آرم

دانشگاه

پروژه درس کنترل فازی

استفاده ازروش کنترل فازی تعمیم یافته تکراری جهت کنترل ربات با استفاده از دینامیک معکوس

نام و نام خانوادگی

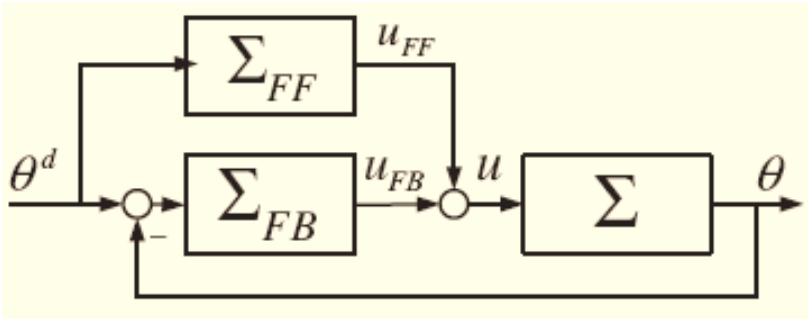
استاد:

فروردین ۱۳۹۳

استفاده ازروش کنترل فازی تعمیم یافته تکراری جهت کنترل ربات با استفاده از دینامیک معکوس

مقدمه

در بسیاری از کاربردهای عملیاتی، جایی که ردیابی یک مسیر ویژه بسیار مهم می باشد، می توان بلوک دیاگرام کنترلی را به صورت زیر در نظر گرفت:



شکل(۱): ساختار feedback و feedforward یک سیستم کنترلی کلی

که به طور مشخص از یک کنترل کننده فیدبک دار و یک کنترل کننده در مسیر پیشرو بهره برده است[۱]. u_{FF} در حقیقت سیگنالی است که باعث رسانیدن خروجی به خروجی مطلوب خواهد شد.و سیستم فیدبک برای کنترل و پایدارسازی سیستم مدل در حضور نامعینی ها مورد استفاده قرار خواهد گرفت. متاسفانه با تقاضای بسیار زیاد صنعت برای استفاده از این روش کنترلی، کمتر در صنایع این روش کنترلی مورد استفاده قرار گرفته است. علت اصلی فقدان استفاده از این روش را می توان، در مشکل بودن استفاده از آن در سیستم های غیرخطی و مدل کردن آنها جویا شد. سیستم های دو ورودی دو خروجی معمولا سیستم های تزویج شده غیرخطی می باشند. درحقیقت مشکل بودن سیستم های چند ورودی چند ورودی ، در حقیقت کنترل کردن آنها با از بین بردن اثر اندرکنش بین کانال ها و درجات آزادی حرکتی آنها می باشد. در حقیقت کنترل کننده تطبیقی برای از بین بردن نامعینی در مدل های سیستم های چند ورودی چند خروجی مطرح شده است. که نمونه های انها را می توان در بازوهای ربات دو درجه آزادی، کنترل زاویه حرکتی وسایل نقلیه زیرآبی و

کنترل زاویه آتش هواپیما در نظر گرفت. اما به دلیل دینامیک بسیار پیچیده آنها، استفاده از کنترل کننده های تطبیقی مشکلات فراوانی را در تحلیل و پیاده سازی سیستم فراهم می آورد. گذشته از این به دلیل اینکه تعدای از پارامترهای طراحی همچون نرخ یادگیری ، به نوع و ساختار کنترل کننده بستگی دارد، اغلب روش های ارائه شده دارای محدودیت و نقص هایی خواهند بود.

کنترل کننده های فازی، جز با قوام ترین کنترل کننده ها در مدل سازی و پیاده سازی سیستم های پیچیده کنترلی هستند. و می توانند خروجی های مطلوبی را برای مدل سیستم حتی با وجود انواع کنترل کننده های طراحی شده برای آن سیستم تولید کنند. مجموعه بهره های فازی با داشتن حداقل خطا برای کنترل سیستم، قوام لازمه را برای داشتن عملکردی مطلوب فراهم می آورند. کاملا مشخص است که کنترل کننده فازی معمولی، برای کنترل سیستم حلقه بسته با حضور فیدبک و مسیر کنترلی پیشرو مناسب نمی باشد. از اینرو از کنترل کننده فازی تعمیم یافته برای کنترل سیستم حلقه بسته استفاده کرده ایم.هدف اصلی در این پروژه دنبال کردن یک مسیر مشخص توسط ربات با استفاده از روش کنترل فازی تعمیم یافته است. ابتدا به بررسی روش inverse dynamics control (IDC) برای کنترل ربات دو درجه آزادی پرداخته و سپس با ایجاد یک سیستم استنتاج فازی تعمیم یافته و اعمال روش کنترل فازی برای آن، سعی در کاهش لرزش و ایجاد حرکت نرم در تعقیب مسیر مربعی می‌کنیم، در مرحله بعد نیز از یادگیری تکراری [Repetitive learning control (RLC)] برای بهبود عملکرد سیستم بهره می‌بریم. به این ترتیب که از داده‌های کنترل قبلی برای ایجاد سیگنال کنترلی جدید استفاده می‌کنیم. این روش در واقع همان روش Iterative learning control (ILC) می باشد که شرایط اولیه برای ربات،در هر بارتکرار،Reset نمی گردد. در واقع دلیل استفاده از این روش،این است که اطلاعات کمتری از سیستم کنترل شده نیاز دارد و توانایی این را دارد که به صورت اتوماتیک سیگنال کنترلی خطای ورودی را بر اساس تکرار های قبلی بدست آورد. ابتدا مدل دینامیکی معمول ربات با استفاده از روش کنترلی دینامیک معکوس آن پیاده سازی شده است، در مرحله بعد به جای مدل دینامیکی از مدل فازی تعمیم یافته ربات استفاده گردیده است. در نهایت روش تکراری یادگیری را در تعقیب مسیر برای تکرارهای بالاتر استفاده می نماییم و نتایج را در محیط متلب شبیه سازی می کنیم.

معرفی سیستم

یک ربات دو درجه آزادی در شکل (۱) نشان داده شده است که دارای یک شانه با چرخش 2pi رادیان و یک آرنج باچرخش 3pi/2 رادیات می باشد.مدهای حرکتی ولرزشی ربات در صفحه ی x-y مورد بررسی قرار می گیرند.معادلات غیر خطی دینامیکی ربات به صورت زیر است:

$$\mathbf{U} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\dot{\mathbf{q}}, \mathbf{q})$$

$$\mathbf{M}(\mathbf{q}) = \begin{bmatrix} mL^2(1.666+\cos q_2) & mL^2(0.333+0.5\cos q_2) \\ mL^2(0.333+0.5\cos q_2) & 0.333mL^2 \end{bmatrix}$$

$$\mathbf{C}(\dot{\mathbf{q}}, \mathbf{q}) = -(0.5mL^2 \sin q_2)\begin{bmatrix} \dot{q}_2(2\dot{q}_1+\dot{q}_2) & -\dot{q}_1 \end{bmatrix}^{\mathrm{T}}$$

$\mathbf{U} = [u_1, u_2]^{\mathrm{T}} =$ torque vector control law

$L = L_1 = L_2 = 4.5$ meters = length of each link

$m = 1.5075$ kg

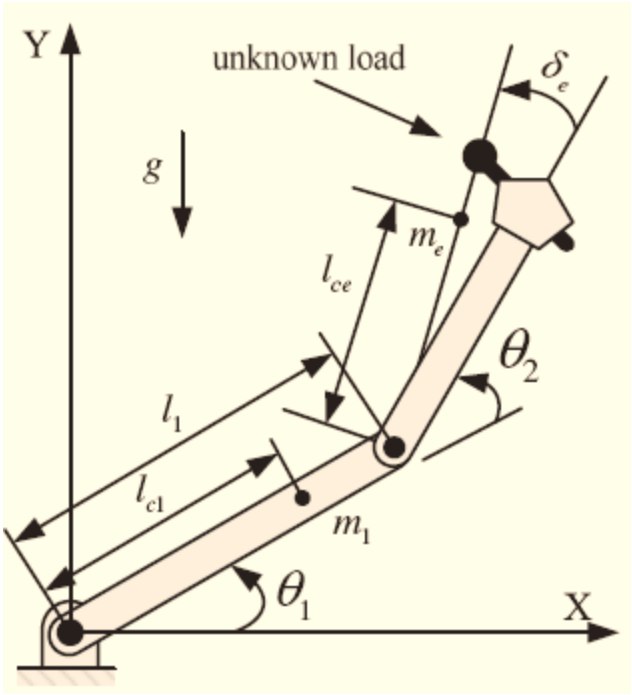
$q_i =$ slew angle, $i = 1, 2$

$\mathbf{q} =$ slew angles vector

$\mathbf{q} =$ slew angular velocities vector

$\zeta = 0.707$, closed-loop damping ratio

$\omega = 8.21$ Hz, first open-loop frequency mode



شکل(۲): مدل بازوی ربات در صفحه x-y

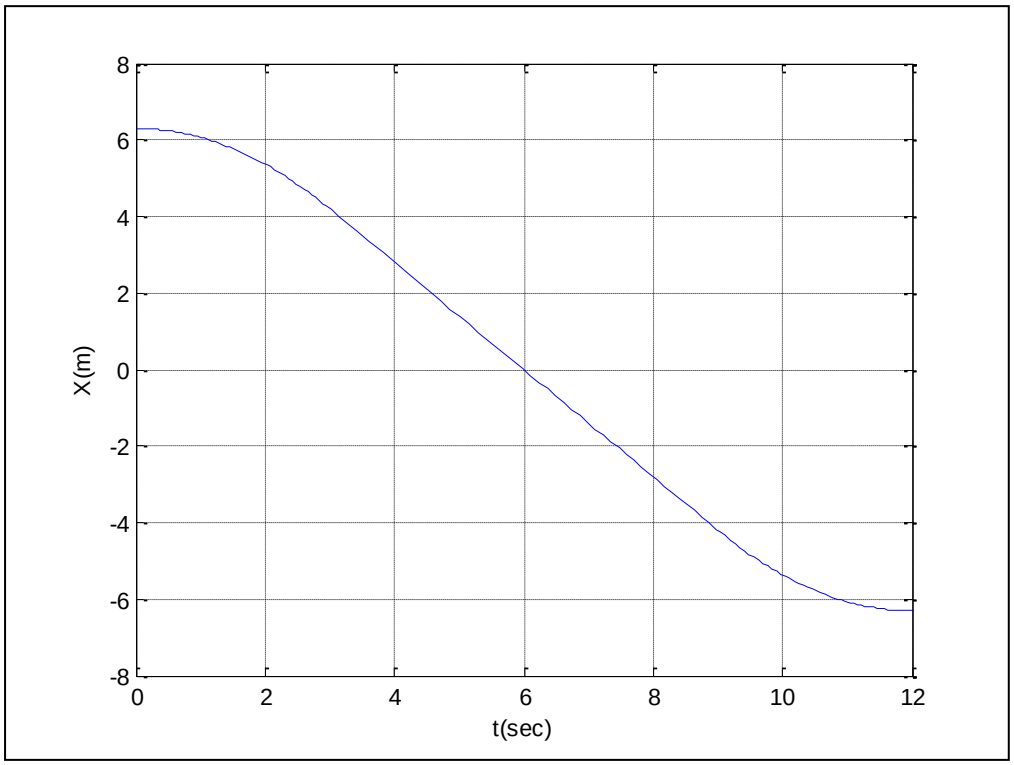
تولید مسیر

به منظور دستیابی به مسیر مربعی ۱۲/۶متر*۱۲/۶ متر از طراحی مسیر با مرزهای سهموی و خط راست در بین آنها استفاده میکنیم. به دلیل اینکه تمایل داریم مسیر خط راست باشد، طراحی را در فضای کارتزین انجام میدهیم. همینطور با توجه به اینکه مسیر مربعی است سرعت در هر کدام از راستای x یا y در انتهای یک ضلعِ مربوط به آن متغیر صفر می‌گردد و برای طراحی نقاط میانی نداریم. به طور خلاصه وقتی در راستای x حرکت میکنیم با توجه به اینکه y بدون حرکت است در انتهای آن ضلع x بدون حرکت شده و yشتاب میگیرد.پس میتوانیم x را بدون نقطه میانی در نظر بگیریم. به منظور طراحی مسیر دو کار میتوانیم انجام دهیم،یکی اینکه شتاب را مشخص در نظر گرفته و آنگاه زمانهای حرکت سهموی و بر روی خط راست را بدست آوریم و دیگری اینکه از روی زمانها شتاب و معادلات حرکت را بیابیم، ما روش دوم را انتخاب می‌کنیم. زمان برای طیِ کل یک ضلع ۱۲ ثانیه و برای T_b=3sec در نظر گرفته شده است. حال از روابط زیر معادلات مورد نظر برای تعقیب مسیر را بدست می‌آوریم.(T_f=12sec)

$$\ddot{x} = \frac{X_{Ef} - X_{E0}}{t_b(t_f - t_b)}$$

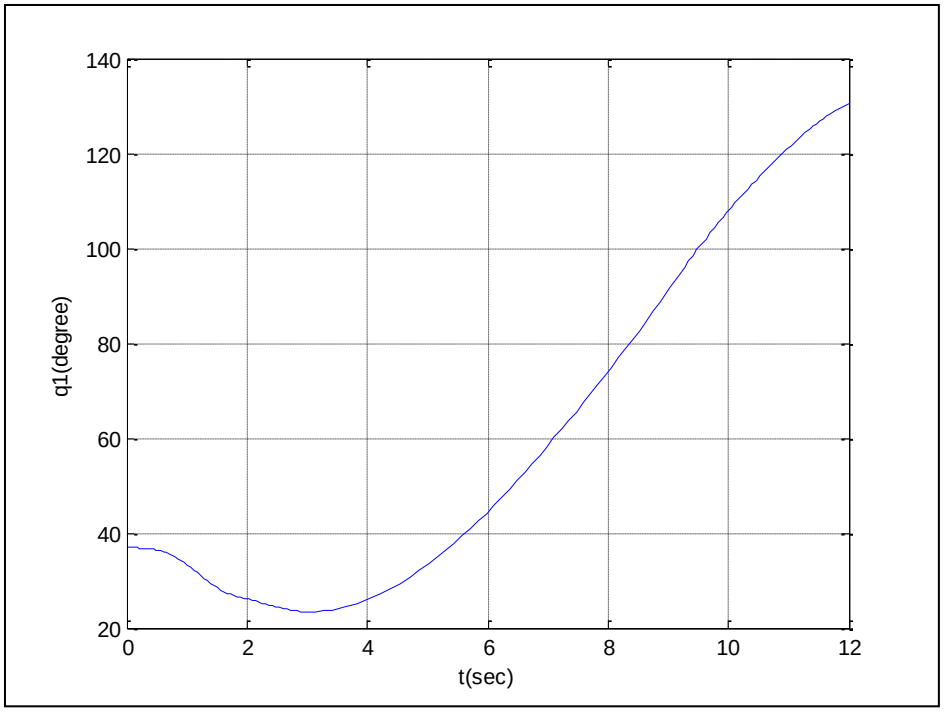
$$x_{E,d}(t) = \begin{cases} X_{E0} + \frac{1}{2}\ddot{x}t^2; 0 \le t \le t_b \\ X_{E0} + \frac{1}{2}\ddot{x}t_b^2 + \ddot{x}t_b(t - t_b); t_b \le t \le t - t_b \\ X_{Ef} - \frac{1}{2}\ddot{x}(t_f - t)^2; t - t_f \le t \le t_f \end{cases}$$

مشابه روابط فوق برای y نیز تکرار میگردد.در استفاده از روابط فوق پیوستگی‌های شتاب و سرعت نیز لحاظ گردیده است.در شکل صفحه بعد یک نمونه از مسیر ایجاد شده آورده شده است.



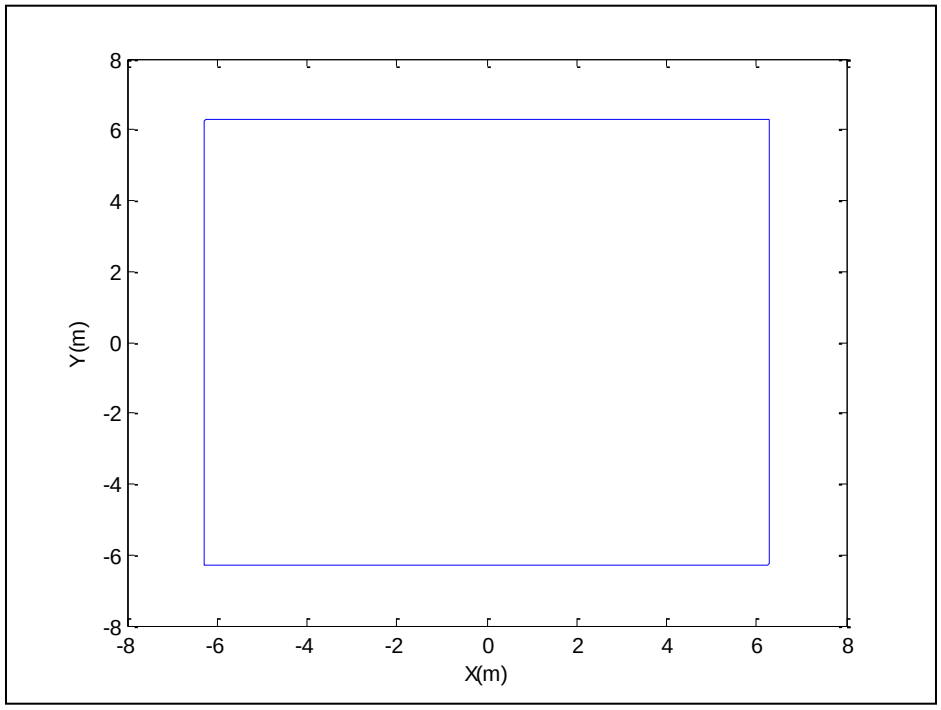
شکل(۳): نمودار مسیر تولید شده برای متغیرX در یکی از ضلعها نسبت به زمان

همینطور تغییرات q1 نیز در شکل زیر نشان داده شده است، که نشان‌دهنده مشتق پذیر بودن آن می باشد.



شکل(۴): نمودار تغییرات زاویه q_1 برای زمان ۱۲ ثانیه در یکی از ضلعها– مربوط به شکل(۱)

مسیر تولید شده در شکل زیر نشان داده شده است.

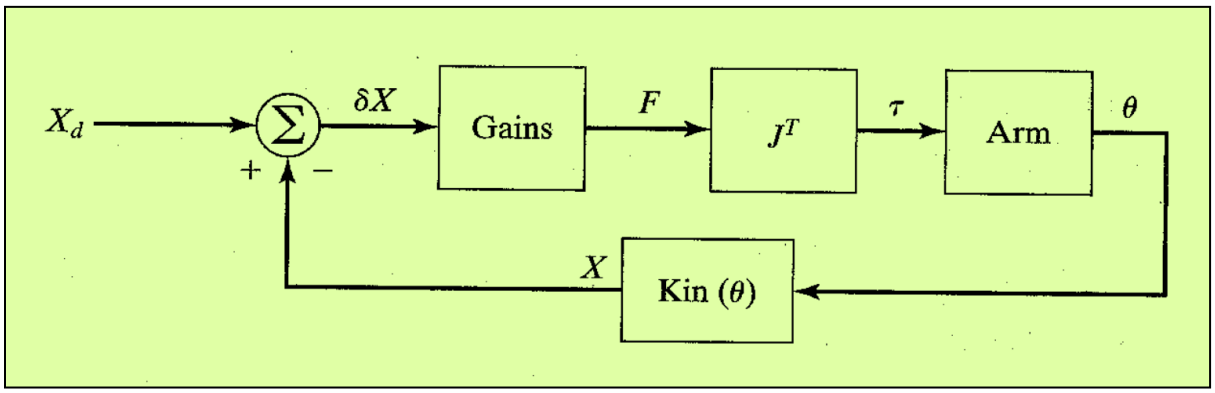


شکل(۵):نمودار مسیر کلی تولید شده برای تعقیب توسط ربات 2R –مربعی به طول و عرض ۱۲/۶ متر

در نهایت با اعمال معادلات فوق و ایجاد مسیر مناسب، **x,y** مطلوب را به سیستم اعمال می‌نماییم.

روش کنترل دینامیک معکوس

روش عمومی این نوع کنترل در شکل زیر آورده شده است ،که به تغییراتی به سیستم مورد نظر ما اعمال گردیده ، در کتاب **Craig** از نام ژاکوبین معکوس برای این روش استفاده شده است.



شکل(۶): نمودار کنترل کلی دینامیک معکوس

معادلات قانون کنترل به صورت ذیل میباشد، که برای میل دادن خطا به سمت صفر از ضرایب K_p,K_d استفاده می‌شود.

$$\mathbf{U} = \mathbf{J}^{\mathrm{T}}(\mathrm{q})\left[\mathbf{K}_{\mathrm{p}}\begin{bmatrix} e_x \\ e_y \end{bmatrix} + \mathbf{K}_{\mathrm{d}}\begin{bmatrix} \dot{e}_x \\ \dot{e}_y \end{bmatrix}\right]$$

$\mathbf{J}(\mathbf{q})$ = Jacobian of direct kinematics

$\mathbf{J}^{T}(\mathbf{q})$ = Jacobian transpose

$\mathbf{K}_{p} = \mathrm{diag}\left[\omega^{2}, \omega^{2}\right]$ = proportional gain

$\mathbf{K}_{d} = \mathrm{diag}\left[2\zeta\omega, 2\zeta\omega\right]$ = derivative gain

e_{x}, e_{y} = errors vector, i.e. $x_{c} - x, y_{c} - y$

$\dot{e}_{x}, \dot{e}_{y}$ = change of input errors vector

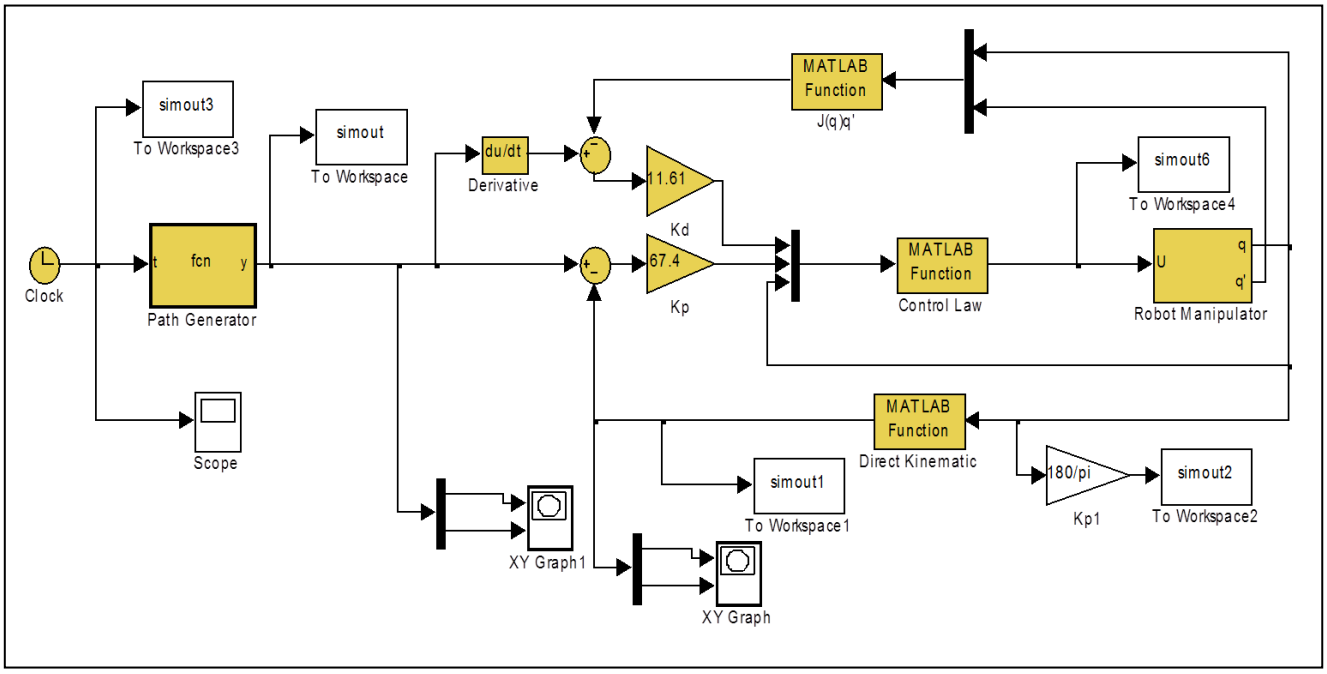
x_{c}, y_{c} = commanded end effector positions

x, y = actual end effector positions

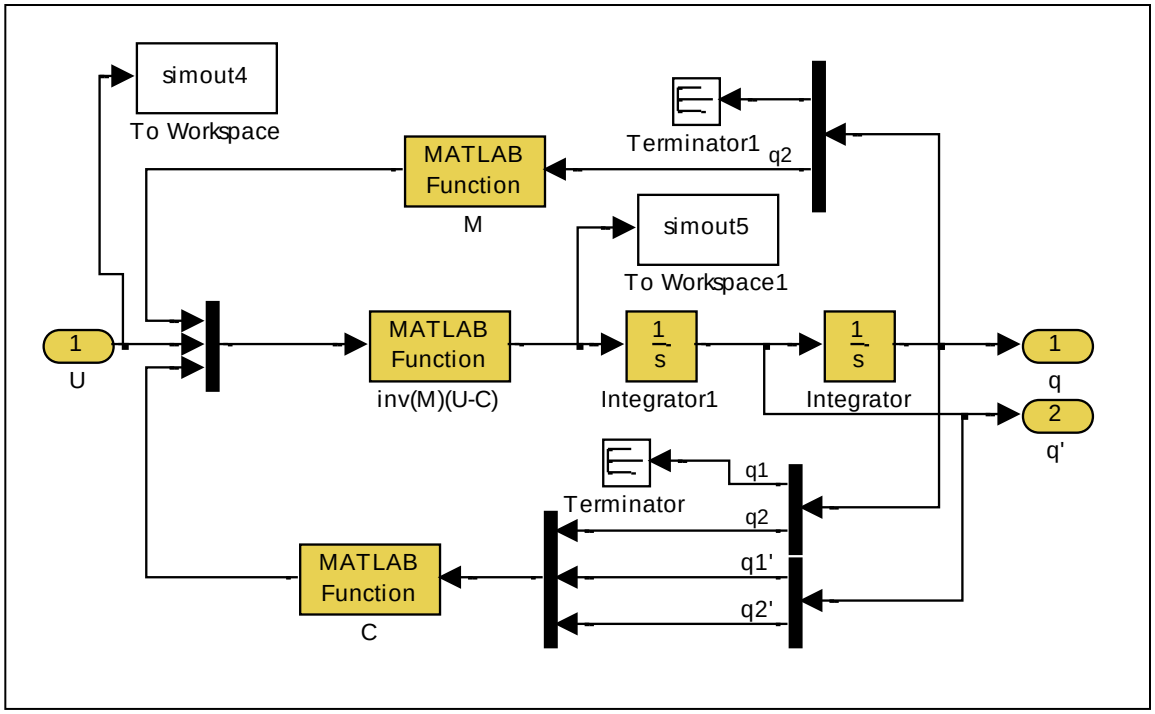
با استفاده از فرمولهای فوق و اهداف کنترلی و صورت مسئله داده شده ضرایب زیر بدست می‌آیند:

$$\mathbf{K}_{p} = \mathrm{diag}\left[67.4, 67.4\right] \text{ and } \mathbf{K}_{d} = \mathrm{diag}\left[11.61, 11.61\right]$$

شکلهای زیر پیاده سازی کنترل کننده فوق و معادلات دینامیکی ربات در محیط SIMULINK را نشان می‌دهد.

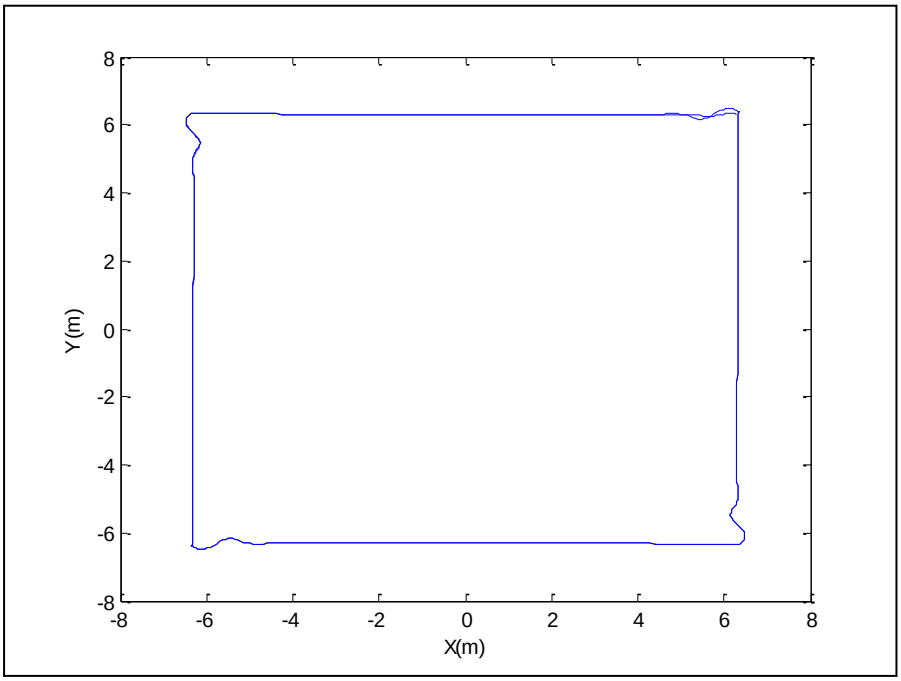


شکل(۷): نمودار شبیه سازی روش دینامیک معکوس با استفاده از تولید مسیر،ربات واقعی و بهره‌ها



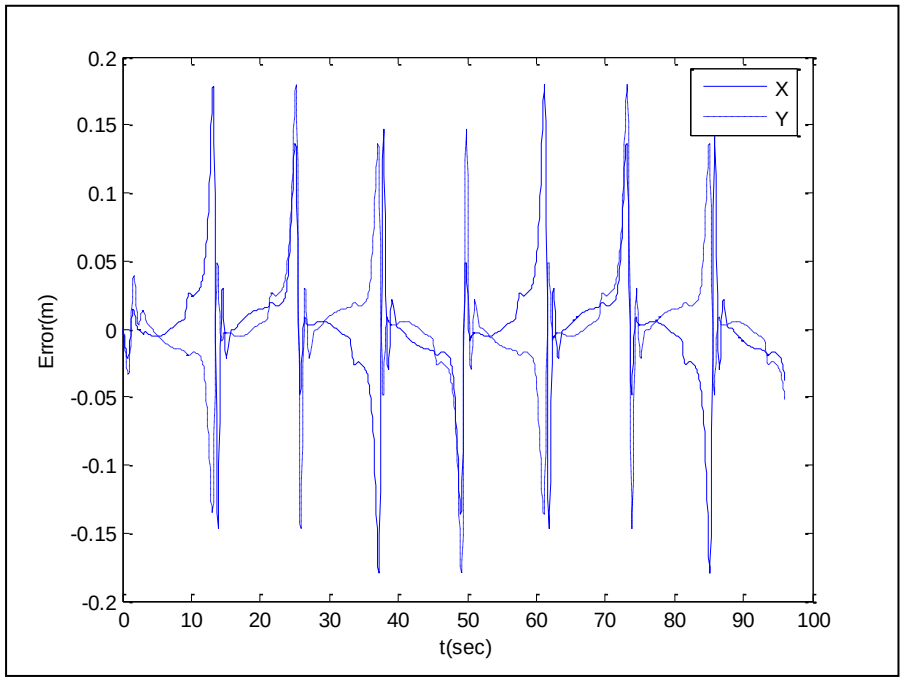
شکل(۸): مدل دینامیکی ربات 2R

نتایج بدست آمده از شبیه‌سازی فوق که مبتنی بر مدل واقعی ربات است در شکلهای زیر آمده است.



شکل(۹):مسیر دنبال شده توسط ربات برای ۴ بار دور زدن – تکرار –

در نمودار ۹ خطای تعقیب مسیر نشان داده شده است. همانطور که دیده میشود حداکثر خطا در حدود ۱۵ سانتیمتر است.



شکل(۱۰): نمودار خطای تعقیب **مسیر** در ۴ بار تکرار برای کنترل دینامیک معکوس

کنترل فازی

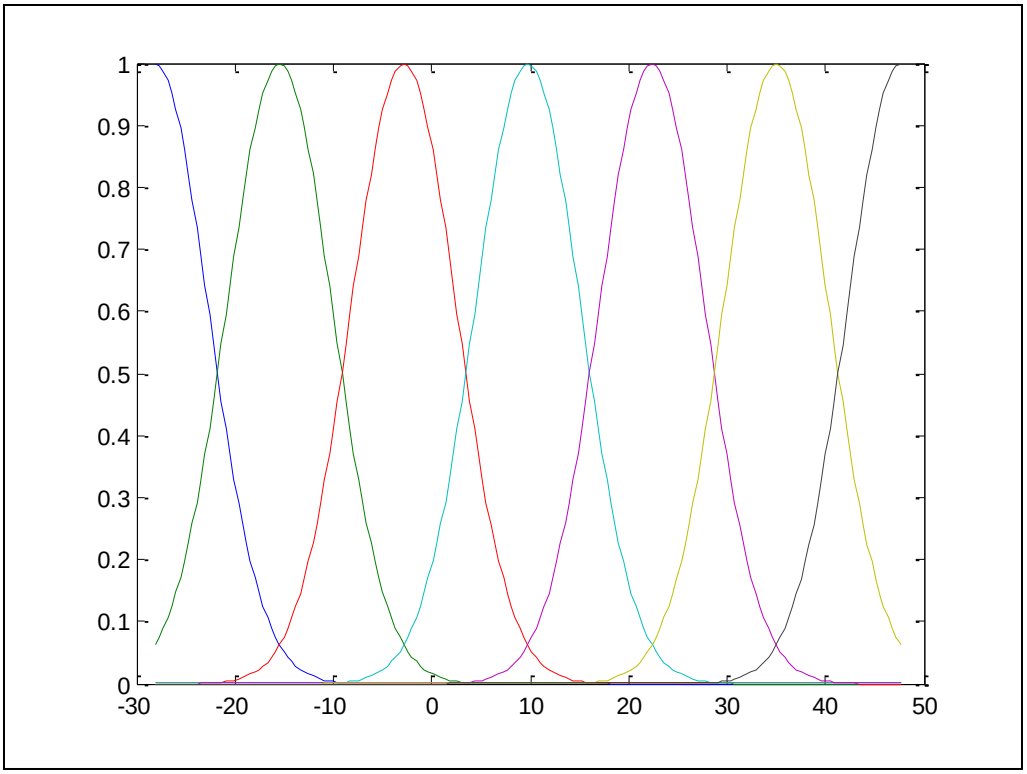
شناسایی دینامیک ربات بوسیله مدل ANFIS و پیاده سازی آن

در روش کنترل سینماتیک معکوس ما از مدل دینامیکی واقعی ربات استفاده کرده و آنرا با معادلات شبیه‌سازی کردیم. در مقاله اصلی از جدول قواعد فازی مربوط به بازوها –به ضمیمه مراجعه شود– برای پیاده سازی استفاده شده است، ولی در اینجا شبکه **ANFIS** جایگزین مدل فازی ارائه شده در مقاله گردیده و نسبت به مقاله نتایج بهتری ارائه شده است. از مدل **ANFIS** برای شناسایی مدل ربات استفاده کرده و این مدل را در **SIMULINK** مورد بررسی قراردادیم.

در واقع با استفاده از ANFIS یک سیستم فازی را به شکل شبکه عصبی نمایش میدهیم تا بتوانیم از قواعد آموزشیِ شبکه عصبی استفاده کنیم. در ابتدا با استفاده از داده‌های ورودی و خروجی یک سیستم استنتاج فازی سوگنو با هفت تابع عضویت گاوسی برای آموزش شبکه ANFIS استخراج گردید و مدل شبکه پس از آموزش به فرم زیر به دست آمد.

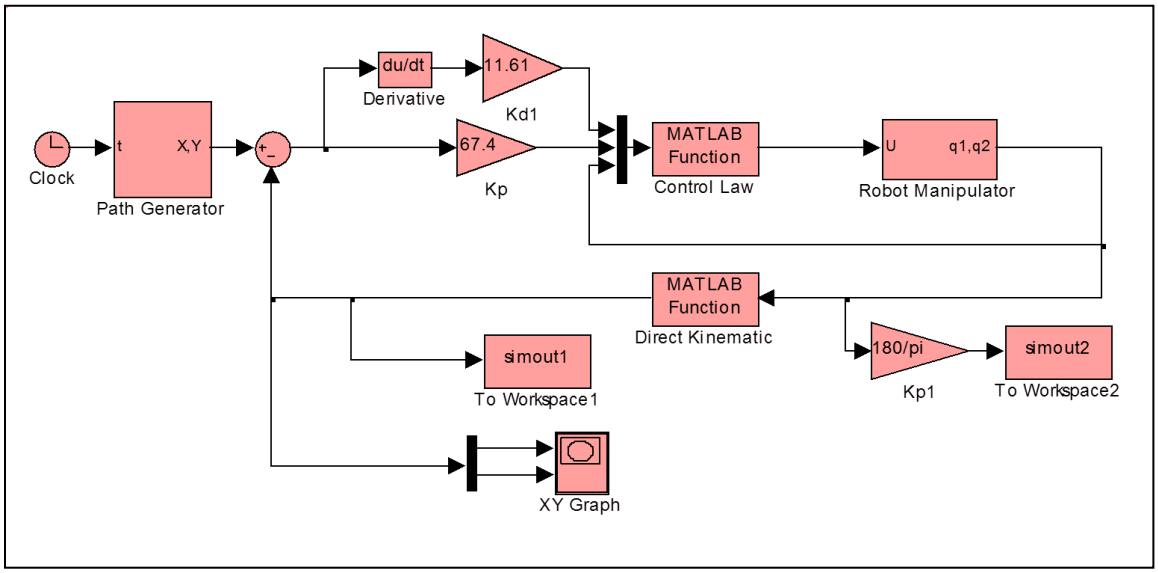
```
  Name              anfis
  Type              sugeno
  Inputs/Outputs    [2 1]
  NumInputMFs       [7 7]
  NumOutputMFs      49
  NumRules          49
  AndMethod         prod
  OrMethod          max
  ImpMethod         prod
. AggMethod         max
. DefuzzMethod      wtaver
. InLabels          input1
.                   input2
. OutLabels         output
. InRange           [-28.08 47.61]
.                   [-6.742 3.9]
. OutRange          [-0.9215 1.337]
```

توابع عضویت هم به فرم ارائه شده در شکل زیر می‌باشند.که با توجه به مدل ربات به دست آمده‌اند. البته داری دو مدل فازی برای هرکدام از خروجیها هستیم که برای اختصار تنها یکی از آنها آورده شده است.

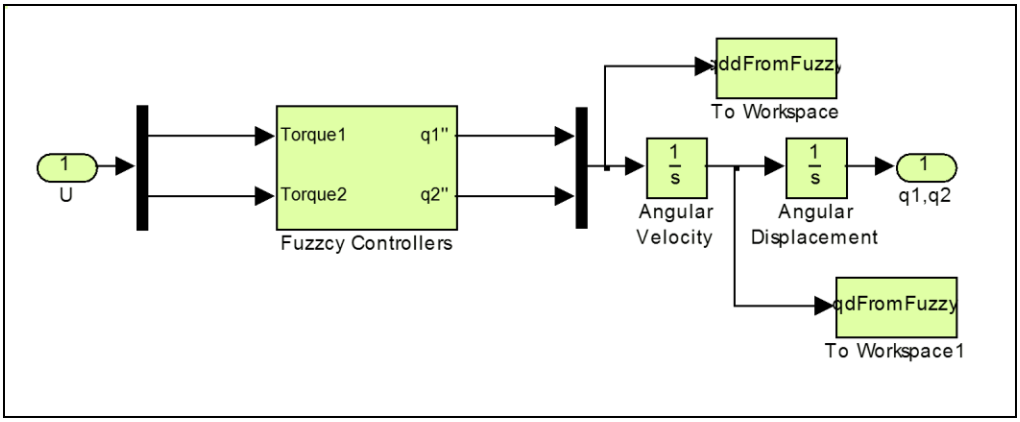


شکل(۱۱): نمونه توابع عضویت گاوسی برای سیستم نروفازی ANFIS برای ورودی X در مدل ربات

بعد از بدست آوردن مدل **ANFIS** مناسب آن را شبیه‌سازی می‌کنیم. که نحوه پیاده سازی در شکل های زیر آمده است.

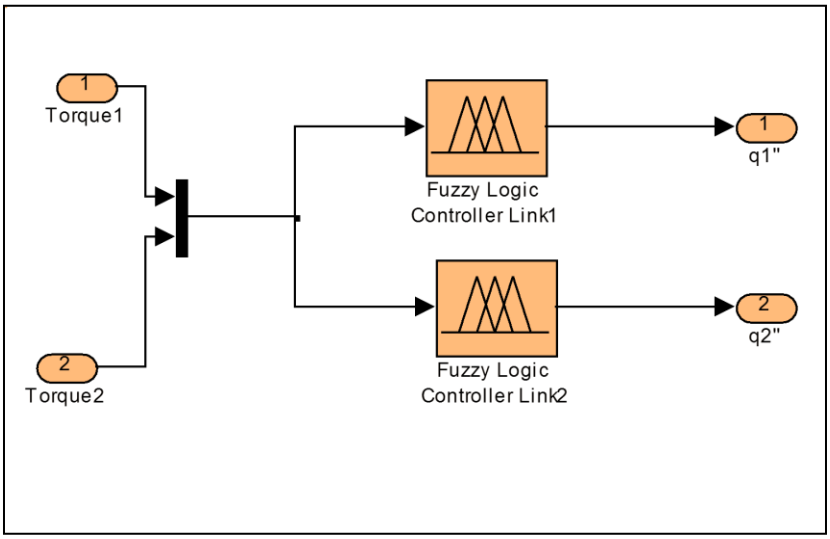


شکل(۱۲):سیستم فازی در محیطSIMULINK همانند شکل(۷) با اندکی تغییر



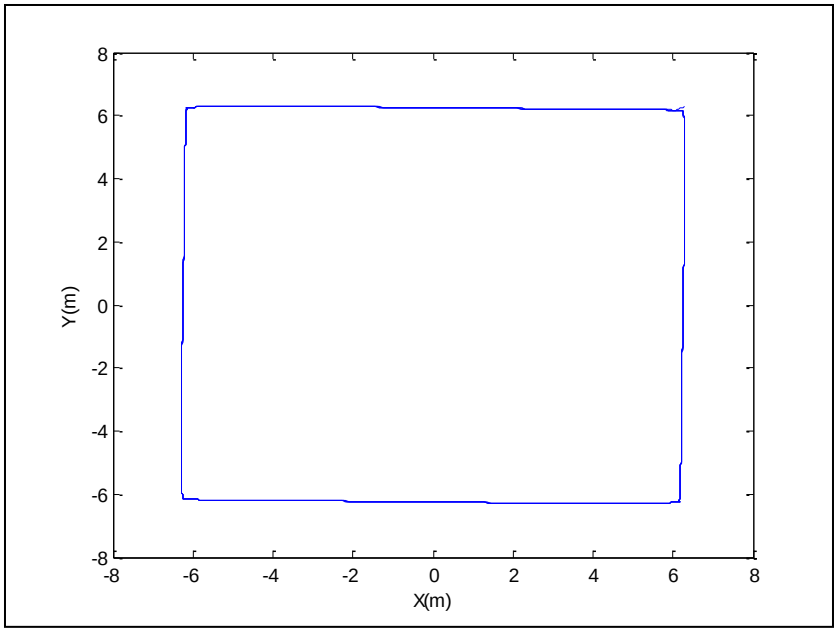
شکل(۱۳): مدل ربات – زیر سیستم Robot Manipulator –

در شکل زیر کنترل کننده‌های بدست آمده از شبکه ANFIS به عنوان مدل سیستم قرار داده شده-اند.



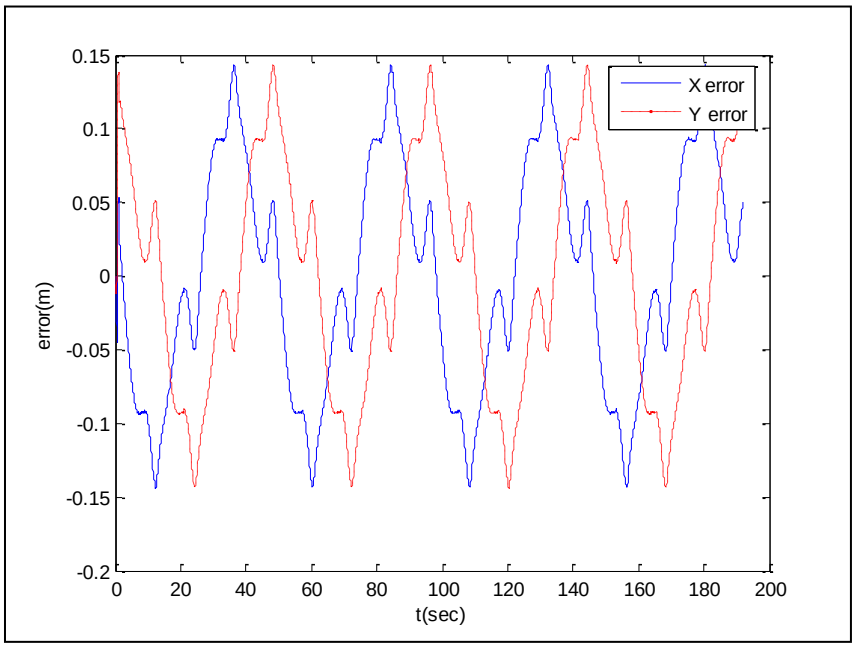
شکل(۱۴): کنترل کننده‌های فازی برای شتاب بازوهای ۱و۲

نتایج شبیه سازی مدل فوق برای دنبال کردن مسیر در شکل ۱۵ نمایش داده شده‌اند. البته مسیر ۴ بار طی شده است، که به علت دقت بالا و کوچک بودن شکل قابل تشخیص نمی‌باشد.



شکل(۱۵):مسیر دنبال شده توسط ربات با کنترل فازی

همانطور که مشاهده می‌شود تعقیب مسیر در مقایسه با کنترل کننده دینامیک معکوس در انتهای ضلعها مطلوبتر است. نمودار خطا برای زمان ۱۹۲ ثانیه -۴ بار دور زدن مسیر مربعی- در شکل زیر برای بررسی بیشتر آورده شده است. با مشاهده این نمودار مشخص می گردد که همان ۱۵ سانتی متر خطا را به نحوی نرمتر داریم.



شکل(۱۶): خطای تعقیب مسیر برای کنترل فازی

یادگیری تکراری

هدف کنترل تکراری آموزش ربات برای انجام حرکتها تکراری است. به نحوی که بعد از هر تکرار کارآیی بهبود یافته و خطا کاهش یابد. در یادگیری تکراری ،ما شرایط اولیه را بعد از هر تکرار صفر نمی‌کنیم ، بلکه قانون کنترل را با مقادیر قبلی کنترلی به روز می‌کنیم. در برخی از مقالات نحوه بروز رسانی مانند معادلات زیر با یک بهره تناسبی یا مشتقی و یک ضریب یادگیری K_L می‌باشد و دارای الگوریتم ذیل می‌باشند.

$$\mathbf{U}_{k+1} = \mathbf{U}_k + \mathbf{K}_{\mathrm{L}} \begin{bmatrix} e_x \\ e_y \end{bmatrix}^k$$

$$\mathbf{U}_{k+1} = \mathbf{U}_k + \mathbf{K}_{\mathrm{L}} \begin{bmatrix} \dot{e}_x \\ \dot{e}_y \end{bmatrix}^{k+1}$$

$$k = 1,2,3,\ldots\ldots\text{n iterations}$$

با بهره‌گیری از فرمولهای بالا میتوان روش یادگیری تکراری را برای قانون کنترل، با جایگزینی C_L به عنوان ضریب یادگیری، به فرم زیر بسط داد

$$\mathbf{U}_{\mathrm{L}}^{i} = \mathbf{U}_{\mathrm{L}}^{i-1} + \mathbf{C}_{\mathrm{L}} \begin{bmatrix} e_x \\ e_y \end{bmatrix}^{i}$$

$$i = 1,2,3,\ldots\ldots\ldots\text{n iterations}$$

در اینجا به روز رسانی با فیدبکهای ساده‌ای از خطای سرعت و موقعیت انجام می‌گیرد.

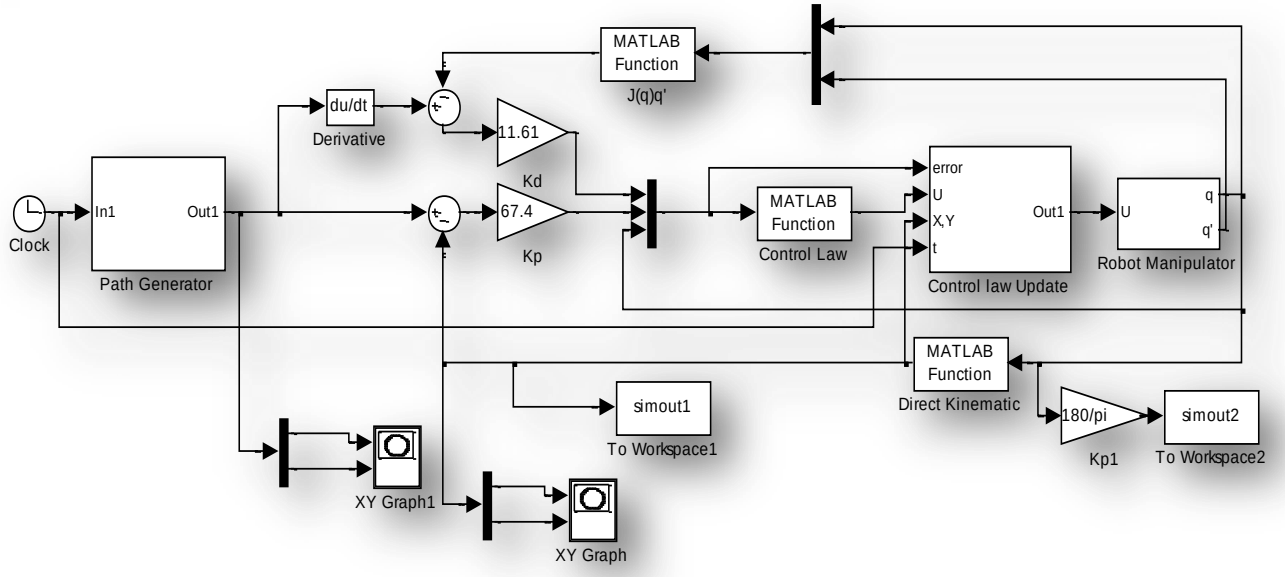
$$\mathbf{U}_{k+1} = \mathbf{U}_k + \mathbf{J}^{\mathrm{T}}(\mathbf{q}) \left(\mathbf{K}_{\mathrm{p}} \begin{bmatrix} e_x \\ e_y \end{bmatrix} + \mathbf{K}_{\mathrm{d}} \begin{bmatrix} \dot{e}_x \\ \dot{e}_y \end{bmatrix} \dot{\mathbf{q}} \right)^{k+1}$$

$$k = 1,2,3,\ldots\ldots\text{n iterations}$$

برای به دست آوردن U_K در معادلات فوق از یک شبکه ی آموزش داده شده‌ی **ANFIS** استفاده کردیم و با اعمال آن به روش کنترلی فوق توانستیم به نتایج خوبی برسیم .

پیاده سازی روش یادگیری تکراری بر روی روش کنترل دینامیک وارون

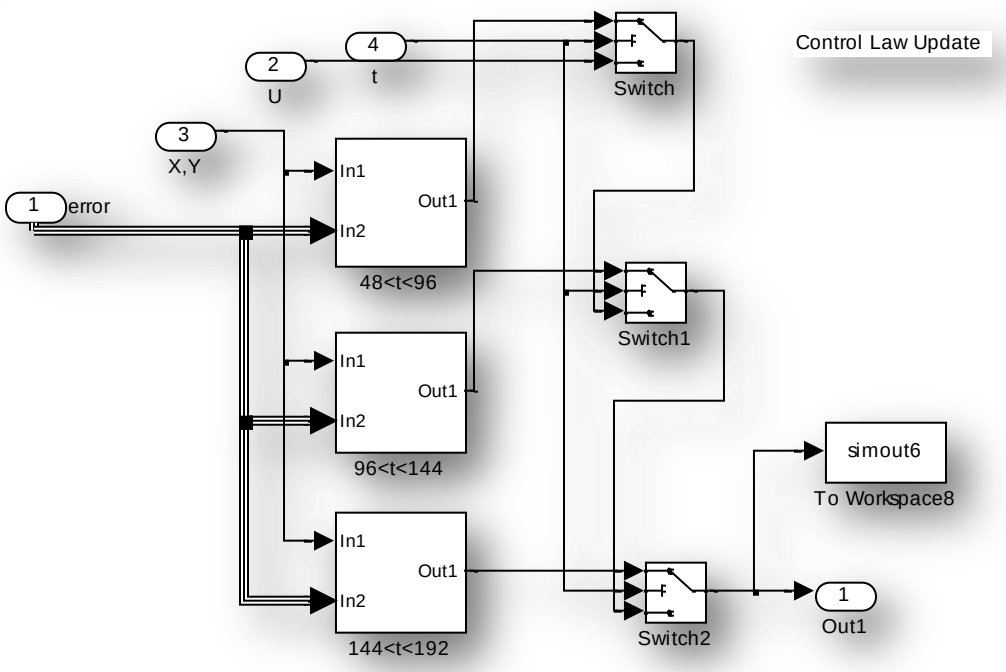
حال روش یادگیری فوق را بر روی سیستم ربات دو درجه آزادی با ملاحظات ذکر شده در صفحه‌ی قبل اعمال می‌نمائیم.در شکلهای زیر نحوه‌ی پیاده‌سازی در محیط **SIMULINK** نشان داده شده است.



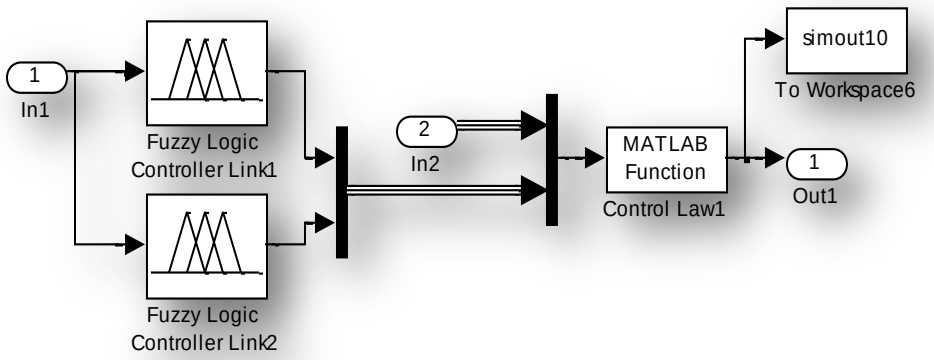
شکل(۱۷): نمودار کنترل با قوانین کنترل بروز شونده

نمودار فوق نشان دهنده فرم قرار گیریِ قانون یادگیری است که در مسیر پیش‌خور قرار گرفته. شکل ۱۸ نمایانگر قانون سوئیچینگ مربوط به تکرارهای متفاوت است. که با توجه به زمان از هرکدام از زیر سیستمهای نشان داده شده استفاده مینماید. همینطور در شکل ۱۹ نحوه تولید و استفاده از داده‌های مراحل قبل نشان‌داده شده که میتوان با استفاده از یک برنامه ساده آنرا تولید کرد ولی ما به صورت

قطع خط آنها را بدست آورده و با استفاده از مکان قرار گیری ربات داده‌های مربوط به قانون کنترل در تکرار قبلی را با استفاده از شبکه‌ی فازی بدست آوردیم.

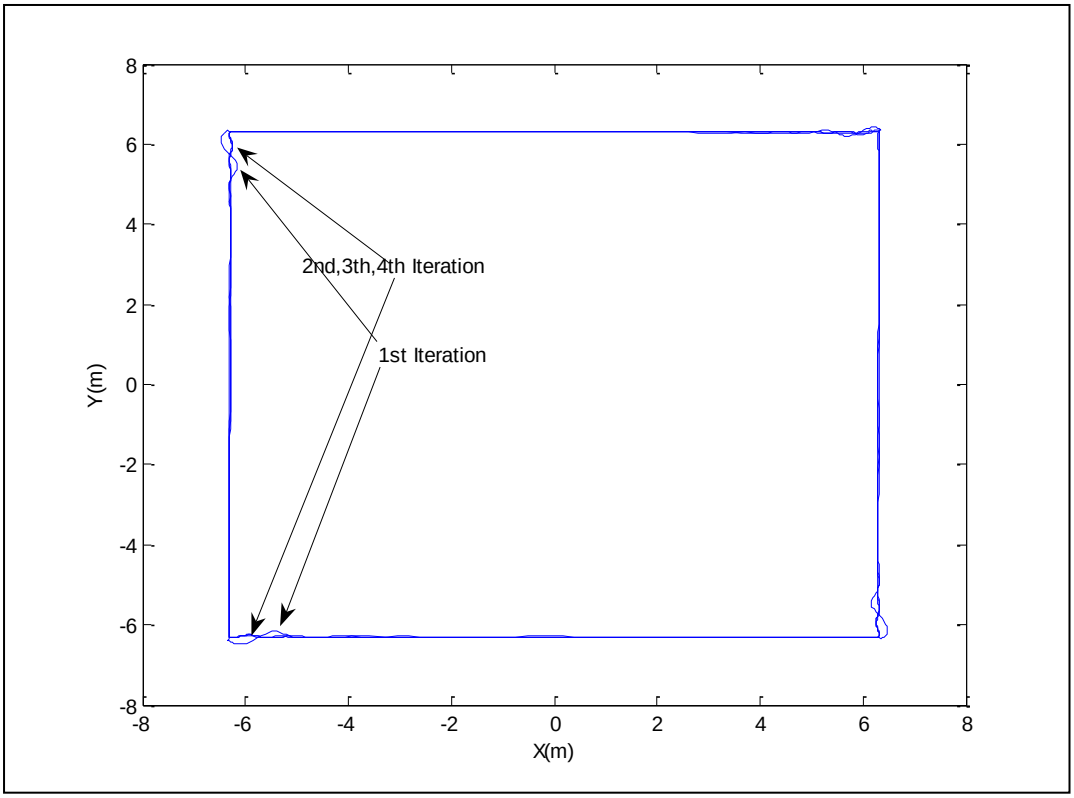


شکل(۱۸): نحوه سوئیچینگ برای قوانین کنترل متفاوت بر حسب زمانt

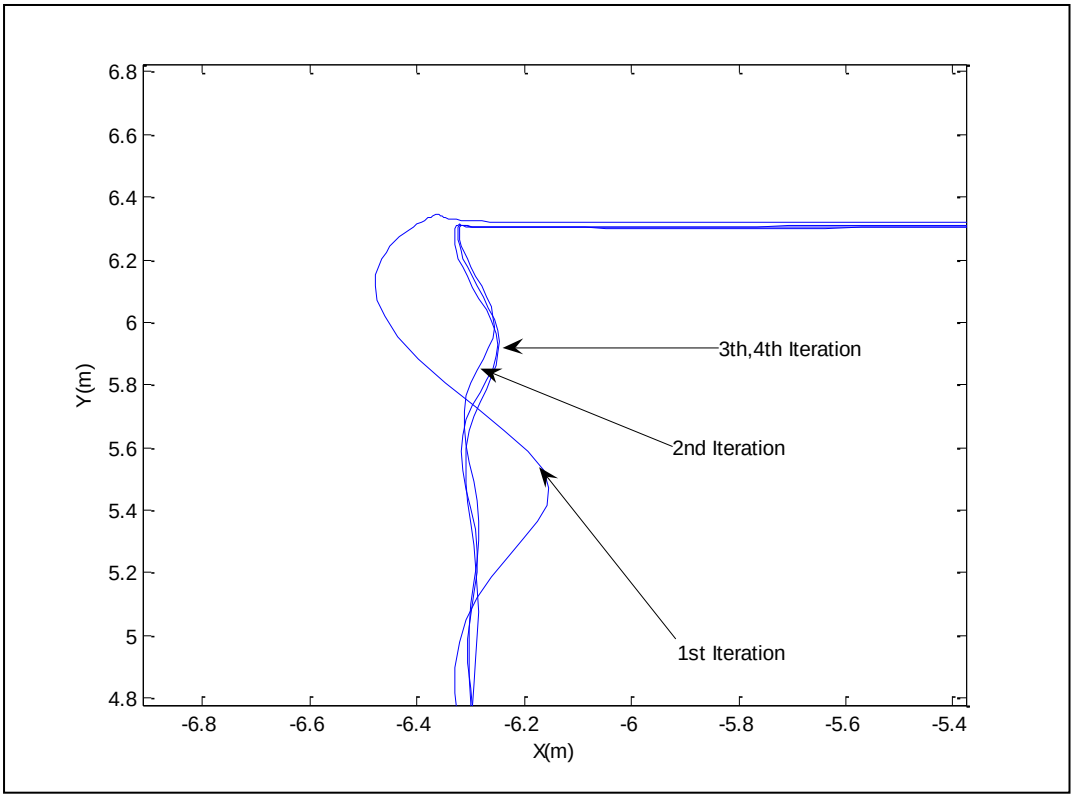


شکل(۱۹): تولید قانون کنترل k+1 ام بر مبنای اطلاعات تکرار k ام

در شکل (۲۱) نتایج حاصل از تعقیب سیستم برای ۴ دور –تکرار- ارائه گردیده است ، مشاهده می-گردد که استفاده از روش یادگیری تکراری باعث بهبود پاسخ و کم شدن خطا به مقدار قابل ملاحظه برای سیستم دینامیکیِ واقعی گردیده است. شکل (۲۲) نشان دهنده‌ی کم شدن خطا در تکرارهای ۲،۳ و ۴ به نسبت تکرار اول به میزان قابل ملاحظه است.



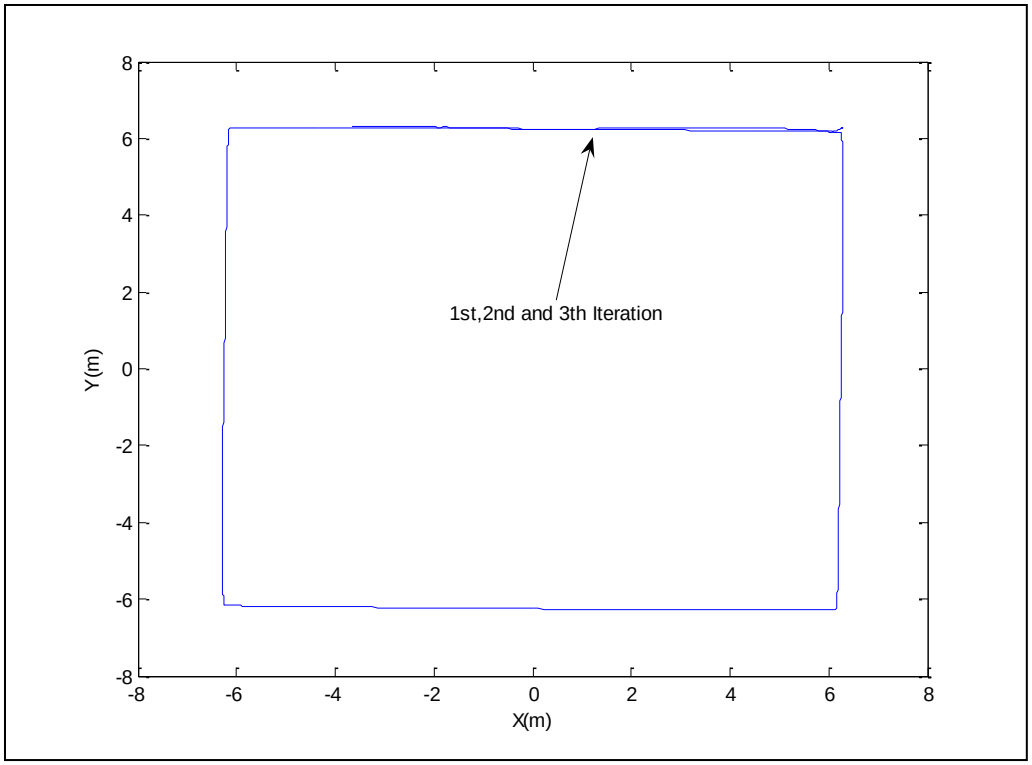
شکل(۲۱): نمودار تعقیب مسیر برای ۴ تکرار



شکل(۲۲): نمودار تعقیب مسیر در گوشه‌ی x=-6.3m,y=6.3m

پیاده سازی روش یادگیری تکراری بر روی مدل فازی

حال در این مرحله همان سیستم یادگیری، بر روی سیستم کنترل فازی اجرا می‌گردد و باعث می-گردد که خطا بهبود یابد ،هرچند به دلیل دقت خوب سیستم در شکل (۱۵) خیلی، تفاوت آشکاری در بهبود کارآیی بوجود نمیاید . نحوه پیاده سازی مشابه شکل(۱۷)است که به جای مدل دینامیکی واقعی سیستم، از سیستمِ فازی استفاده می‌کنیم.



همانطور که مشاهده می‌شود در نمودار فوق دارای بهبود خیلی واضحی در پاسخ نیستیم، ونمیتوان گفت که کدام روش بهتر است.

نتیجه گیری

در این پروژه ابتدا به بررسی روش کنترل دینامیک معکوس برای یک ربات دو درجه آزادی که یک مسیر مربعیِ از پیش تعیین شده را دنبال می‌کرد پرداختیم.همانطور که مشاهده شد در انتهای هر ضلع مسیر دارای خطا و در واقع لرزش بودیم که مشخصه نامطلوبی در بازوهای مکانیکی است. آنگاه سعی کردیم با تغییر نحوه کنترل، از کنترل کلاسیک به فازی و توسعه آن به مدل نروفازیِ **ANFIS** این لرزش را بهبود ببخشیم، که موفق به انجام اینکار به قیمت از دست دادن دقت در نقاط میانیِ مسیر شدیم. در نهایت با استفاده از یک مدل یادگیری، به کمک داده‌های کنترلیِ تکرارهای قبل- با توجه به کاربرد بازوها در محیطهای دارای تکرار فعالیت- توانستیم قوانین کنترل خود را بروز رسانی کنیم. با اعمال این شیوه در کنترل کلاسیک به هدف کاهش لرزش با همان دقت قبلی دست یافتیم، ولی در کنترل فازی به نتیجه‌ای مطابق با آنچه پیش بینی میکردیم نرسیدیم و دقت تنها اندکی بهبود یافت.

منابع

[1] Abdel Badie Sharkawy,” **A computationally efficient fuzzy control scheme for a class of MIMO systems**,” Alexandria Engineering Journal (2013) 52, 583–594,

[2] A.Green, J. Z. Sasiadek," **INVERSE DYNAMICS AND FUZZY REPETITIVE LEARNING FLEXIBLE ROBOT CONTROL**", 15th Triennial World Congress, Barcelona, Spain,2002.

[3] John J.Craig, "**INTRODUCTION TO ROBOTICS**", 3th edition, Prentice Hall.