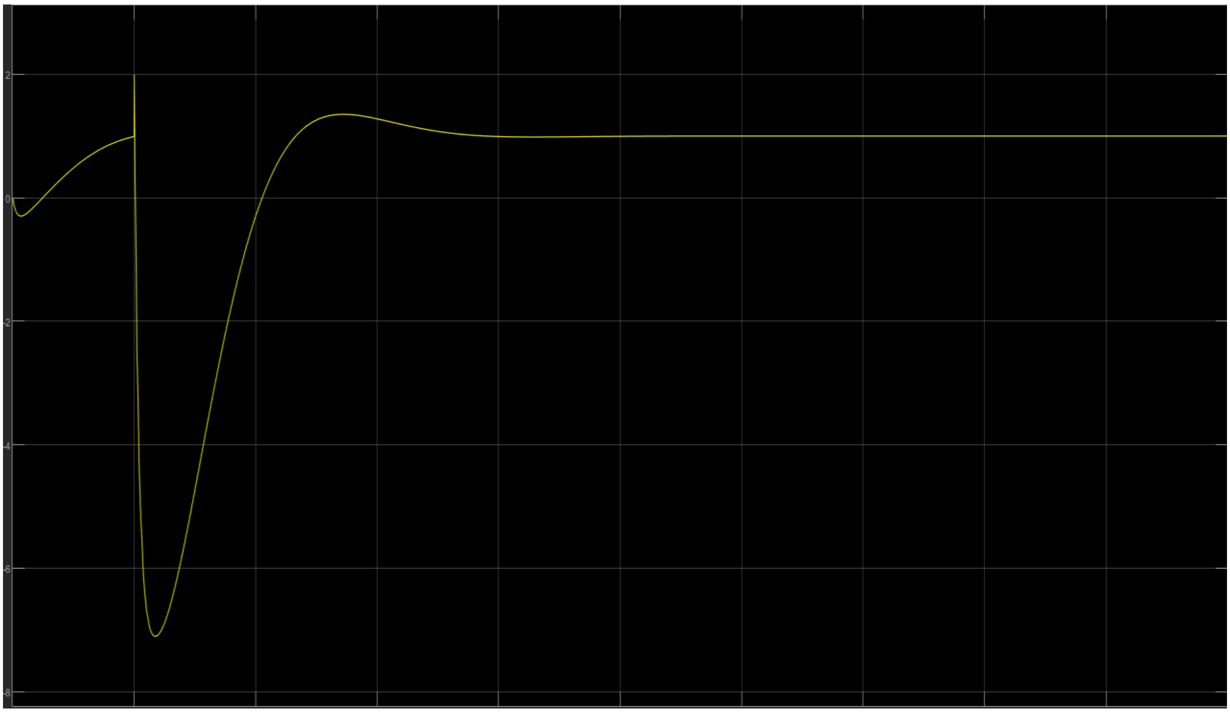
شکل ۸-۷ : system with controller - xdot - noise



شکل ۸-۸ : نتیجه‌ی اعمال نویز به سیستم

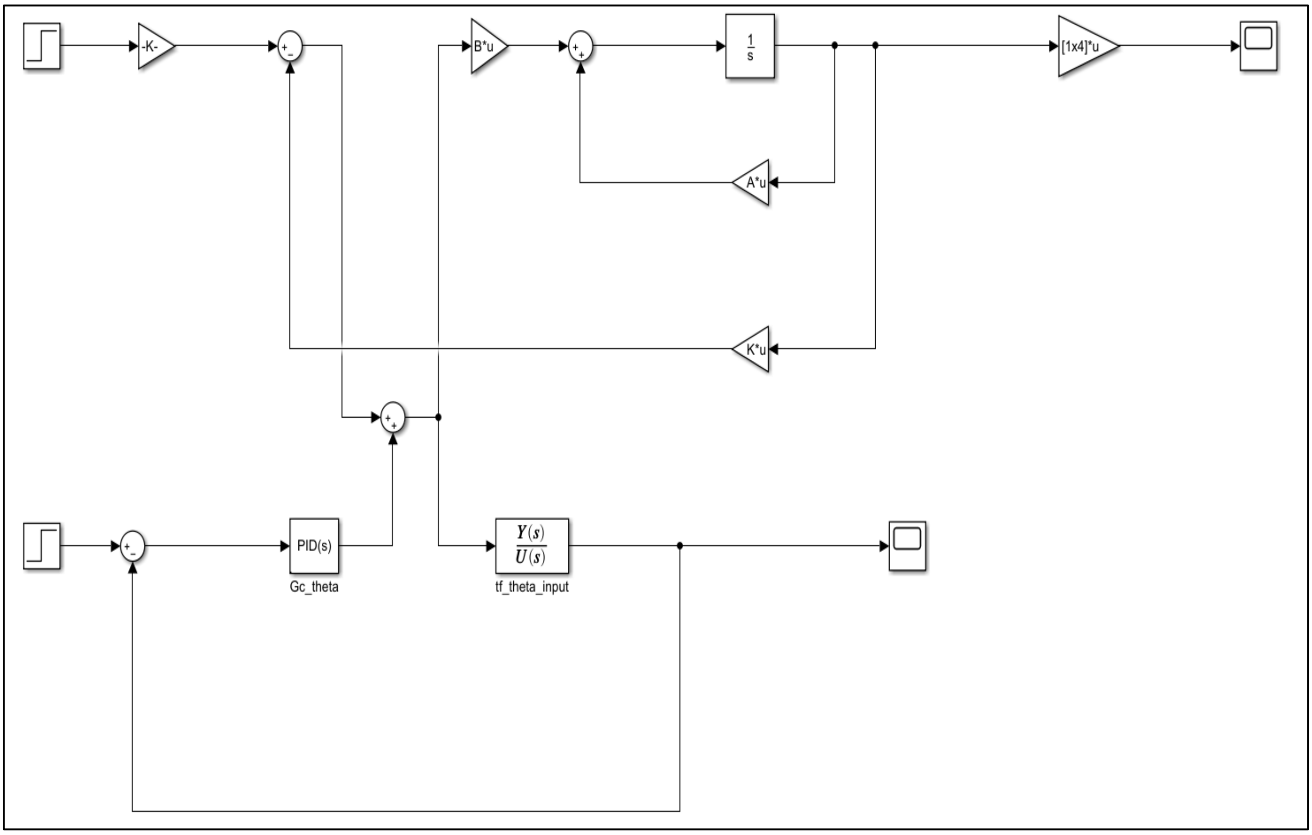


شکل ۸-۹ : system with controller - xdot - disturbance



شکل ۸-۱۰ : نتیجه‌ی اعمال اغتشاش به سیستم

ولی باید توجه داشت که سرعت و زاویه هر دو در یک سیستم کنترل می‌شوند. برای شبیه سازی سیستم segway باید نتیجه‌ی حاصل از هر دو کنترلر که بر روی سیستم ها زده شده است، با هم جمع شود و به ورودی توابع تبدیل زاویه و سرعت داده شود پس نتیجه‌ی شبیه سازی نهایی به شکل زیر می‌باشد.



شکل ۸-۱۱ : نتیجه‌ی نهایی شبیه سازی سیستم segway کنترل شده در سیمولینک

۹– شبیه ساز گرافیکی

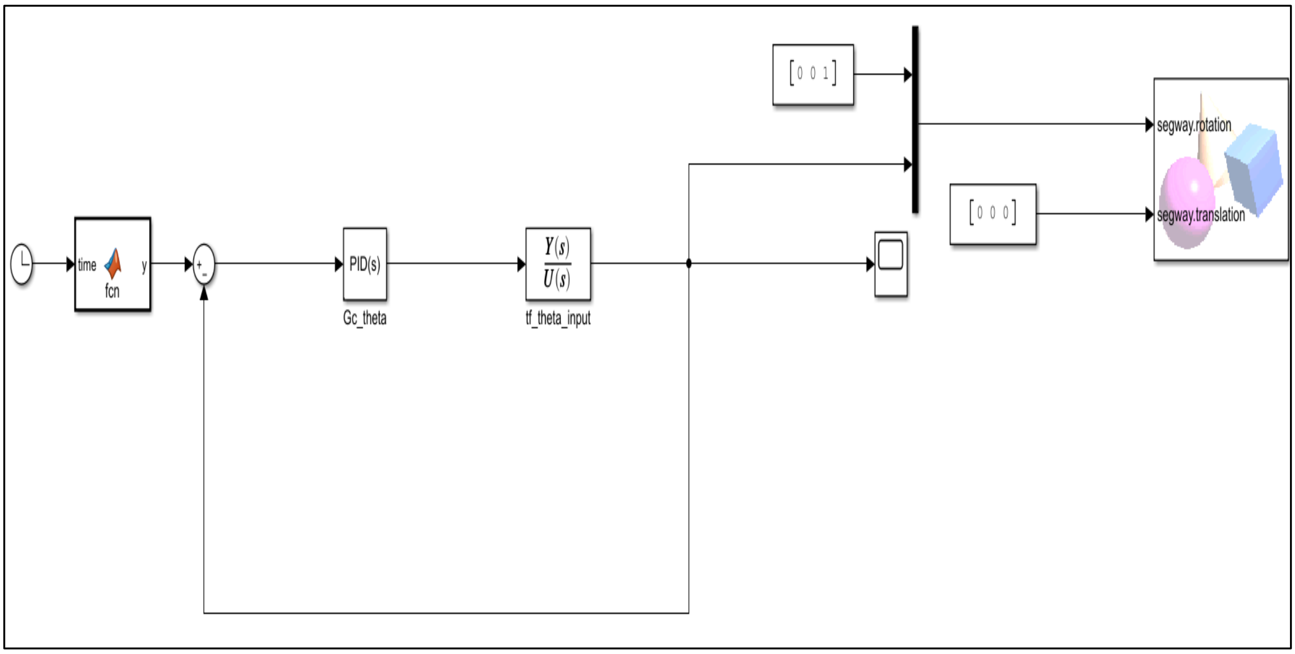
vrml یک واقعیت مجازی است که قابلیت نمایش گرافیکی خروجی‌ها را در متلب فراهم می سازد.(باید توجه شود که شبیه ساز نیست فقط نمایش‌دهنده است.)

برای طراحی این شبیه ساز ابتدا یک فایل wrl برای مدل segway،که به صورت آماده در اینترنت موجود می باشد نیاز داریم.

در متلب برای پیاده سازی این شبیه ساز یک بلوک vr sink قرار داده شده و فایل wrl تولیدی به آن داده شده است.

سپس با تعیین rotation و translation به عنوان ورودی‌های بلوک vr sink، به translation یک بردار سه‌تایی به عنوان ورودی داده شد و به rotation با استفاده از mux یک ورودی برای جهت زاویه و یک ورودی برای مقدار زاویه داده شده است.

که البته در این پروژه شبیه ساز گرافیکی فقط روی سیستمی که زاویه‌ی آن به عنوان خروجی کنترل شده پیاده شده است.



شکل ۹-۱ : پیاده‌سازی سیستم با شبیه ساز گرافیکی

۱۰- منابع

[۱] M. Abdelati and W. Younis, "Design and Implementation of an Experimental Segway Model," ۲۰۰۹.

[۲] J. van der Veen, "Stabilization and Trajectory Tracking of a Segway," University of Groningen Faculty of Science and Engineering, ۲۰۱۸.

[۳] R. Babazadeh, A. Gogani Khiabani and H. Azmi, "Optimal control of Segway personal transporter," in *۲۰۱۶ ۴th International Conference on Control, Instrumentation, and Automation (ICCIA)*, ۲۰۱۶.