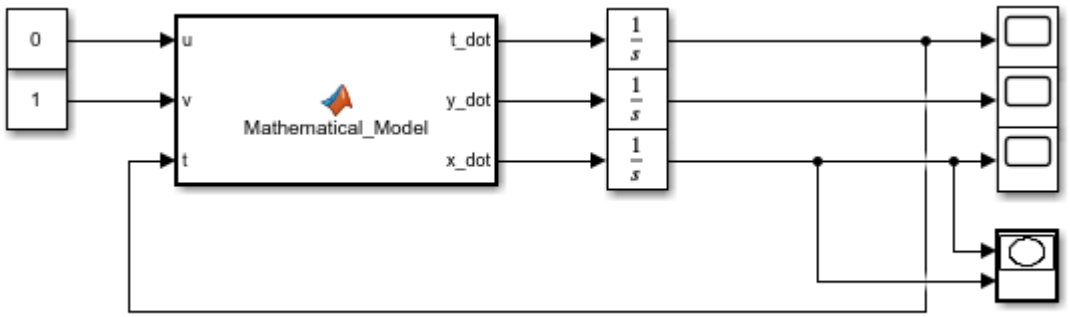
به نام خدا

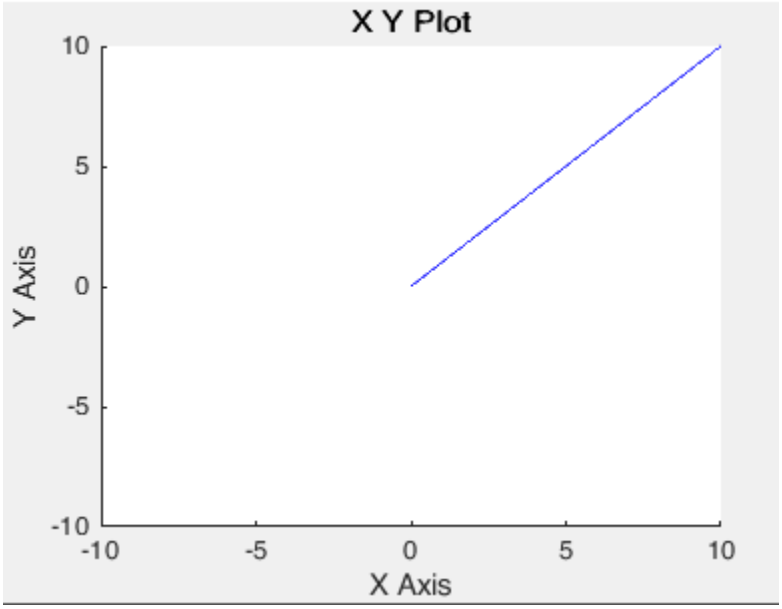
ارسلان فیروزی – ۹۷۱۰۲۲۲۵

تمرین سوم درس آشنایی با سیستم‌های رباتیک

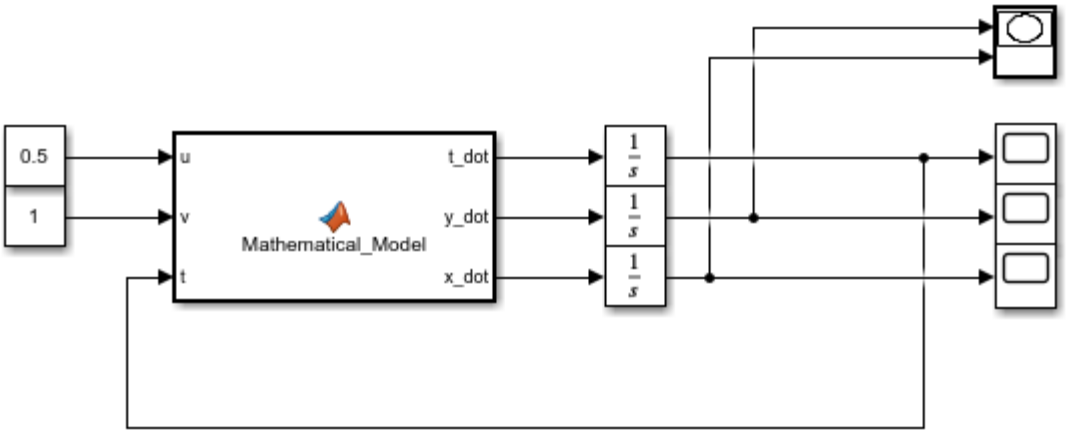
۱. شماتیک استفاده شده در سوال یک:

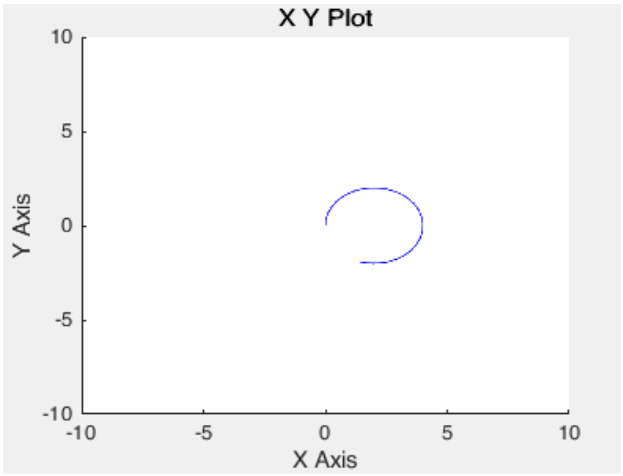


جهت سنجش عملکرد درست مدل ریاضی ربات، با ورودی سرعت خطی ۱ متر بر ثانیه و سرعت زاویه ای صفر تست کردم و انتظار داشتم که ربات تنها رو به جلو حرکت کند:

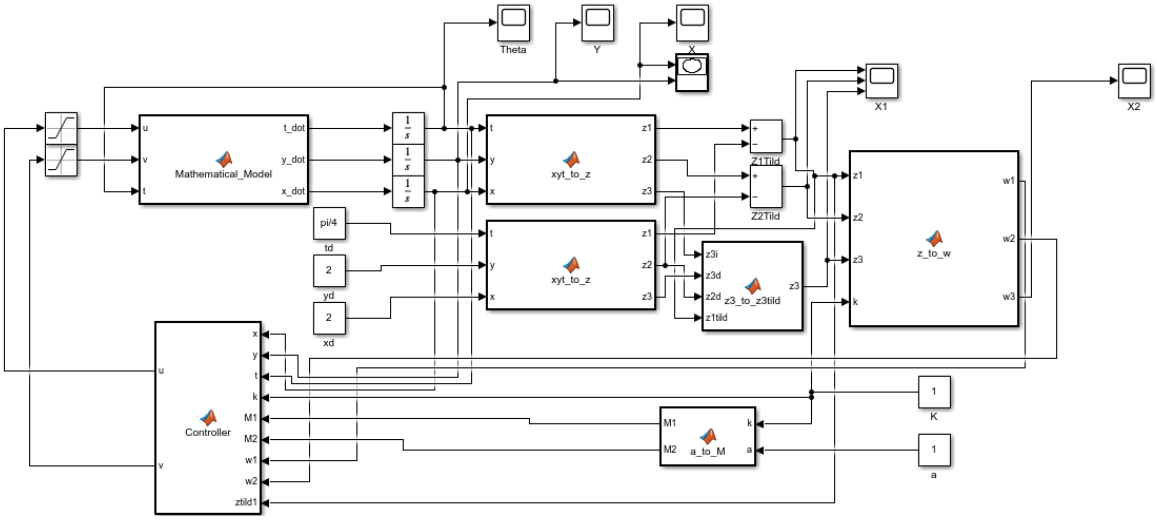


همچنین با ورودی سرعت خطی ۱ متر بر ثانیه و سرعت زاویه ای ۰.۵ رادیان بر ثانیه نیز تست کردم و انتظار داشتم که یک مسیر مارپیچ طی شود:





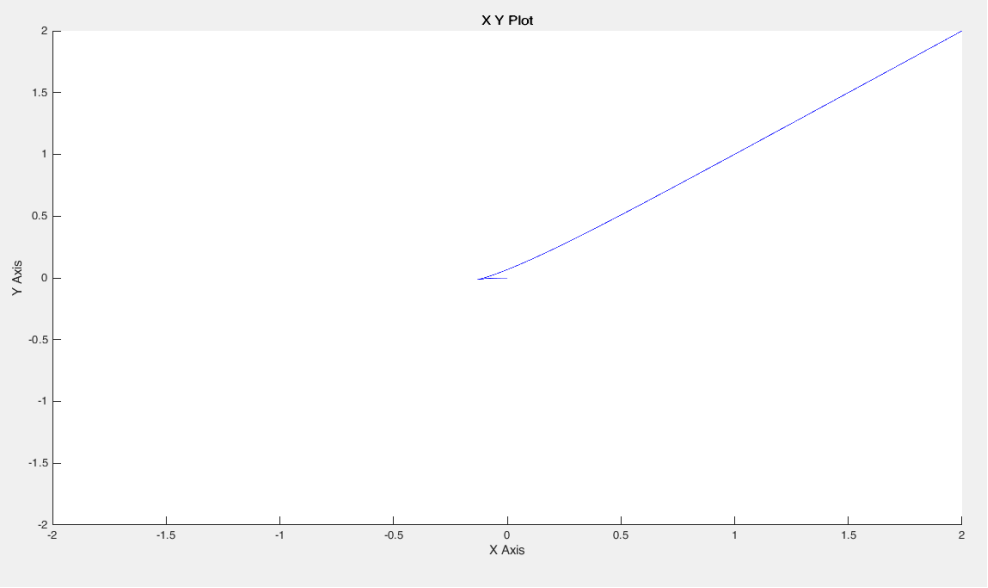
۲. در این سوال از شماتیک زیر جهت پیاده سازی کنترلر استفاده کردم:



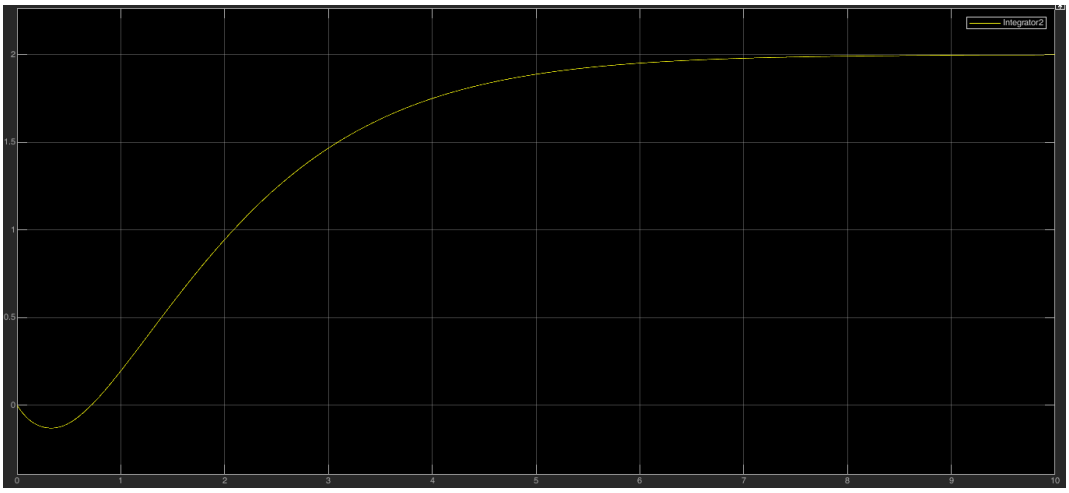
از ۲ بلوک اشباع کننده که محدودیت های حداکثر سرعت خطی و زاویه ای را پیاده سازی می‌کند استفاده کردم. کنترلر با استفاده از تبدیل متغیر ها از فضای x,y,theta به فضای Z که معادلات دیفرانسیل در آن زنجیره ای بود استفاده شد و سپس از فضای Z به $\tilde{Z}$ و از فضای $\tilde{Z}$ به ω انتقال دادم. از همان روابط بدست آمده و معرفی شده در جزوه جهت اینکار استفاده شده است. سپس به صورت عددی قانون کنترل را پیاده سازی کردم. در پیاده سازی قانون کنترل فرض کردم که در مرحله بدست آوردن قانون کنترل مناسب پس از رسیدن به معادلات LTI، مقادیر ویژه ماتریس بدست آمده هردو a- هستند (a>0). و به همین دلیل در کل ۲ متغیر جهت تعیین در قانون کنترل ظاهر شد که سپس بر اساس تاثیر آن ها بر روی جواب نهایی ماکسیمم مقدار آن ها را پیدا کردم که شرط بیشتر شدن مقدار سرعت خطی و زاویه ای از مقدار بیشینه نقض نشود.

با توجه به شبیه سازی و سعی و خطا بیشینه مقدار k برابر با ۳ و بیشینه مقدار a برابر با ۱ شد.

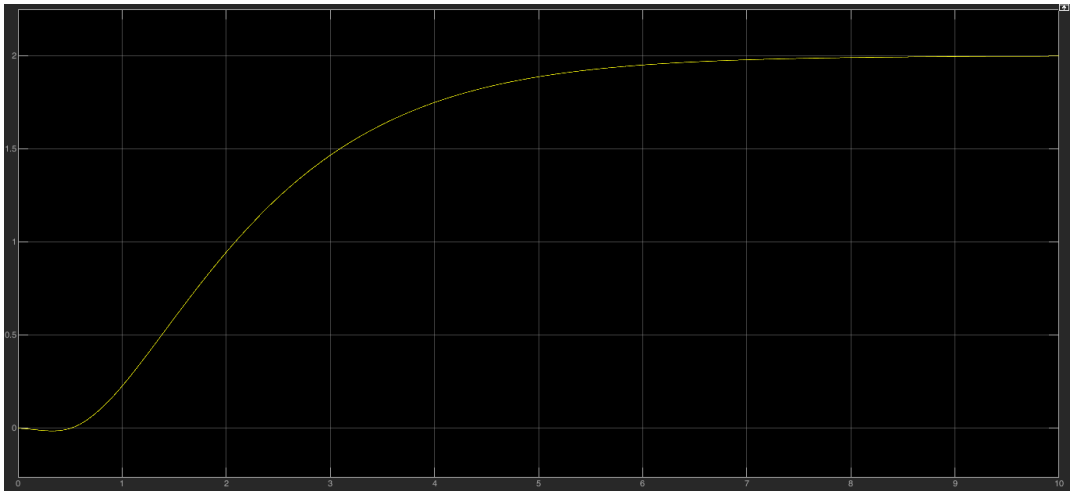
مسیر حرکت ربات به ازای هدف $[2, 2, \frac{\pi}{4}]$:



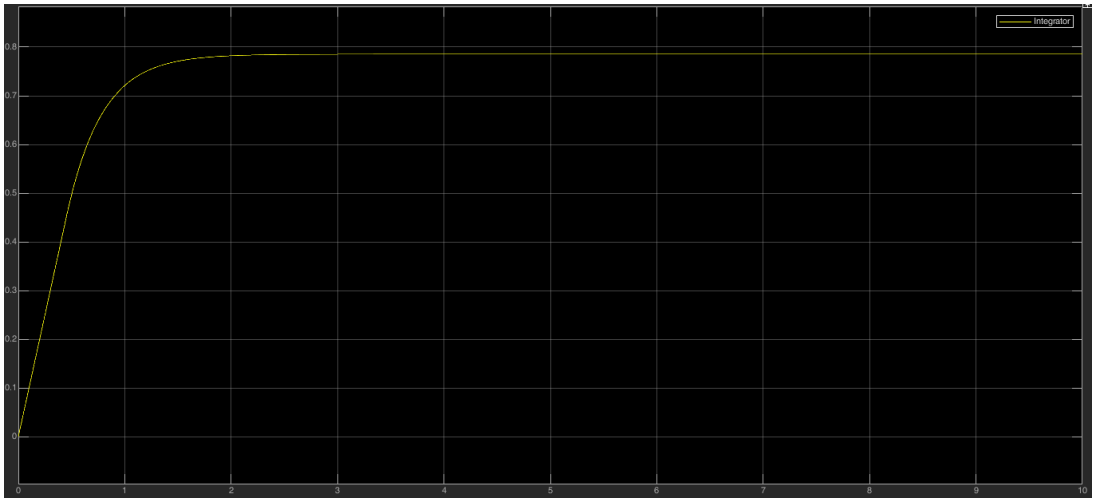
نحوه تغییرات x در طول زمان:



نحوه تغییرات y در طول زمان:

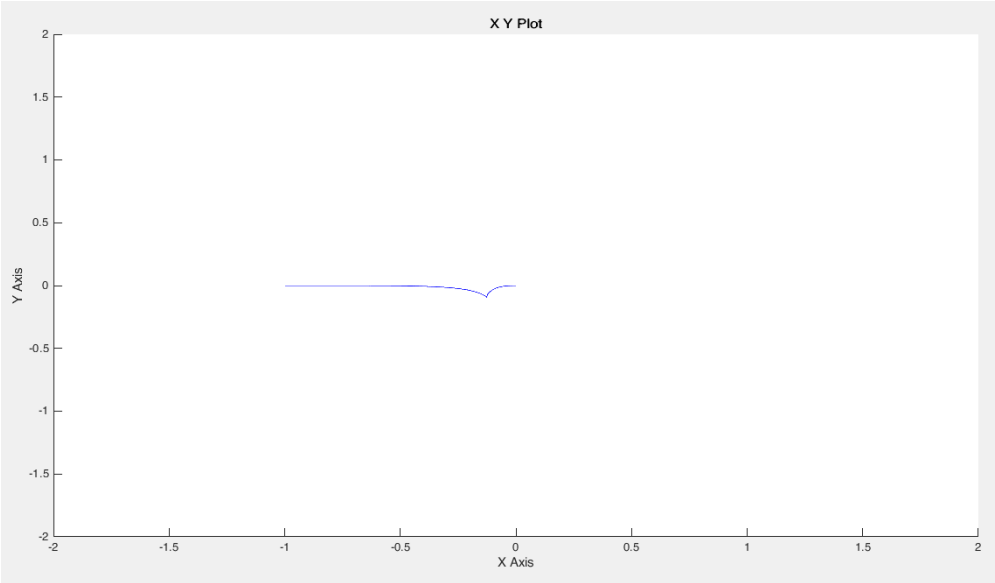


نحوه تغییرات Theta در طول زمان:

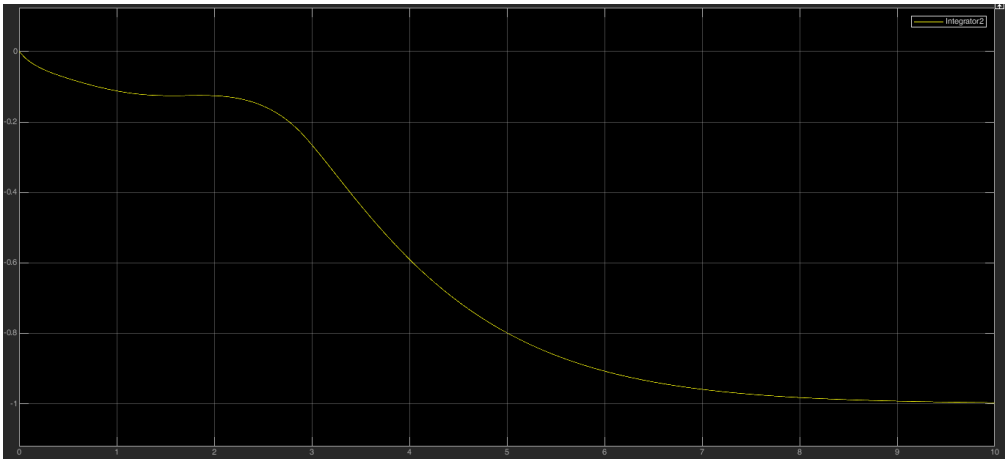


نتایج نشان می‌دهد تقریبا در ثانیه ۷ از شروع حرکت ربات، به مقصد می‌رسد. اینطوری بنظر میرسد که ربات ابتدا به عقب حرکت میکند و همزمان زاویه خود را اصلاح میکند، سپس بعد از اصلاح زاویه به سمت مقصد مکانی حرکت می‌کند.

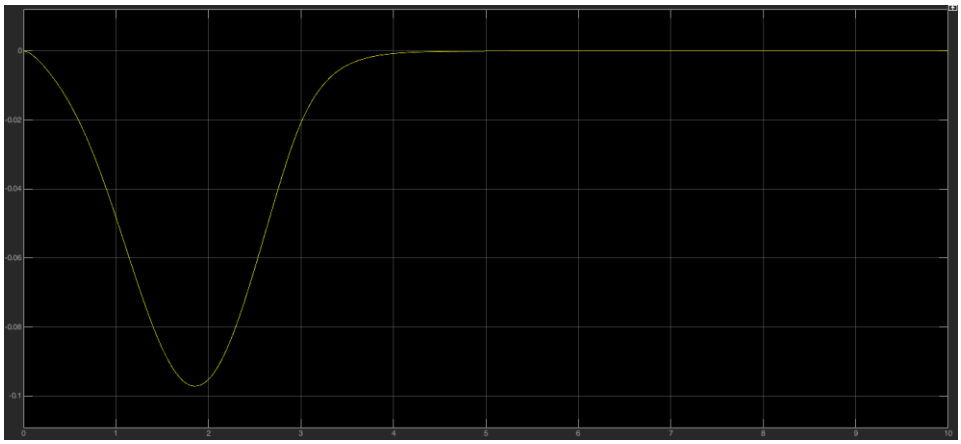
۳. برای مقصد $[-1, 0, \pi]$ مسیر حرکت ربات به صورت زیر شد:



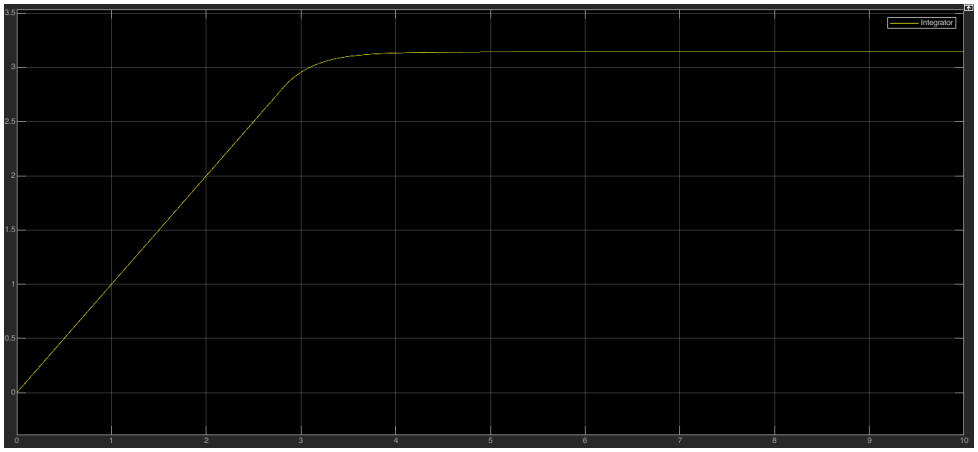
نحوه تغییرات x در طول زمان:



نحوه تغییرات y در طول زمان:

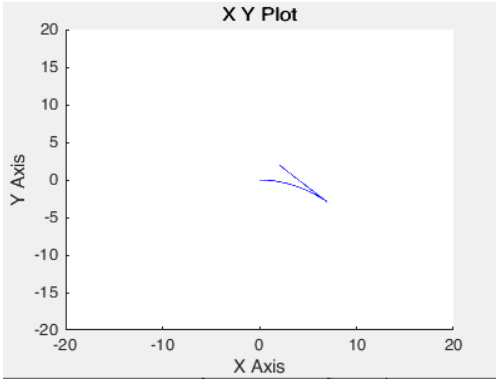


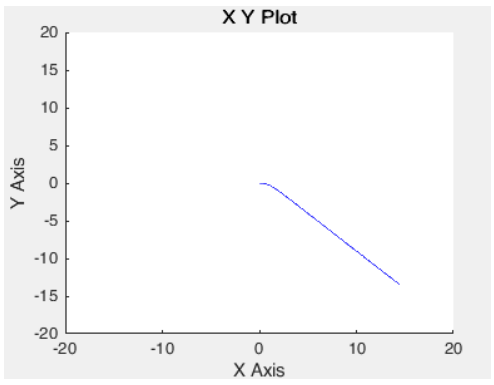
نحوه تغییرات Theta در طول زمان:



بنظر میرسد ربات همزمان به حرکت به سمت عقب زاویه خود را تا حدی اصلاح می‌کند و سپس همزمان با حرکت رو به جلو به سمت مقصد مکانی زاویه خود را کاملا اصلاح میکند.

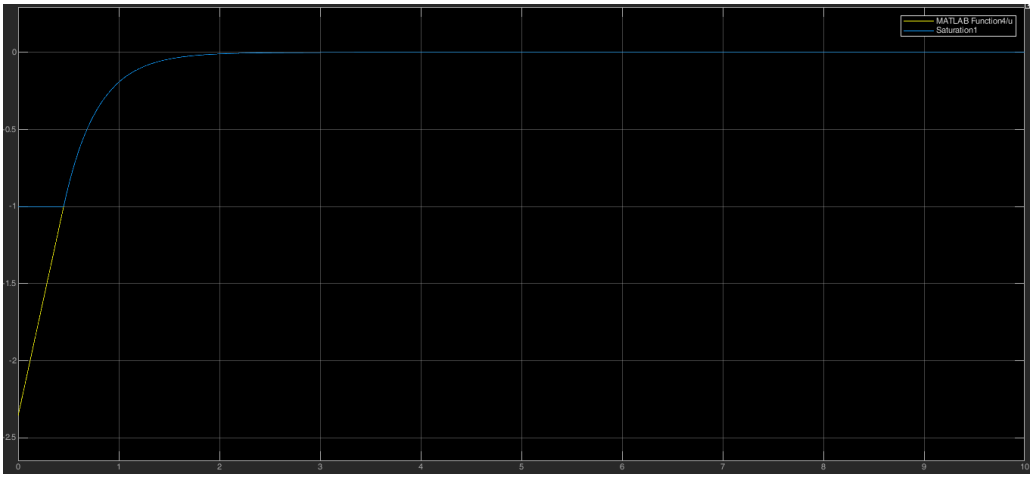
۴. برای مقصد $\left[2, 2, -\frac{\pi}{4}\right]$ مسیر حرکت ربات به صورت زیر شد (در راست با اعمال محدودیت بیشینه سرعت و در سمت چپ بدون در نظر گرفتن آن):



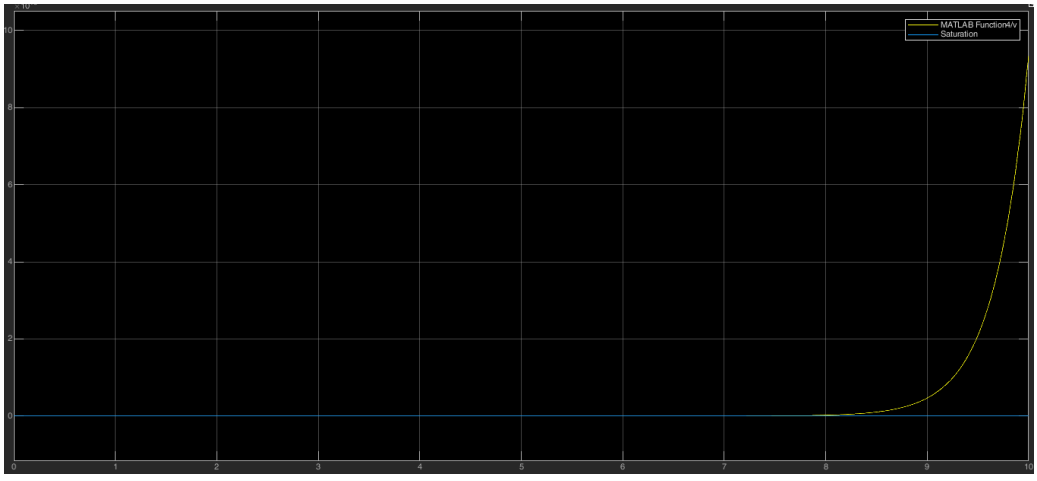


محدودیت سرعت باعث میشود که عملکرد ربات آنطور که کنترلر انتظار دارد محقق نشود. در اصل پس از اینکه ربات زاویه خود را اصلاح کرد در راستی مسیر گذرنده از مختصات ۲ و ۲، با عقب رفتن به آن می‌رود. اما با اعمال محدودیت این اتفاق نمی افتد چون زاویه به زاویه دلخواه نزدیک میشود و برابر نمیشود و آن هم به علت تفاوت سیگنال خروجی کنترلی و اعمال به ربات است:

سرعت زاویه‌ای:

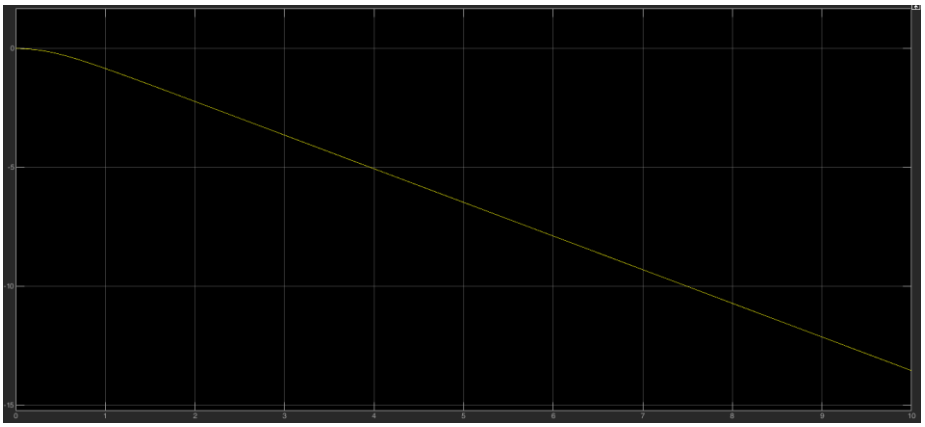


سرعت خطی:

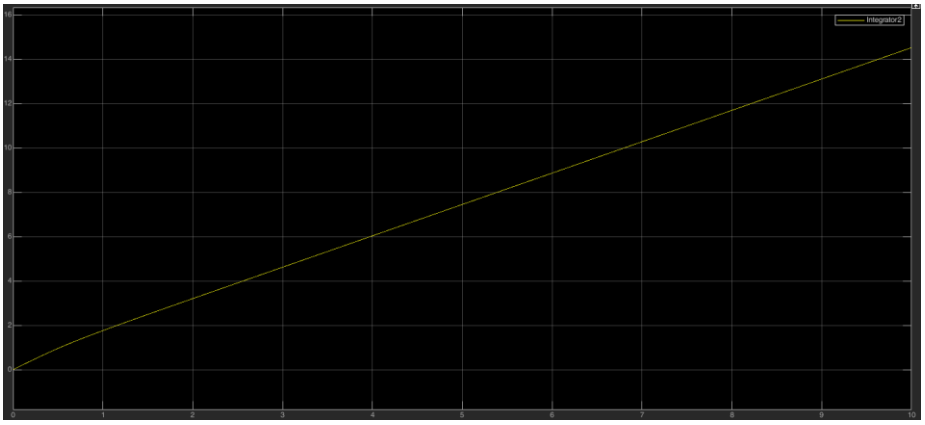


برای رفع این مشکل با ضرایب K و a کار کردم و تغییر دادم اما به نتیجه نرسیدم و بنظر می‌رسد که محدودیت ذاتی بیشینه سرعت خطی و زاویه ای باعث ایراد می‌شود.

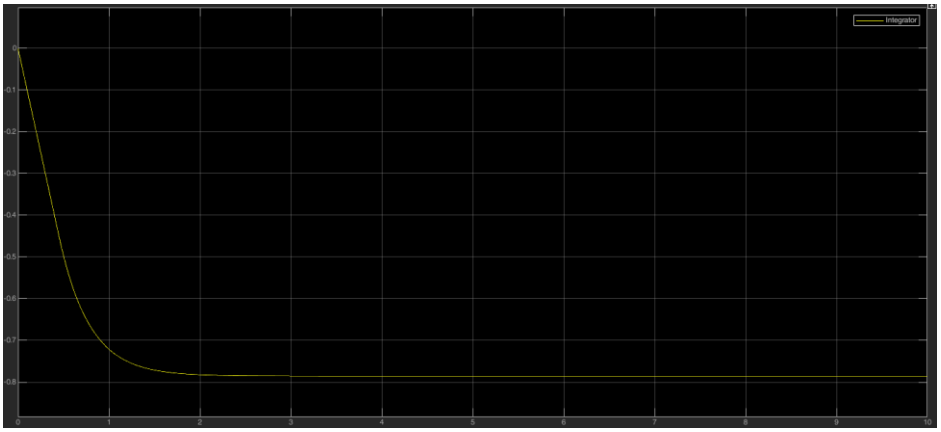
سیگنال y:



سیگنال x:



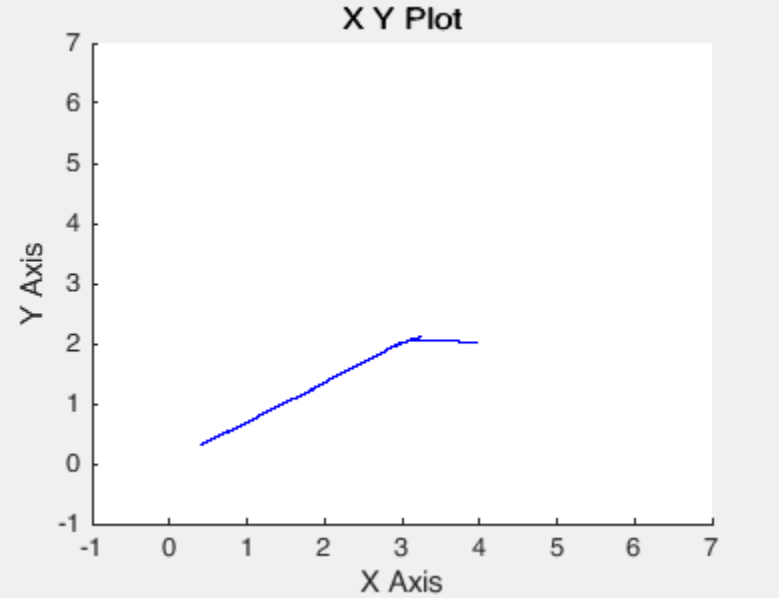
سیگنال تتا:



۵. من در این سوال با فرض اینکه موانع توسط ربات شناخته شده است، ابتدا از طریق الگوریتمی مسیر با کمترین مسافت و هموار را با ۲ مرحله بدست می‌آورم. مرحله اول پیدا کردن تمام مسیرهای بدون برخورد با موانع و سپس در مرحله دوم انتخاب مسیری که

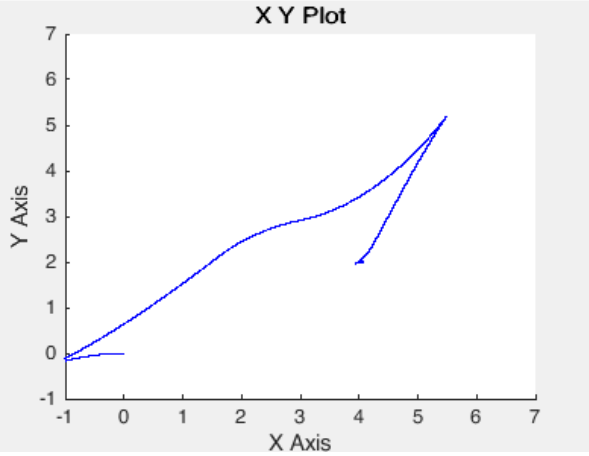
گوشه تیز نداشته باشد و کوتاه‌ترین مسیر باشد. این روش در پکیج Shortest Path with Obstacle Avoidance (لینک) پیاده سازی شده است. من این مسیر را پیدا کردم و تعدادی نقطه در این مسیر انتخاب کردم. سپس به ربات ماموریت دادم که این نقاط را یکی پس از دیگری طی کند تا به مقصد برسد. کد این کار در main.m قرار دارد (از توابع دیگری نیز استفاده کردم تا که در کنار فایل main قرار دارد.). ایراد اصلی که در این روش هست، نامعلوم بودن زاویه حرکت مسیر است. که من این زاویه را تقریب زدم ولی کنترلر ربات به نحوی است که تعداد خیلی زیادی عقب و جلو می‌کند تا بتواند به آن زاویه و نقطه برسد و این مسیر حرکت را بسیار ناهموار میکند و همچنین به دلیل محدودیت سرعت بیشینه خطی و زاویه‌ای، رفتن به هرنقطه دلخواه همانطور که در بخش (d) دیدیم، امکان پذیر نیست. به این دو دلیل این روش نامناسب بود و من آن را در فایل‌های Qe_2.slx و main.m قرار دارد. اما من در ادامه راه دیگری برای حل سوال انجام دادم که بعد از گزارش نتایج روش اولیه، آن را توضیح میدهم:

مسیر حرکت اشتباه ربات:

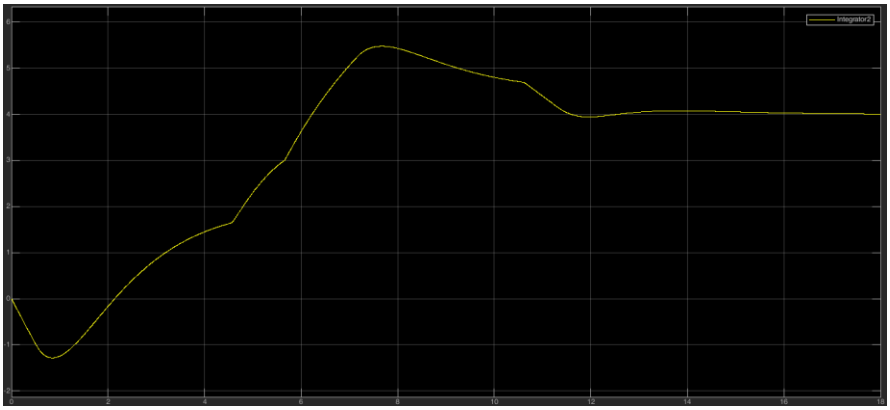


برای رسیدن به پاسخ سوال تعدادی نقطه (۴ نقطه) به صورتی که به موانع برخورد نکند، انتخاب کردم و تلاش کردم تا ربات به صورت مشابه پارک دوبل در جای مناسب پارک کند. فایل این قسمت در Qe.slx قرار دارد.

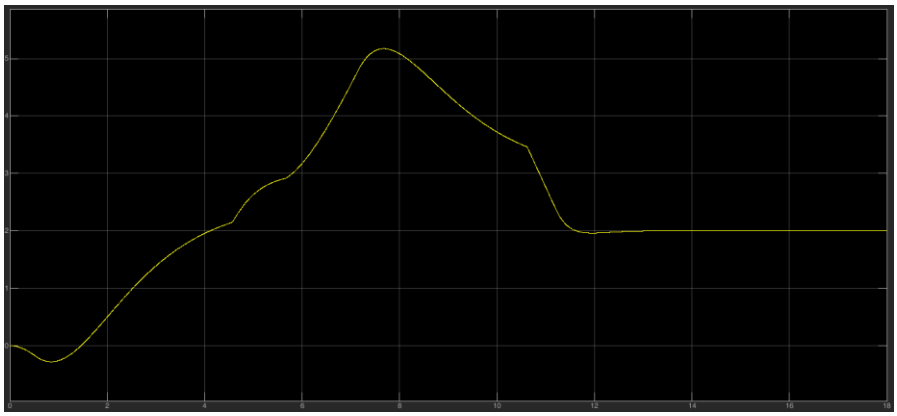
خروجی با این روش:



تغییرات x در طول زمان:



تغییرات y در طول زمان:



که همانطور که مشاهده می‌کنید بدون برخورد با موانع میتواند در جای مناسب با زاویه صفر قرار گیرد.