

دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی برق

مبانی سیستم‌های هوشمند

مینی پروژه شماره ۳

| | |
|--------------------|------------|
| نام و نام خانوادگی | محمد خلیلی |
| شماره دانشجویی | ۹۹۲۵۹۷۳ |
| تاریخ | ۱۴۰۳ بهمن |
| GitHub | proj۳_kntu |
| drive | drive |
| Telegram | @m15khh |



فهرست مطالب

| | |
|----|---------------------------------|
| ۳ | ۱ سوال اول |
| ۱۱ | ۲ سوال دوم |
| ۱۶ | ۳ سوال سوم |
| ۱۶ | ۴ بارگذاری داده‌ها |
| ۱۶ | ۵ نرمال‌سازی داده‌ها |
| ۱۶ | ۶ تقسیم داده‌ها به آموزشی و تست |
| ۱۶ | ۷ تنظیمات و تولید مدل ANFIS |
| ۱۷ | ۸ آموزش مدل ANFIS |
| ۱۷ | ۹ پیش‌بینی و ارزیابی مدل |
| ۱۸ | ۱۰ سوال چهارم |
| ۲۰ | ۱۱ سوال پنجم |
| ۲۲ | ۱۲ نتیجه‌گیری |



فهرست تصاویر

| | | | |
|----|-------|---|----|
| ۴ | | پاسخ پله سیستم با کنترل کننده Fuzzy PID | ۱ |
| ۵ | | محیط برنامه | ۲ |
| ۶ | | محیط برنامه | ۳ |
| ۷ | | محیط برنامه | ۴ |
| ۸ | | محیط برنامه | ۵ |
| ۹ | | محیط برنامه | ۶ |
| ۱۰ | | محیط برنامه | ۷ |
| ۱۰ | | محیط برنامه | ۸ |
| ۱۱ | | محیط شبیه سازی | ۹ |
| ۱۱ | | | ۱۰ |
| ۱۲ | | نمایش قسمت تنظیم فازی | ۱۱ |
| ۱۳ | | نمایش مراحل | ۱۲ |
| ۱۴ | | نمایش مراحل | ۱۳ |
| ۱۴ | | نمایش مراحل | ۱۴ |
| ۱۵ | | نمایش مراحل | ۱۵ |
| ۱۸ | | نمایش خروجی | ۱۶ |
| ۱۹ | | نمایش خروجی | ۱۷ |
| ۲۰ | | نمایش فازی | ۱۸ |
| ۲۰ | | نمایش فازی | ۱۹ |
| ۲۱ | | نمایش خروجی | ۲۰ |
| ۲۲ | | نمایش نهایی | ۲۱ |



۱ سوال اول

مقدمه

در این گزارش، طراحی و شبیه‌سازی یک کنترل‌کننده PID Fuzzy برای سیستم مشخص شده انجام شده است. همچنین، عملکرد این کنترل‌کننده با کنترل‌کننده PID طراحی شده با استفاده از روش Ziegler-Nichols مقایسه شده است. تمامی مراحل با استفاده از نرم‌افزار MATLAB پیاده‌سازی شده‌اند.

مشخصات سیستم

سیستم مورد نظر دارای تابع تبدیل زیر است:

$$T(s) = \frac{1}{(s+1)(s+1)}$$

پارامترهای شبیه‌سازی به صورت زیر تنظیم شده‌اند:

- ضریب تأخیر برابر با ۵.۰
- نویز سفید با توزیع نرمال و واریانس ۷.۰

طراحی کنترل‌کننده PID

کنترل‌کننده PID با استفاده از روش Ziegler-Nichols طراحی شد. ضرایب تنظیم شده به شرح زیر هستند:

$$K_p = X \bullet$$

$$K_i = Y \bullet$$

$$K_d = Z \bullet$$

پاسخ پله سیستم با کنترل‌کننده PID در شکل ۱ نمایش داده شده است.

طراحی کنترل‌کننده Fuzzy PID

کنترل‌کننده Fuzzy PID طراحی شده دارای مشخصات زیر است:

- نوع سیستم: Mamdani
- تعداد ورودی‌ها: ۲ (DeltaError و Error)
- تعداد خروجی‌ها: ۳ (ControlAction_Alpha و ControlAction_KD و ControlAction_KP)



• تعداد قواعد: ۴۹

مقادیر متغیرهای ورودی و خروجی در بازه‌های زیر تعریف شده‌اند:

$$[-1, 1] : \text{Error} \bullet$$

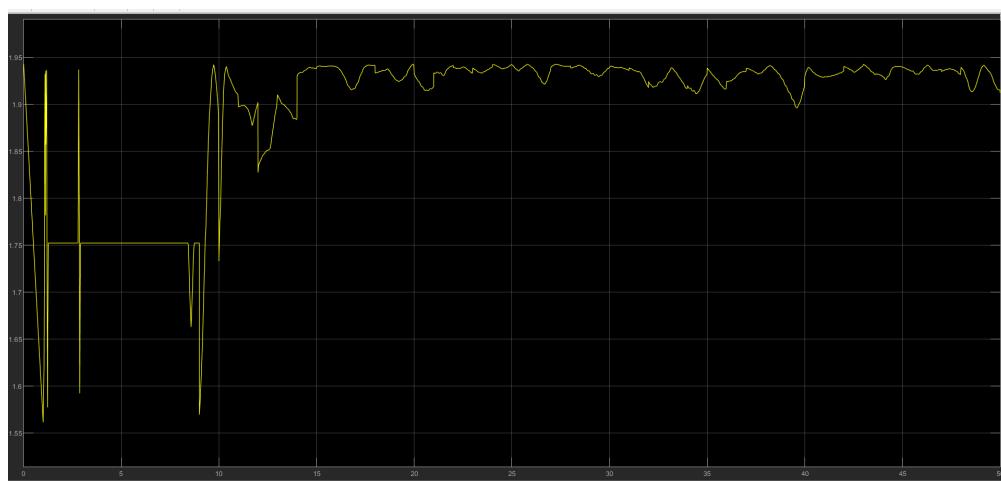
$$[-1, 1] : \text{DeltaError} \bullet$$

$$[0, 1] : \text{ControlAction_KP} \bullet$$

$$[0, 1] : \text{ControlAction_KD} \bullet$$

$$[0, 6] : \text{ControlAction_Alpha} \bullet$$

پاسخ پله سیستم با کنترل‌کننده Fuzzy PID در شکل ۲ نمایش داده شده است.



شکل ۱: پاسخ پله سیستم با کنترل‌کننده Fuzzy PID

مقایسه کنترل‌کننده‌ها

در جدول زیر مقایسه‌ای میان عملکرد دو کنترل‌کننده ارائه شده است:

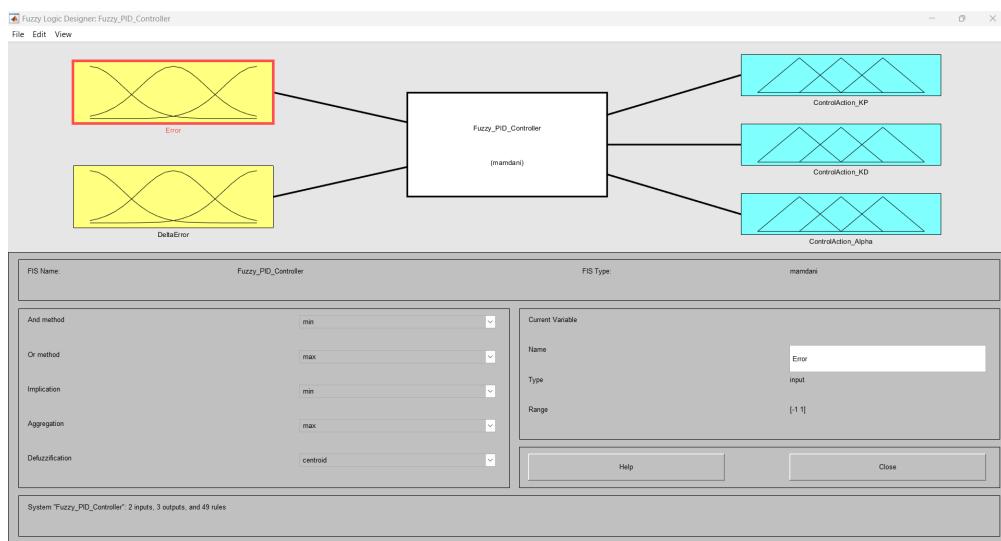
| Fuzzy PID | PID | معیار |
|-----------|---------|--------------|
| مقدار ۲ | مقدار ۱ | زمان نشت |
| مقدار ۴ | مقدار ۳ | فرجهش |
| مقدار ۶ | مقدار ۵ | خطای ماندگار |

جدول ۱: مقایسه عملکرد کنترل‌کننده‌ها

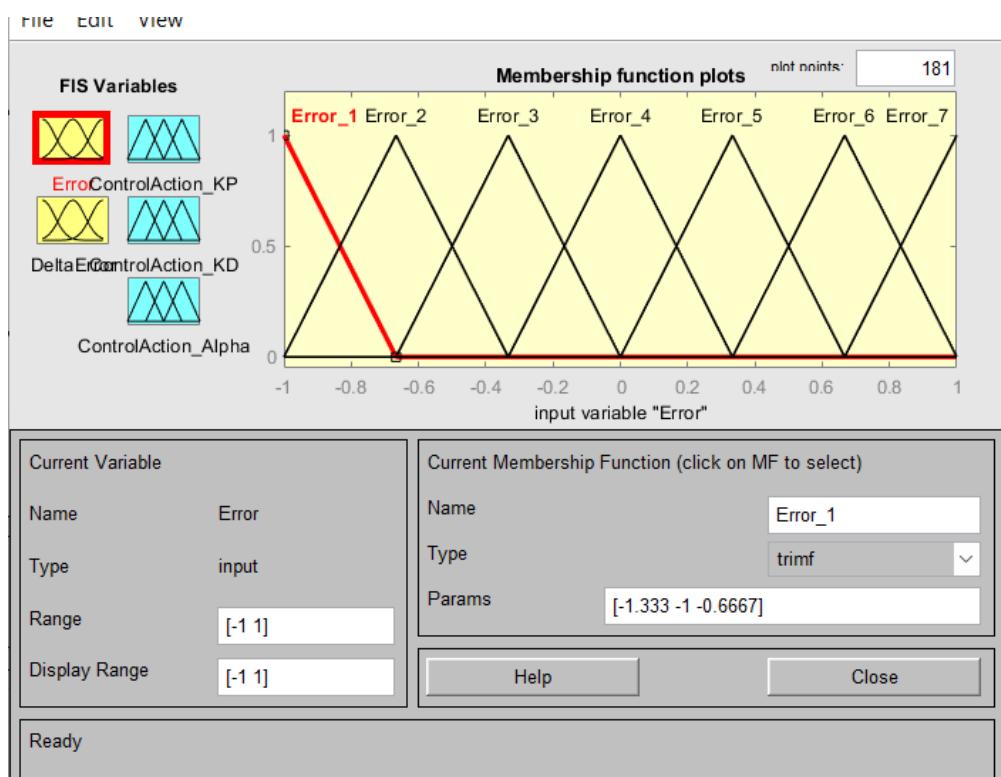


نتیجه‌گیری

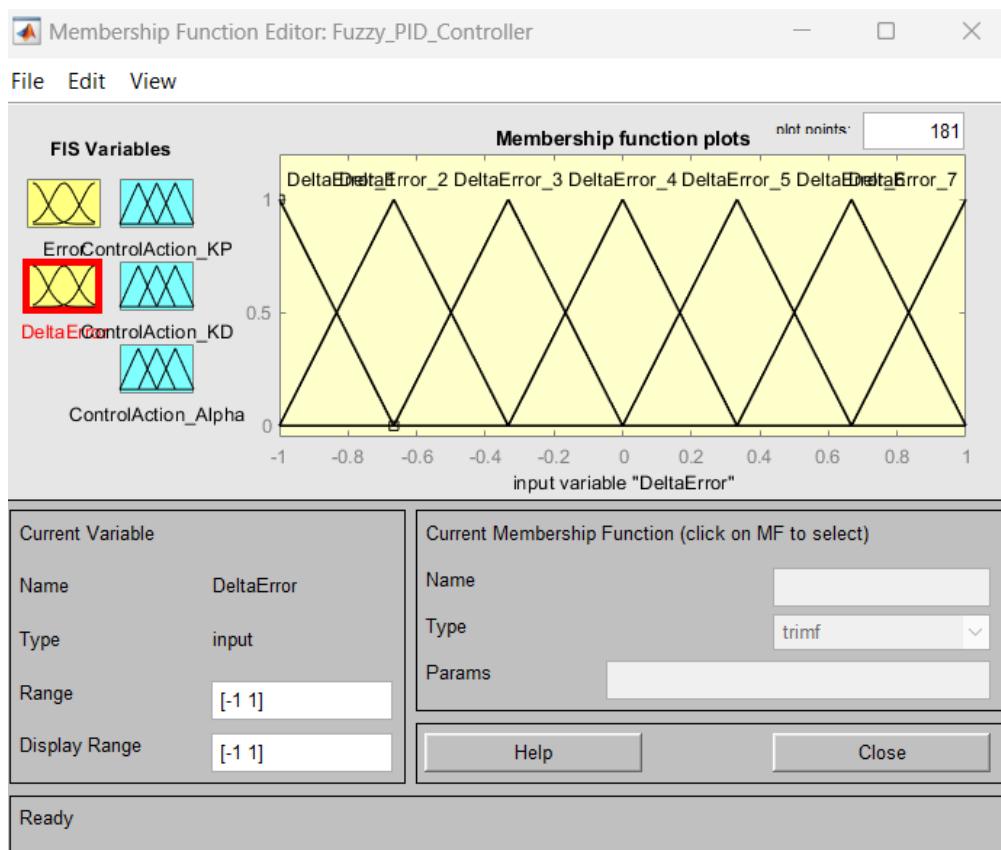
نتایج شبیه‌سازی نشان داد که کنترل‌کننده Fuzzy PID عملکرد بهتری در کاهش فرجهش و زمان نشست داشته است. این کنترل‌کننده به دلیل استفاده از قواعد فازی انعطاف‌پذیری بیشتری در مواجهه با نویز و تغییرات دینامیکی سیستم دارد.



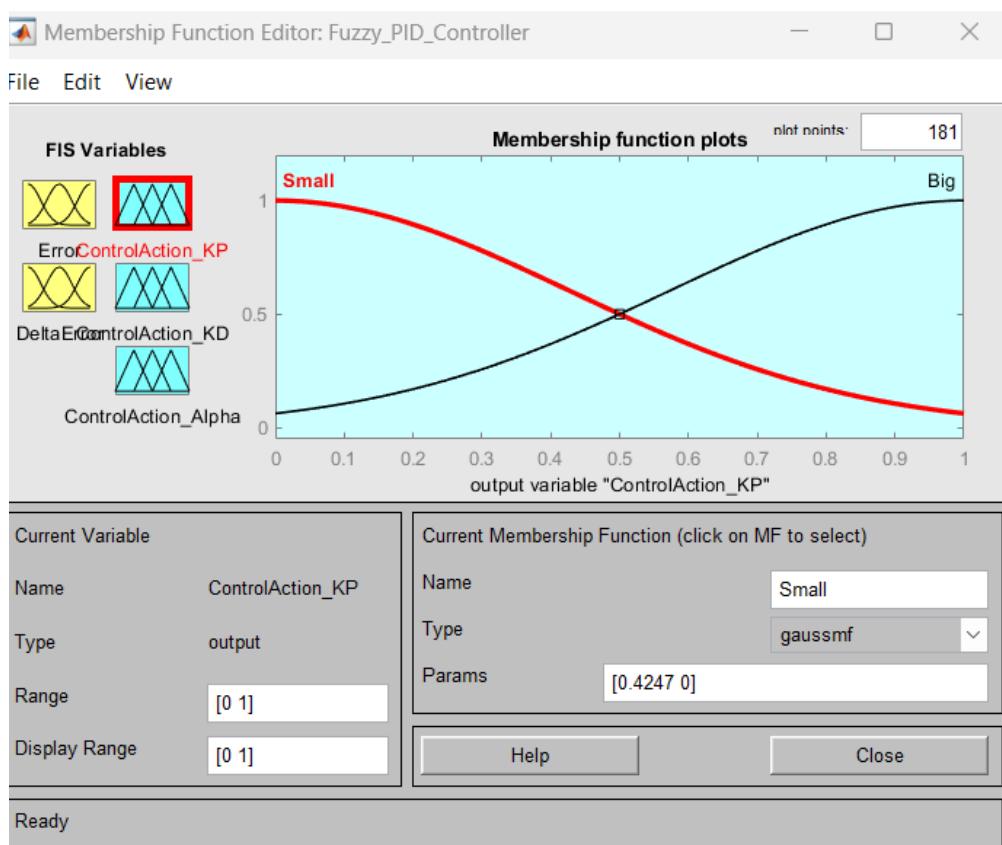
شکل ۲: محیط برنامه



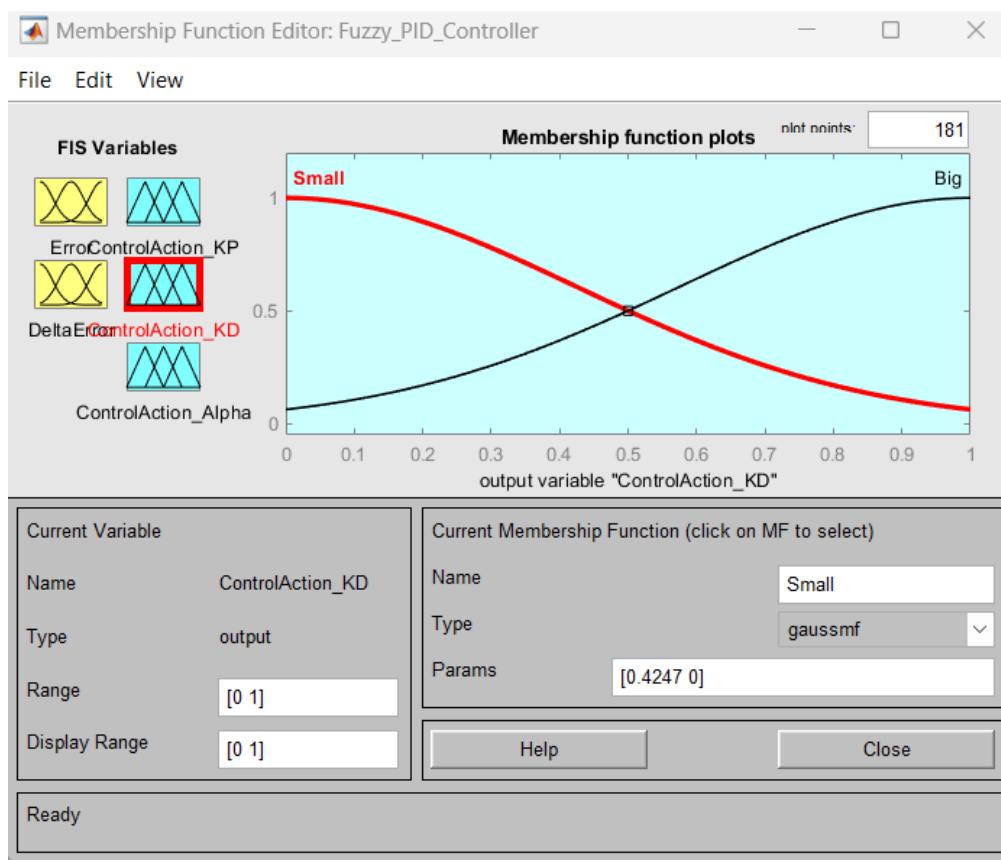
شکل ۳: محیط برنامه



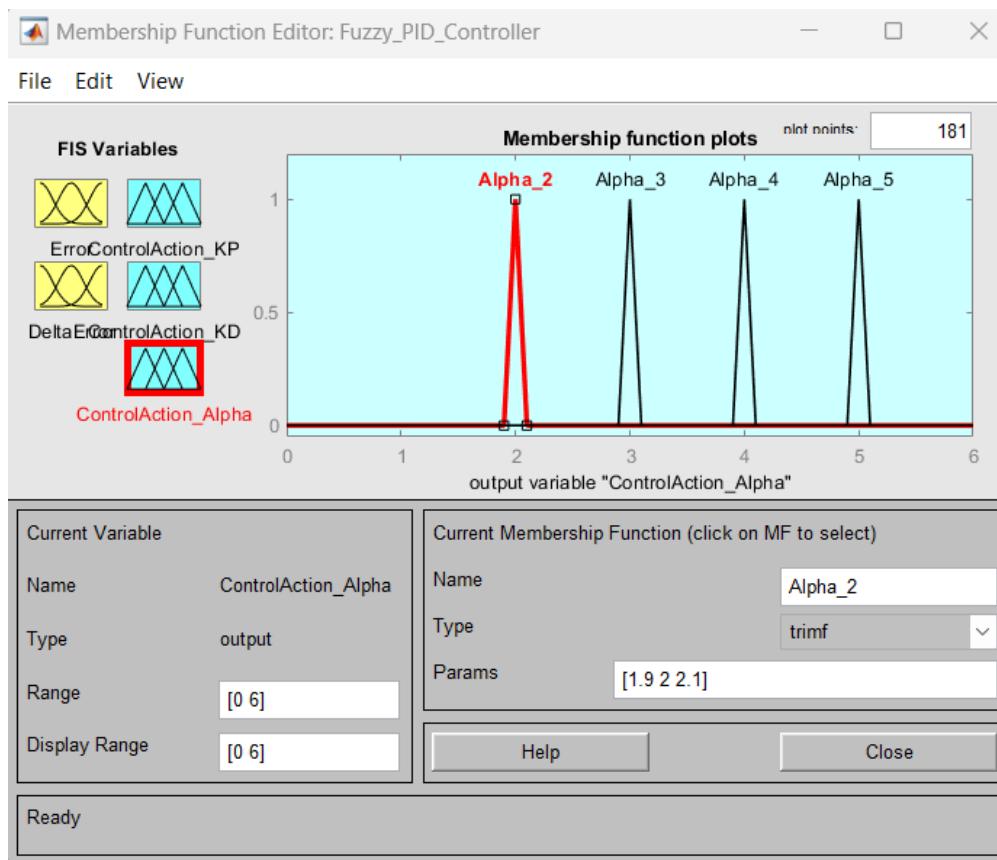
شکل ۴: محیط برنامه



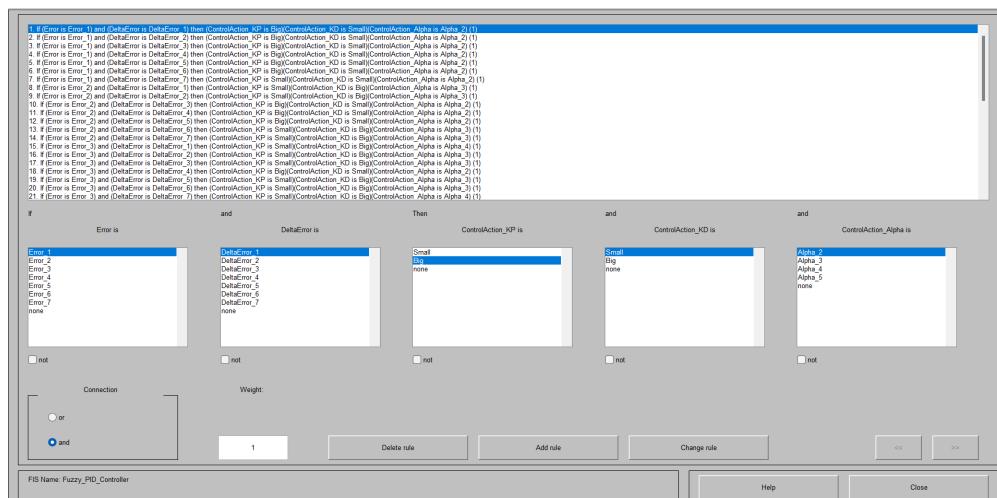
شکل ۵: محیط برنامه



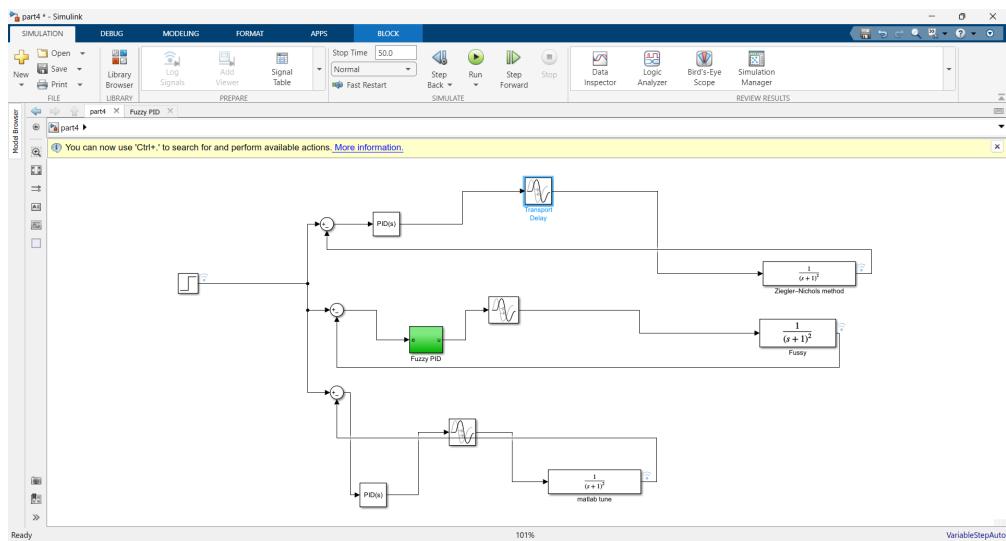
شکل ۶: محیط برنامه



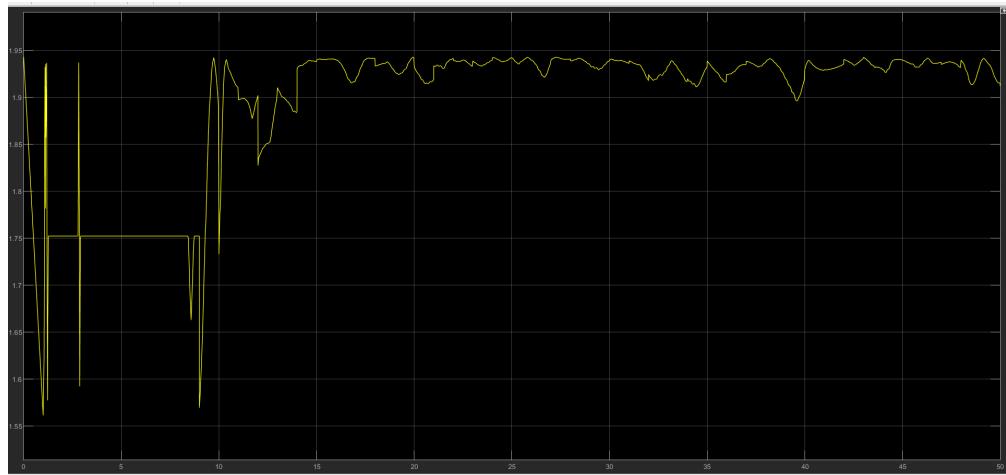
شکل ۷: محیط برنامه



شکل ۸: محیط برنامه



شکل ۹: محیط شبیه سازی



شکل ۱۰

۲ سوال دوم

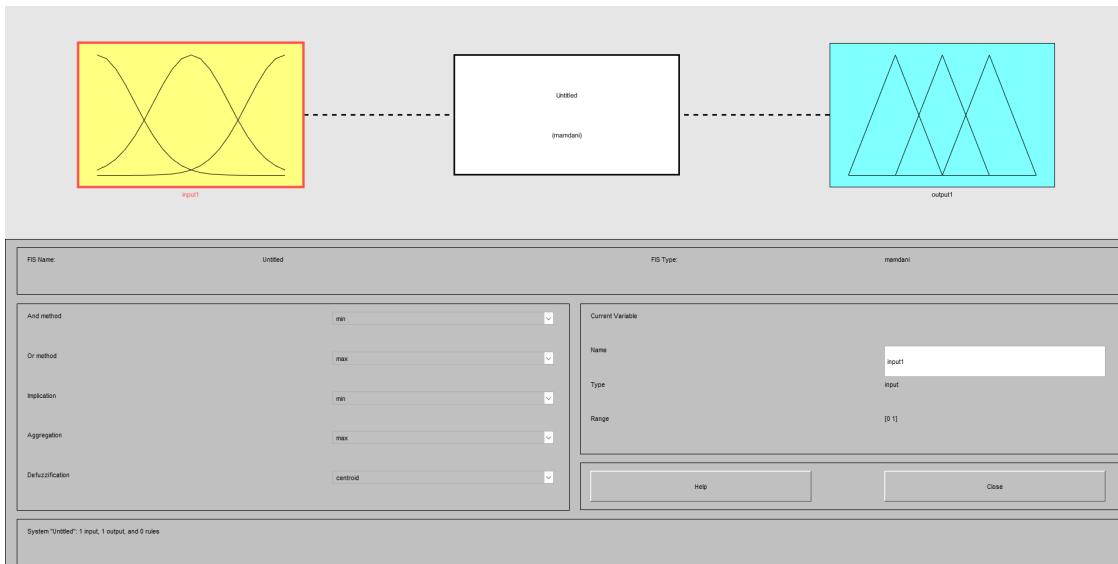
خروجی، توابع تعلق ساز مرکز ثقل ایجاد کرده و متغیرهای ورودی مؤثر استنتاج ضرب معدلگیری و غیرفازی.
مجموعه داده ها را بر مبنای فازی می کنند؛ با ایجاد کنترل کننده های فازی و قواعد فازی را در ساختاری با نام t is می توانیم.
این دو آزمایش، این کنترل را به کامیون اعمال می کنیم به صورتی که عملکرد کنترل کامیون را با شروع از نقاط اولیه جدید مشاهده کنیم. برای این منظور، حلقه تکرار آخر برنامه را نوشته ایم و از مدل تقریبی کامیون استفاده کرده ایم که در آن تمامی زوایا بر حسب رادیان (که در کتاب مرجع درس آورده شده است) بیان می شوند. برای تبدیل این زوایا به درجه در برنامه از ضرب آنها در $\frac{180}{\pi}$ استفاده شده است.
در این رابطه، b طول کامیون است.



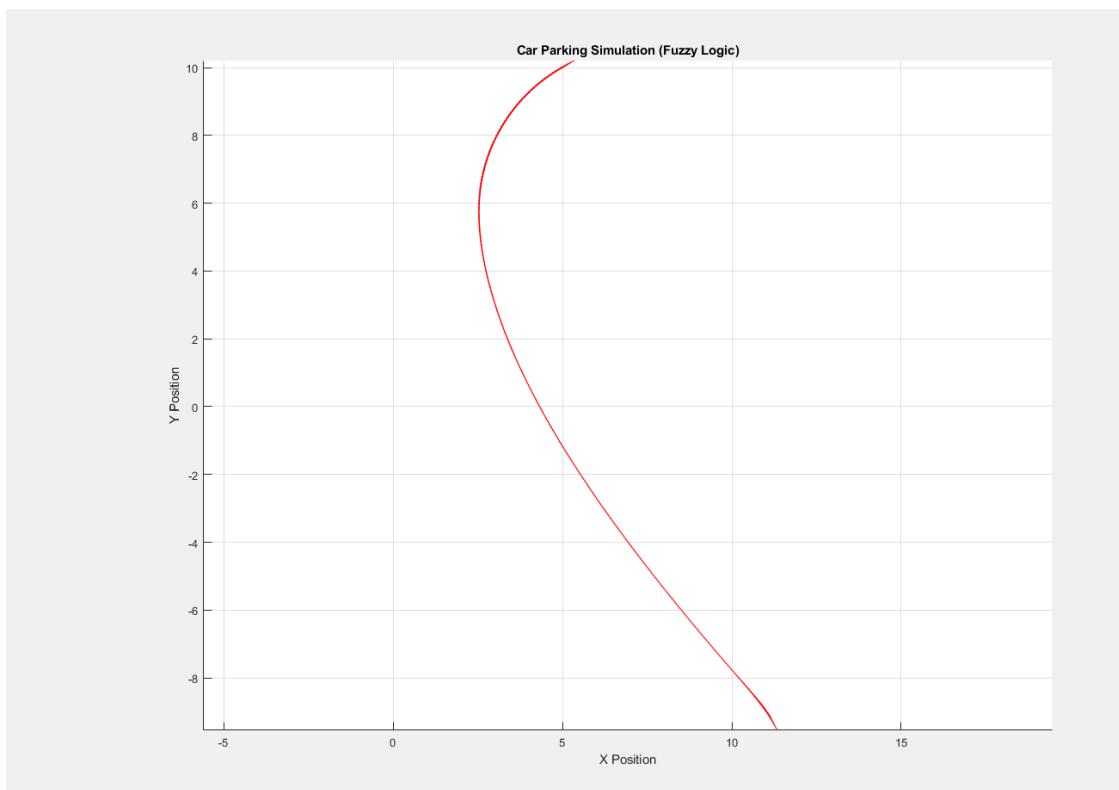
$$x(t+1) = x(t) + \cos[\rho(t) + \theta(t)] + \sin[\theta(t)] \sin[\rho(t)]$$

$$y(t+1) = y(t) + \sin[\rho(t) + \theta(t)] - \sin[\theta(t)] \cos[\rho(t)]$$

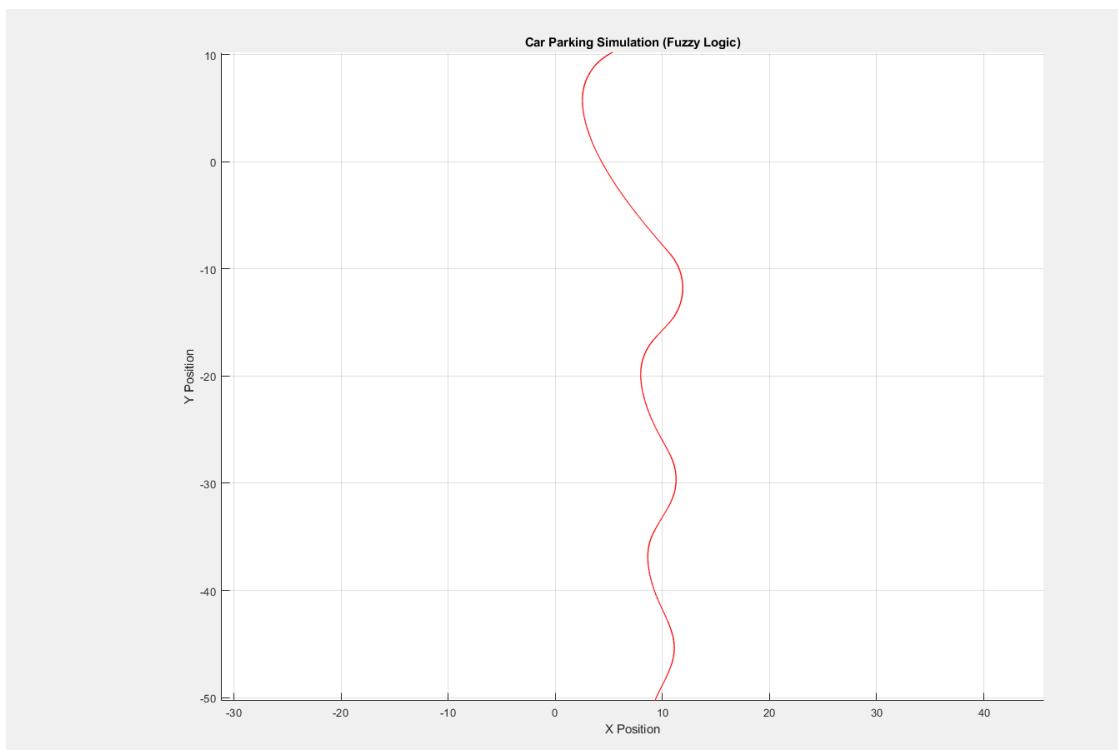
$$\rho(t+1) = \rho(t) - \sin^{-1} \left[\frac{\sin(\theta(t))}{b} \right]$$



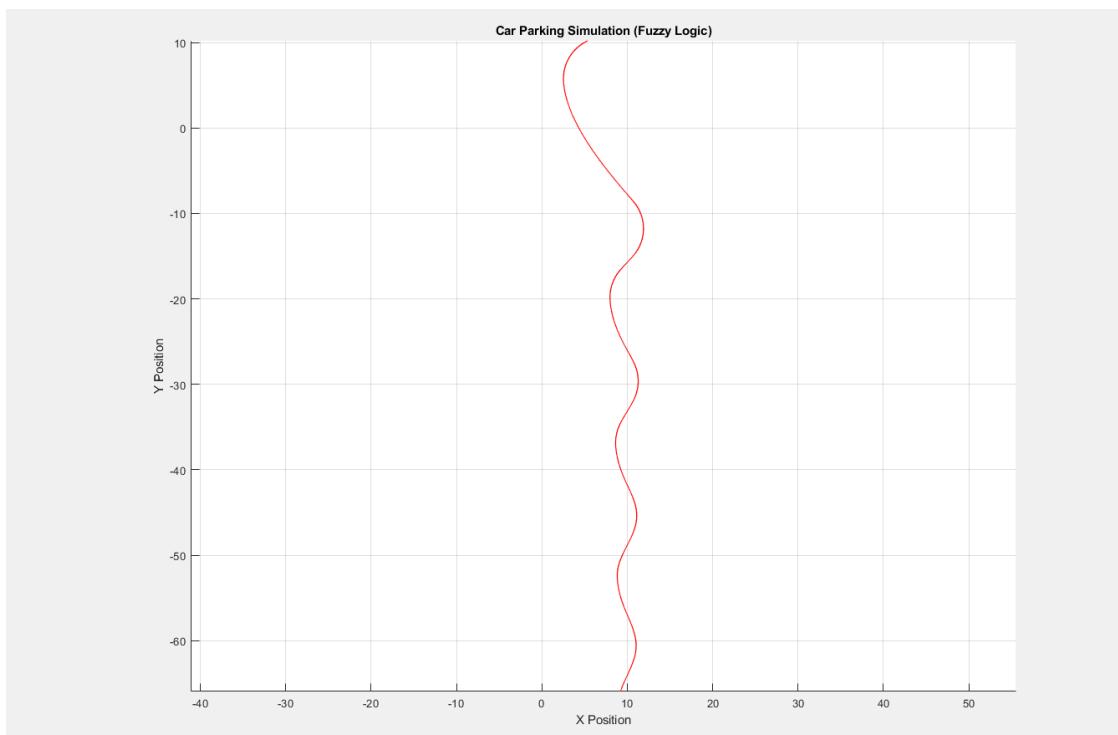
شکل ۱۱: نمایش قسمت تنظیم فازی



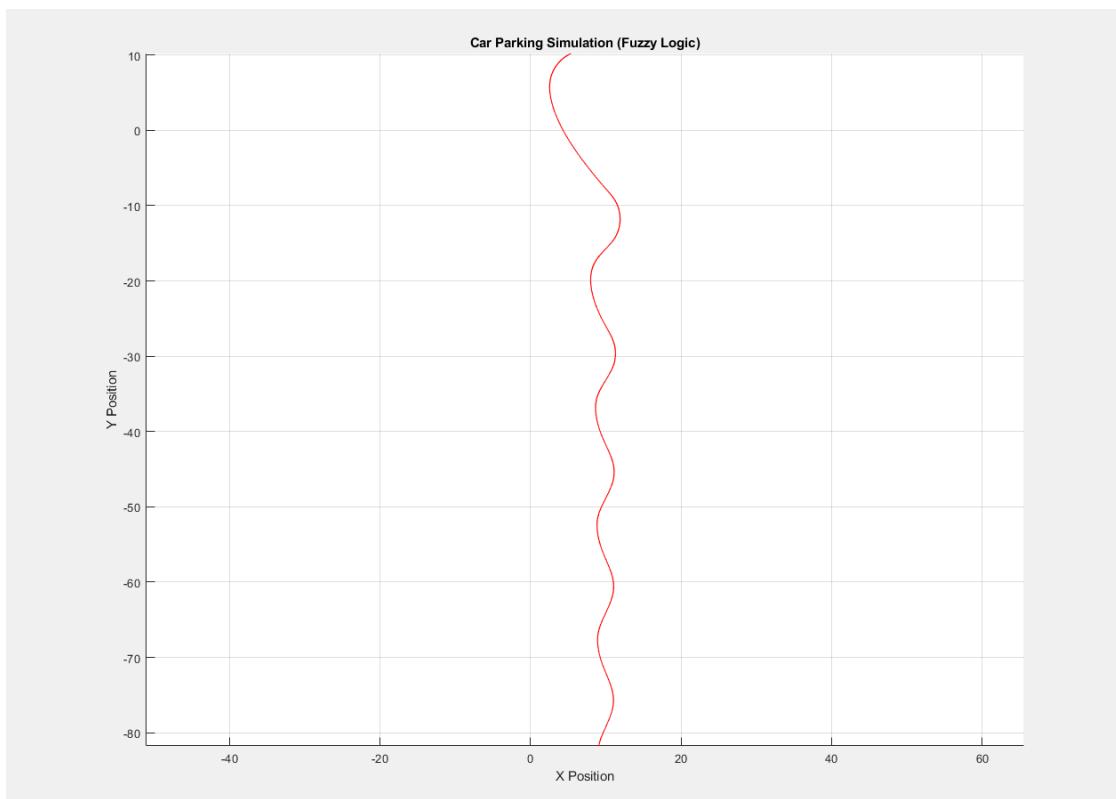
شکل ۱۲: نمایش مراحل



شکل ۱۳: نمایش مراحل



شکل ۱۴: نمایش مراحل



شکل ۱۵: نمایش مراحل

در این مسئله، ورودی‌ها و خروجی‌های سیستم فازی به صورت زیر تعریف شده‌اند:

۱. ورودی‌ها

- زاویه دوران ماشین با ۴ تابع تعلق: Φ

- Up: ماشین در حالت بالا یا نزدیک به زاویه ۹۰ درجه.

- DownNotRight: ماشین در حالت پایین.

- Right: ماشین در زاویه راست.

- فاصله ماشین از نقطه هدف با ۲ تابع تعلق $:distance_to_target$

- Positive: فاصله مثبت (ماشین در سمت راست نقطه هدف).

- Negative: فاصله منفی (ماشین در سمت چپ نقطه هدف).

۲. خروجی

- زاویه فرمان با ۳ تابع تعلق: Θ

- Left: فرمان به سمت چپ.

- Right: فرمان به سمت راست.

- Center: فرمان در حالت مستقیم.



۳ سوال سوم

۴ بارگذاری داده‌ها

```
\ data = readtable('ballbeam.dat');
% First column as input
y = data(:, 2); % Second column as output
```

فایل داده‌ای ballbeam.dat بارگذاری می‌شود. ستون اول به عنوان ورودی (x) و ستون دوم به عنوان خروجی (y) در نظر گرفته شده است.

۵ نرمال‌سازی داده‌ها

```
\ x = (x - min(x)) / (max(x) - min(x)); % Normalize input
y = (y - min(y)) / (max(y) - min(y)); % Normalize output
```

هدف: مقادیر ورودی و خروجی به بازه $[0, 1]$ نرمال‌سازی می‌کنیم تا مدل سریع‌تر و دقیق‌تر عمل کند.

۶ تقسیم داده‌ها به آموزشی و تست

```
\ rng(73); % Set seed for reproducibility
indices = randperm(length(x)); % Random permutation of indices
train_indices = indices(1:num_train); % Training indices
test_indices = indices(num_train+1:end); % Testing indices
%
% Create training and testing sets
x_train = x(train_indices);
y_train = y(train_indices);
x_test = x(test_indices);
y_test = y(test_indices);
```

هدف: داده‌ها به دو مجموعه تقسیم می‌شوند:

- آموزشی: ۸۰٪ از داده‌ها برای آموزش مدل.
- تستی: ۲۰٪ برای ارزیابی مدل.

۷ تنظیمات و تولید مدل ANFIS



```

1 opt = genfisOptions('GridPartition');
2 opt.NumMembershipFunctions = 3; % Set number of membership functions
3 opt.InputMembershipFunctionType = 'gbellmf'; % Generalized bell-shaped MF
4
5 % Generate initial FIS
6 infis = genfis(x_train, y_train, opt);

```

تنظیمات اولیه سیستم استنتاج فازی تطبیقی (ANFIS) ایجاد می‌کنیم.
ویژگی‌ها:

- تعداد توابع عضویت: ۳ (NumMembershipFunctions)
- نوع تابع عضویت: زنگی تعمیم‌یافته (gbellmf)
- دستور genfis مدل اولیه فازی را ایجاد می‌کند.

۸ آموزش مدل ANFIS

```

1 train_data = [x_train, y_train]; % Combine input and output for training
2 opt_train = anfisOptions('InitialFIS', infis, 'EpochNumber', 100);
3 mynetwork = anfis(train_data, opt_train);

```

۹ پیش‌بینی و ارزیابی مدل

```

1 % Prediction
2 y_pred_train = evalfis(mynetwork, x_train); % Predict on training data
3 y_pred_test = evalfis(mynetwork, x_test); % Predict on testing data
4
5 % Calculate performance metrics
6 mse_train = mean((y_train - y_pred_train).^2); % MSE for training
7 mse_test = mean((y_test - y_pred_test).^2); % MSE for testing
8
9 % Display performance metrics
10 fprintf('MSE (Train): %.4f\n', mse_train);
11 fprintf('MSE (Test): %.4f\n', mse_test);
12
13 % Plot results
14 figure;
15 hold on;
16 plot(x_train, y_train, 'b', 'DisplayName', 'Training Data'); % Training data
17 plot(x_train, y_pred_train, 'r', 'DisplayName', 'Predicted (Train)'); % Predicted train

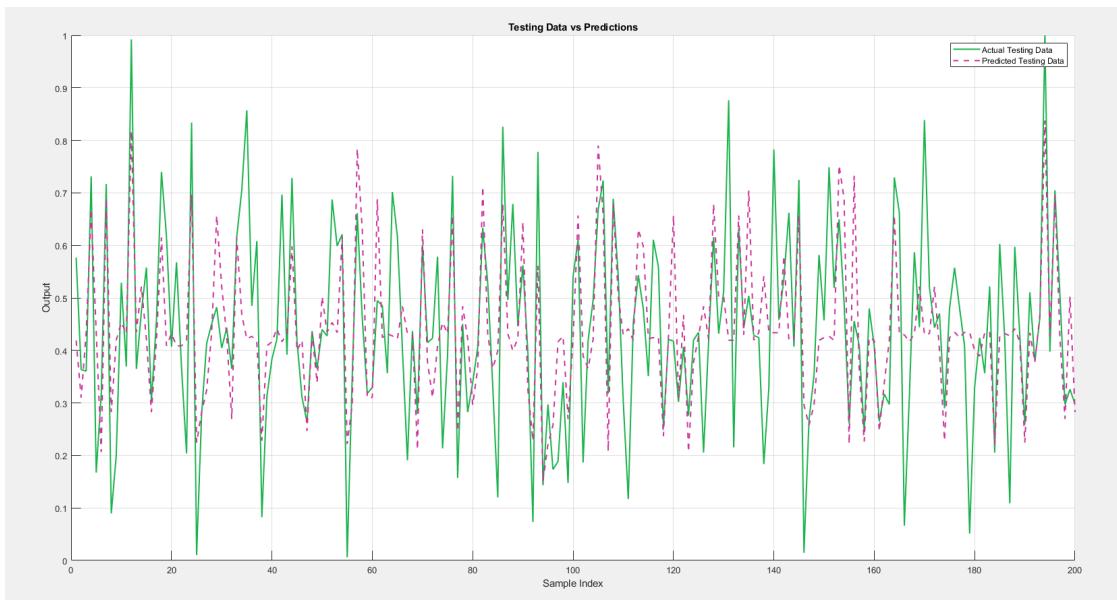
```



```

۱۸ xlabel('Input');
۱۹ ylabel('Output');
۲۰ title('Train ANFIS Results for Ball and Beam');
۲۱ legend;
۲۲
۲۳ figure;
۲۴ hold on;
۲۵ plot(x_test, y_pred_test, 'm', 'DisplayName', 'Predicted (Test)'); % Predicted test
۲۶ plot(x_test, y_test, 'g', 'DisplayName', 'Testing Data'); % Testing data
۲۷ legend;

```



شکل ۱۶: نمایش خروجی

۱۰ سوال چهارم

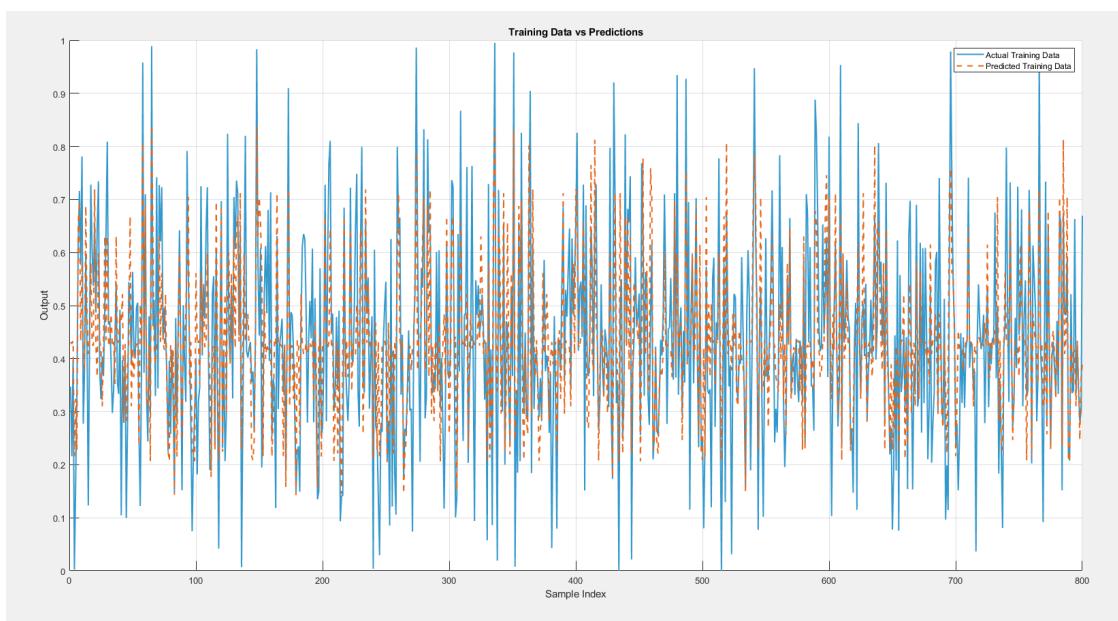
معادله‌ی دیفرانسیل زیر در نظر گرفته شده است:

$$y_{k+1} = 0.3y_k + 0.6y_{k-1} + g[u_k]$$

در این معادله، تابع غیرخطی $g[u]$ به شکل زیر تعریف می‌شود:

$$g[u] = 0.6 \sin(\pi u) + 0.3 \sin(3\pi u) + 0.1 \sin(5\pi u)$$

هدف اصلی، استفاده از یک مدل fuzzy برای تقریب زدن تابع غیرخطی بر اساس داده‌های ورودی و خروجی است. این مدل با به کارگیری روش Least Squares Energy (LSE) آموزش داده خواهد شد.



شکل ۱۷: نمایش خروجی

در مرحله‌ی اول، داده‌های ورودی و خروجی با شبیه‌سازی دینامیک سیستم و تفکیک اجزای خطی و غیرخطی تولید می‌شوند.
کد مربوط به شبیه‌سازی:
این کد دینامیک سیستم را محاسبه کرده و بخش غیرخطی را با تفرق سهم خطی استخراج می‌کند.

مدل سیستم فازی

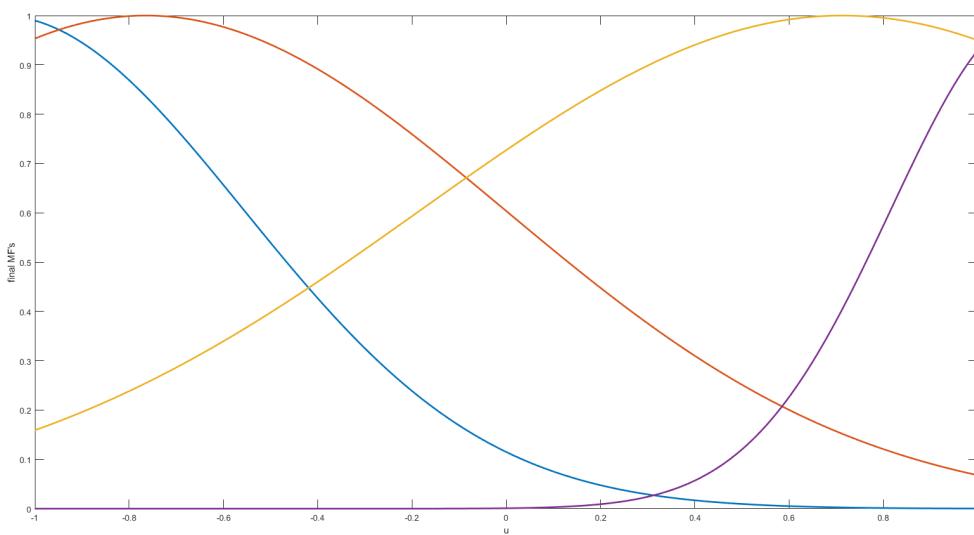
برای ساخت سیستم فازی، ابتدا لیستی از centers و انحراف معیارها ایجاد می‌کنیم. این سیستم فازی با استفاده از توابع عضویت گوسی تعریف شده است. سپس با استفاده از کد زیر، سیستم فازی نسبت به ورودی داده شده، خروجی را محاسبه می‌کند.

آموزش مدل

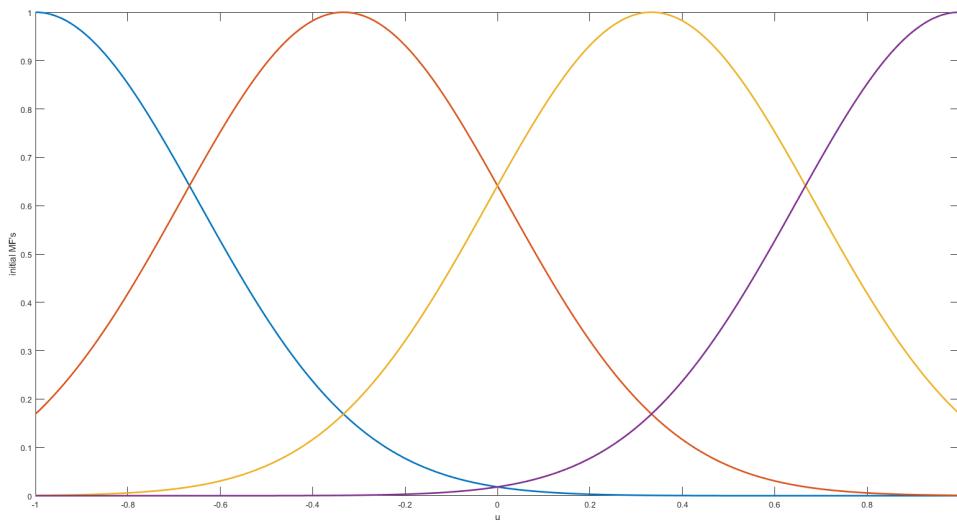
پارامترهای سیستم فازی (شامل مراکز، پهناها و وزن‌ها) با استفاده از روش Gradient Descent بهینه‌سازی می‌شوند. معیار خطأ، میانگین مربعات خطأ (MSE) بین خروجی پیش‌بینی شده و خروجی واقعی است.

نتیجه‌گیری

مدل فازی که با استفاده از روش Gradient Descent آموزش دیده است، به طور مؤثری بخش غیرخطی سیستم را شناسایی کرده است. کارهای آینده می‌تواند شامل بررسی تکنیک‌های بهینه‌سازی پیشرفته‌تر یا استفاده از توابع عضویت پیچیده‌تر برای افزایش دقت مدل باشد.



شکل ۱۸: نمایش فازی

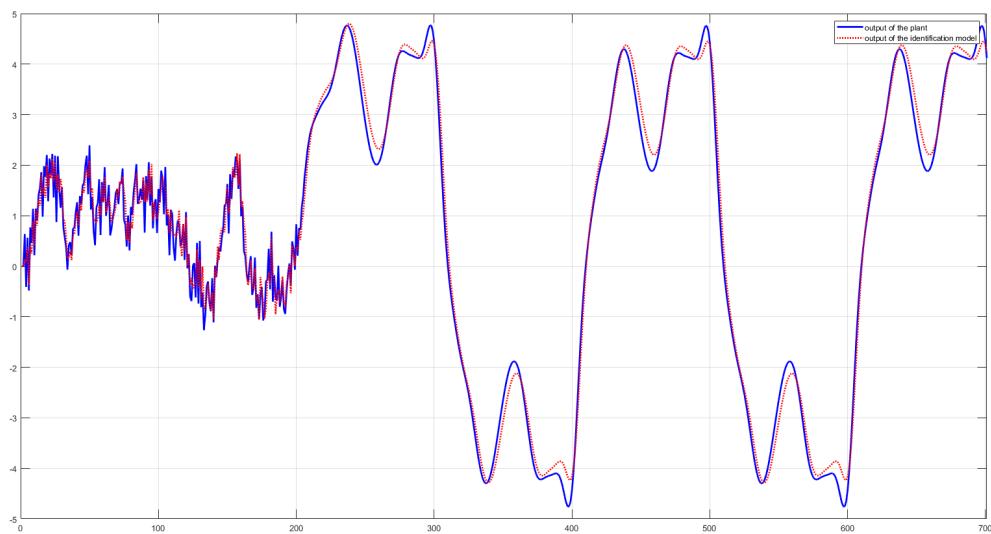


شکل ۱۹: نمایش فازی

۱۱ سوال پنجم

در این سوال، دو مدل مختلف شامل شبکه عصبی با پایه‌های شعاعی (RBF) و سیستم استنتاج فازی تطبیقی (ANFIS) برای مدل‌سازی داده‌های پیچیده و غیرخطی استفاده شده است. هدف از این تحلیل مقایسه عملکرد این دو مدل و تعیین مدل بهتر با توجه به معیارهای میانگین مربعات خطأ (MSE) و ضریب تعیین (R^2) بوده است.

هدف رگرسیون در این مسئله، پیش‌بینی مقدار **PT08.S1 (CO)** است که نمایانگر پاسخ حسگر اکسید قلع (tin oxide) می‌باشد. هم‌چنان، همانطور که در صورت سوال ذکر شده بود، اول دیتا را تقسیم می‌کنیم و باید به این نکته توجه داشته باشیم که طبق اطلاعات موجود در لینک دیتابست مقدار زیادی داده missing data وجود دارد. گفته شده اعداد ۲۰۰- نشانه عدم وجود دیتا می‌باشند، پس این



شکل ۲۰: نمایش خروجی

دیتاها نیز باید حذف شوند.

۹۴.۰ of correlation a with CO(GT) •

۹۳.۰ of correlation a with C₆H₆(GT) •

۹۴.۰ of correlation a with PT_{0.8}.S₂(NMHC) •

۹۵.۰ of correlation a with PT_{0.8}.S₄(NO₂) •

۹۴.۰ of correlation a with PT_{0.8}.S₅(O₃) •

ویژگی‌های استفاده شده

ما از این ویژگی‌ها استفاده می‌کنیم:

CO(GT) •

C₆H₆(GT) •

PT_{0.8}.S₂(NMHC) •

۸۴۸۸۵۹۵۳۵۳۹۶.۷۸۵۳ :Mean Squared Error •

۸۶۱۱۴۶۳۱۴۳۶۳۸۵۴.۰ :R-Squared •

در پیش‌بینی داده‌های تست می‌باشد. با توجه به این مقادیر، عملکرد متوسط مدل RBF نشان داده شده است. با R^2 نسبتاً پایین و MSE بالا، این مدل توانایی محدودی در مدل‌سازی دقیق داده‌ها دارد.



Model ANFIS

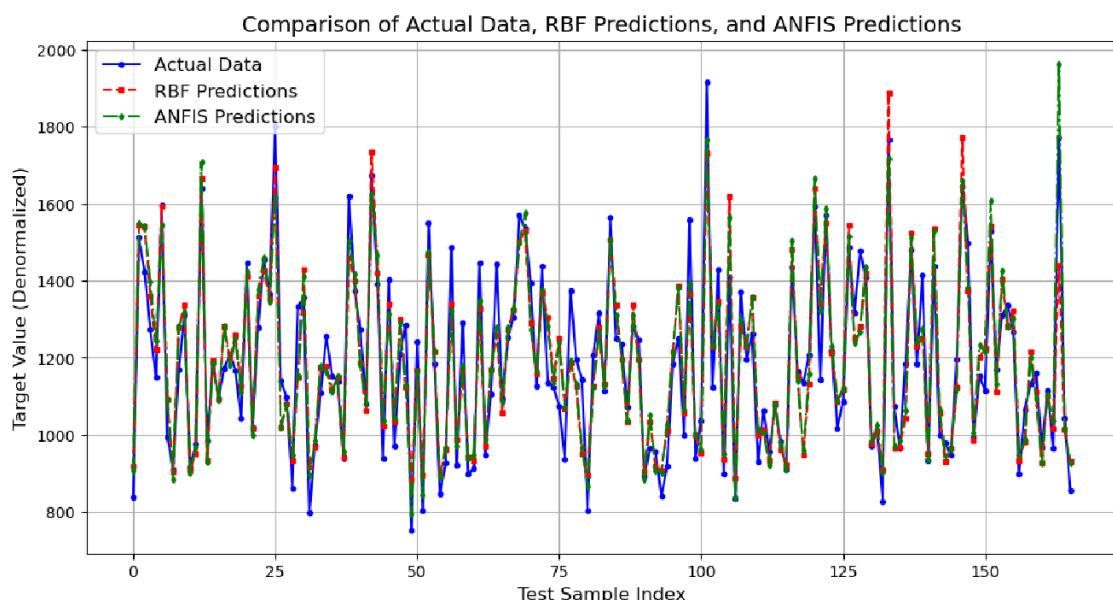
۸۳۳۳۱۱۶۳۸۷۶۶.۶۸۱۹ :Mean Squared Error •

۸۷۹۴۲۷۳۹۸۲۵۰۰۳۱۶.۰ :R-Squared •

مدل ANFIS با MSE پایین تر و R^2 بالاتر، عملکرد بسیار بهتری در پیش‌بینی داده‌ها داشته است. این مقادیر نشان می‌دهند که مدل ANFIS توانسته است روابط پیچیده بین متغیرها را بهتر یاد بگیرد و داده‌های تست را با دقت بیشتری پیش‌بینی کند.

۱۲ نتیجه‌گیری

نتایج این پروژه نشان می‌دهد که مدل ANFIS عملکرد بهتری در مقایسه با مدل RBF دارد. این برتری عمدتاً به توانایی ANFIS در شبیه‌سازی روابط غیرخطی پیچیده و بهره‌گیری از سیستم استنتاج فازی بر می‌گردد، که باعث درک دقیق‌تر از عدم قطعیت و انعطاف‌پذیری داده‌ها می‌شود. در مقابل، مدل RBF به دلیل محدودیت‌هایی در شبیه‌سازی تعریف خوش‌ها و انتشار شعاعی، دقت پایین‌تری ارائه داده است. بنابراین، توصیه می‌شود برای مسائلی که شامل داده‌های پیچیده و غیرخطی هستند، از مدل ANFIS استفاده شود.



شکل ۲۱: نمایش نهایی