SPRAWOZDANIE

ZASTOSOWANIE STEROWNIKA ROZMYTEGO W MATLABIE

Przedmiot: *Podstawy sztucznej inteligencji*

Autor: *Dávid Ali*

Kontakt: *s26567@st.pans.nysa.pl*

Data: *22. 11. 2023 r.*

Spis treści

Cel

Celem tego ćwiczenia było opracowanie sterownika rozmytego pozwalającego określić poziom przyspieszenia samochodu w zależności od odległości od następnego pojazdu, aktualnej prędkości oraz mokrości drogi.

Zagadnienia wstępne

System wnioskowania rozmytego

Skrót **FIS** odnosi się do systemu wnioskowania rozmytego (ang. Fuzzy Inference System). Jest to system oparty na logice rozmytej, który używa reguł i zbiorów rozmytych (funkcji przynależności rozmytej) do podejmowania decyzji na podstawie nieprecyzyjnych danych wejściowych.

FIS jest strukturą, która składa się z:

Zbiorów rozmytych (ang. Membership Functions – MF): Definiują kształt funkcji przynależności dla zmiennych wejściowych i wyjściowych.

Reguł sterowania (ang. Control Rules): Opisują relacje między zbiorami wejściowymi a wyjściowymi za pomocą warunków „jeśli... to...”.

Silnika wnioskowania (ang. Inference Engine): Odpowiada za przetwarzanie danych wejściowych zgodnie z regułami sterowania w celu uzyskania wyników na wyjściu.

Mechanizmu defuzyfikacji (ang. Defuzzification): Konwertuje rozmyte wyniki z silnika wnioskowania na konkretne wartości liczbowe.

FIS jest wykorzystywany w sterowaniu rozmytym, gdzie precyzyjne decyzje nie są możliwe lub gdy dane wejściowe są niejednoznaczne lub trudne do opisania za pomocą tradycyjnych metod.

Przykłady zastosowań FIS obejmują systemy sterowania samochodami, systemy wnioskowania medycznego, systemy diagnozowania awarii, sterowanie procesami przemysłowymi itp. Jest to użyteczne narzędzie w sytuacjach, gdzie konwencjonalne metody sterowania mogą być niewystarczające lub nieelastyczne wobec zmienności danych.

Stworzenia systemu rozmytego

Definicja zmiennych wejściowych i wyjściowej

Zmienna wejściowa 1: Odległość *D* [m] (blisko, średnio daleko, daleko).

Zmienna wejściowa 2: Aktualna prędkość *V* [km/h] (wolno, średnio, szybko).

Zmienna wejściowa 3: Ilość opadu na drodze *X* [mm] (sucha, mokra).

Zmienna wyjściowa: Przyspieszenie *A* [m/s²] (zwolnienie, utrzymanie, przyspieszenie).

Definicja funkcji przynależności

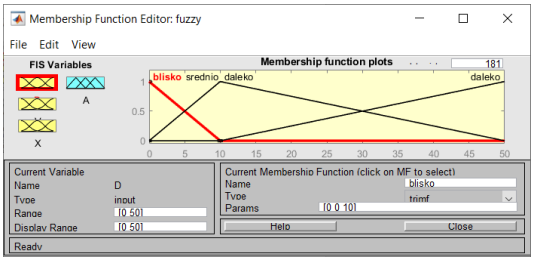
Dla każdej zmiennej wejściowej i wyjściowej trzeba zdefiniować funkcje przynależności (ang. Membership Functions – MF), np. trójkątne lub trapezoidalne, które opisują różne poziomy każdej zmiennej. Wybór funkcji przynależności ma wpływ na zachowanie i wydajność systemu sterowania rozmytego. W przypadku tego konkretnego systemu, wybrano funkcje trójkątne dla każdego wejścia i wyjścia. Funkcje trójkątne są często używane dla wyjść w sterownikach rozmytych ze względu na swoją prostotę i interpretowalność.

Zmienna wejściowa 1: Odległość *D* [m]

Zakres odległości: od 0 do 50 metrów.

Tabela 1: Funkcje przynależności zmiennej wejściowej D

Nazwa	Typ	Parametry
blisko	trimf	[0 0 10]
srednio_daleko	trimf	[0 10 50]
daleko	trimf	[10 50 50]



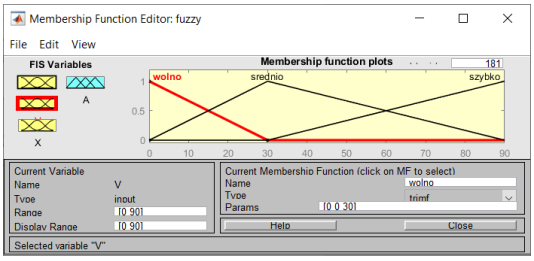
Rysunek 1: Zmienna wejściowa D

Zmienna wejściowa 2: Aktualna prędkość *V* [km/h]

Zakres prędkości: od 0 do 90 km/h.

Tabela 2: Funkcje przynależności zmiennej wejściowej V

Nazwa	Typ	Parametry
wolno	trimf	[0 0 30]
srednio	trimf	[0 30 90]
szybko	trimf	[30 90 90]



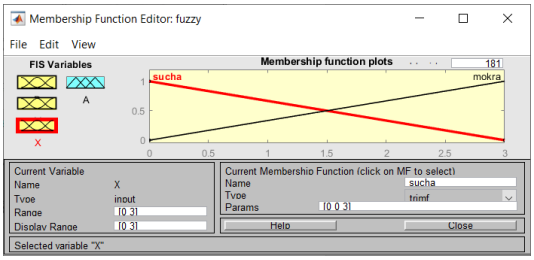
Rysunek 2: Zmienna wejściowa V

Zmienna wejściowa 3: Opady na drodze *X* [mm]

Zakres ilości opadu: od 0 do 3 mm.

Tabela 3: Funkcje przynależności zmiennej wejściowej X

Nazwa	Typ	Parametry
sucha	trimf	[0 0 3]
mokra	trimf	[0 3 3]



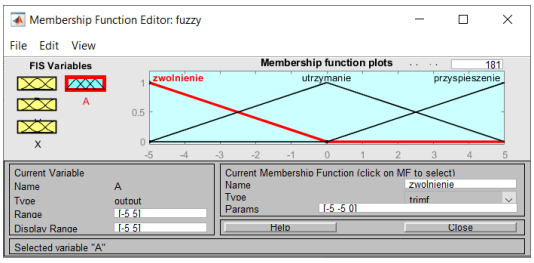
Rysunek 3: Zmienna wejściowa X

Zmienna wyjściowa: Przyspieszenie *A* [m/s²]

Zakres przyspieszenia: od -5 do 5 m/s².

Tabela 4: Funkcje przynależności zmiennej wyjściowej A

Nazwa	Typ	Parametry
zwolnienie	trimf	[-5 -5 0]
utrzymanie	trimf	[-5 0 5]
przyspieszenie	trimf	[0 5 5]



Rysunek 4: Zmienna wyjściowa A

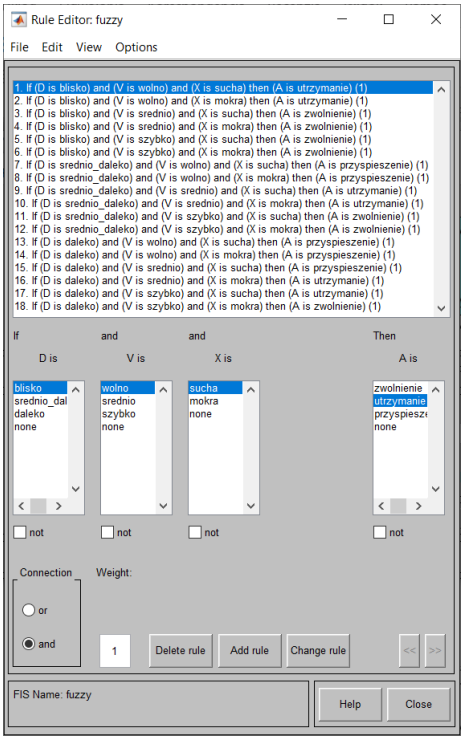
Warto zauważyć, że ocena jakości funkcji przynależności wymagałaby także testów i analizy na rzeczywistych danych lub symulacjach, aby potwierdzić, czy wybór tych funkcji przynależności jest optymalny dla danego problemu. Jednakże, w teorii, wybór trójkątnych funkcji przynależności zarówno dla wejść jak i wyjścia wydaje się być sensownym i adekwatnym wyborem w kontekście opisywanego systemu sterowania rozmytego.

Tworzenie bazy reguł

Baza reguł definiuje związki między zmiennymi wejściowymi a wyjściową, wykorzystując warunki *„jeśli... to...”*. W tym przypadku, przyjmując trzy zmienne wejściowe (odległość, prędkość, ilość opadu) oraz jedną zmienną wyjściową (przyspieszenie), możemy określić reguły oparte na tych zmiennych. Importując plik typu *.fis* w Fuzzy Logic Designerze można ją modyfikować w intuicyjnym graficznym środowisku, aby definiować reguły takich jak np. *„Jeśli D jest blisko i V jest wolno i X jest sucha, to A jest utrzymanie.”*

Tabela 5: baza reguł

Nr.	D	V	X	A
1	blisko	wolno	sucha	utrzymanie
2	blisko	wolno	mokra	utrzymanie
3	blisko	wolno	sucha	zwolnienie
4	blisko	srednio	mokra	zwolnienie
5	blisko	srednio	sucha	zwolnienie
6	blisko	szybko	mokra	zwolnienie
7	srednio_daleko	szybko	sucha	przyspieszenie
8	srednio_daleko	wolno	mokra	przyspieszenie
9	srednio_daleko	wolno	sucha	utrzymanie
10	srednio_daleko	wolno	mokra	utrzymanie
11	srednio_daleko	srednio	sucha	zwolnienie
12	srednio_daleko	srednio	mokra	zwolnienie
13	daleko	szybko	sucha	przyspieszenie
14	daleko	szybko	mokra	przyspieszenie
15	daleko	wolno	sucha	przyspieszenie
16	daleko	wolno	mokra	utrzymanie
17	daleko	srednio	sucha	utrzymanie
18	daleko	srednio	mokra	zwolnienie



Rysunek 5: baza reguł

Testowanie i optymalizacja

Przetestowanie systemu na różnych kombinacjach danych wejściowych, aby sprawdzić, czy uzyskane wyniki są zgodne z oczekiwaniami. Dostosowanie funkcji przynależności i reguły w razie potrzeby, aby uzyskać pożądane wyniki.

Implementacja systemu

Po zdefiniowaniu systemu rozmytego można użyć go w skrypcie MATLABa do symulacji i przewidywania odpowiednich przyspieszeń na podstawie danych wejściowych.

Skrypt MATLAB *script.m*

Skrypt wczytuje plik konfiguracyjny *fuzzy.fis*, wyświetla właściwości systemu, tworzy wykresy funkcji przynależności dla wejść i wyjścia, generuje powierzchnię odpowiedzi systemu oraz ewaluuje go dla różnych zestawów danych wejściowych. Wyniki ewaluacji zostały wyświetlone dla konkretnych przypadków, prezentując przyspieszenie samochodu w zależności od odległości, prędkości i ilości opadu na drodze.

Załadowanie wcześniej omówionego pliku *fuzzy.fis*, jednocześnie wyświetlanie informacji o załadowanym systemie wnioskowania rozmytego:

```
fuzzy = readfis("fuzzy.fis")
```

```
fuzzy =

  mamfis with properties:

                      Name: "fuzzy"
                 AndMethod: "min"
                  OrMethod: "max"
           ImplicationMethod: "prod"
           AggregationMethod: "sum"
       DefuzzificationMethod: "centroid"
                    Inputs: [1×3 fisvar]
                   Outputs: [1×1 fisvar]
                     Rules: [1×18 fisrule]
     DisableStructuralChecks: 0

    See 'getTunableSettings' method for parameter optimization.
```

Tworzenie figury:

```
fig = figure;
```

Graficzne przedstawienie systemu FIS za pomocą *plotfis*:

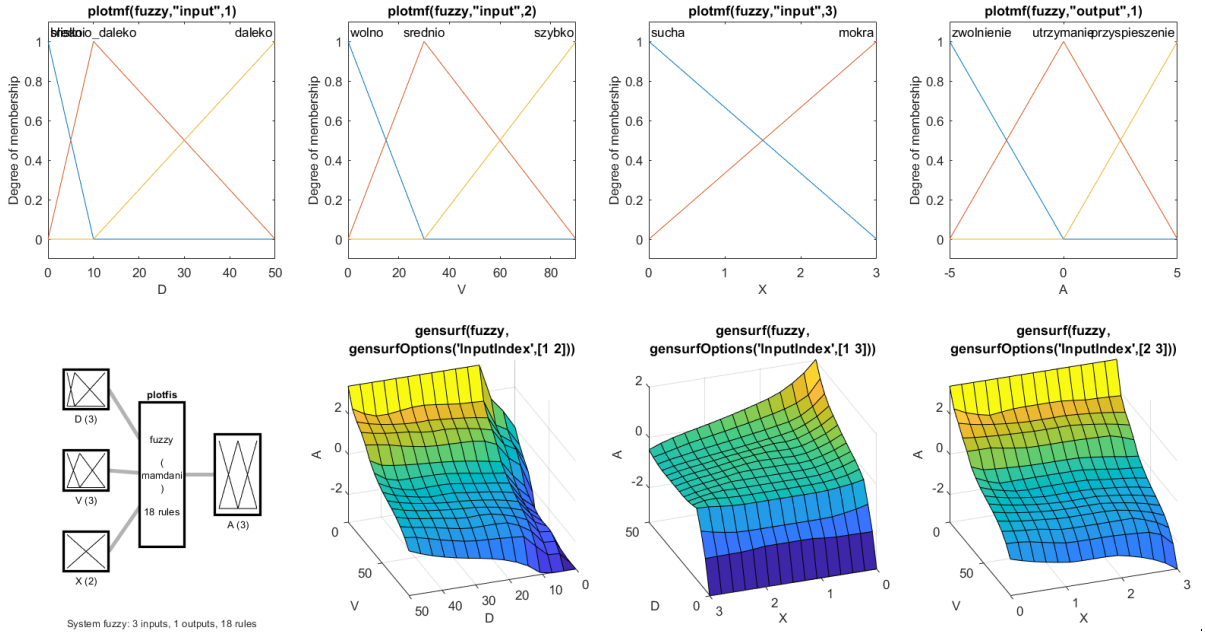
```
subplot(2,4,5);
plotfis(fuzzy)
title("plotfis");
```

Generowanie wykresu funkcji przynależności dla każdego zbioru wejściowego i wyjściowego:

```
subplot(2,4,1);
plotmf(fuzzy,"input",1);
title('plotmf(fuzzy,"input",1)');

subplot(2,4,2);
plotmf(fuzzy,"input",2);
title('plotmf(fuzzy,"input",2)');

subplot(2,4,3);
plotmf(fuzzy,"input",3);
title('plotmf(fuzzy,"input",3)');

subplot(2,4,4);
plotmf(fuzzy,"output",1);
title('plotmf(fuzzy,"output",1)');
```

Generowanie trójwymiarowych powierzchni, które przedstawiają odpowiedzi systemu sterowania rozmytego dla różnych kombinacji danych wejściowych:

```
subplot(2,4,6);
gensurf(fuzzy, gensurfOptions('InputIndex',[1 2]));
view(160, 30);
title(sprintf("gensurf(fuzzy,\ngensurfOptions('InputIndex',[1 2]))"));

subplot(2,4,7);
gensurf(fuzzy, gensurfOptions('InputIndex',[1 3]));
view(250, 30);
title(sprintf("gensurf(fuzzy,\ngensurfOptions('InputIndex',[1 3]))"));

subplot(2,4,8);
gensurf(fuzzy, gensurfOptions('InputIndex',[2 3]));
view(70, 30);
title(sprintf("gensurf(fuzzy,\ngensurfOptions('InputIndex',[2 3]))"));
```



Rysunek 6: Wygenerowane wykresy (przy użyciu funkcji wbudowanych)

Definicja liczby punktów siatki dla każdego wymiaru:

```
N = 20;
```

Definicja minimalnych i maksymalnych wartości dla parametru odległości:

```
D_min = 0;
D_max = 50;
```

Utworzenie wektora wartości równomiernie rozłożonych dla parametru odległości:

```
D = linspace(D_min, D_max, N);
```

Definicja minimalnych i maksymalnych wartości dla parametru prędkości:

```
V_min = 0;
V_max = 120;
```

Utworzenie wektora wartości równomiernie rozłożonych dla parametru prędkości:

```
V = linspace(V_min, V_max, N);
```

Definicja minimalnych i maksymalnych wartości dla parametru opadu

```
X_min = 0;
X_max = 3;
```

Utworzenie wektora wartości równomiernie rozłożonych dla parametru opadu:

```
X = linspace(X_min, X_max, 6);
```

Definicja palety kolorów dla wykresu powierzchni:

```
cmap = [[linspace(1, 1) 0 linspace(1,0)]',[linspace(0, 1) 0
linspace(1,1)]',[linspace(0, 1) 1 linspace(1,0)]'];
```

Utworzenie siatki dla wartości odległości i prędkości:

```
[D_grid, V_grid] = meshgrid(D, V);
```

Inicjalizacja pustej macierzy do przechowywania wartości przyspieszenia:

```
A = zeros(length(D), length(V));
```

Utworzenie dwóch figur do wyświetlania wykresów powierzchni:

```
fig1 = figure;
fig2 = figure;
```

Iteracja po wszystkich wartościach opadu:

```
for x = 1:length(X)
```

Iteracja po wszystkich wartościach odległości i prędkości oraz obliczenie przyspieszenia poprzez wyliczenie systemu wnioskowania rozmytego:

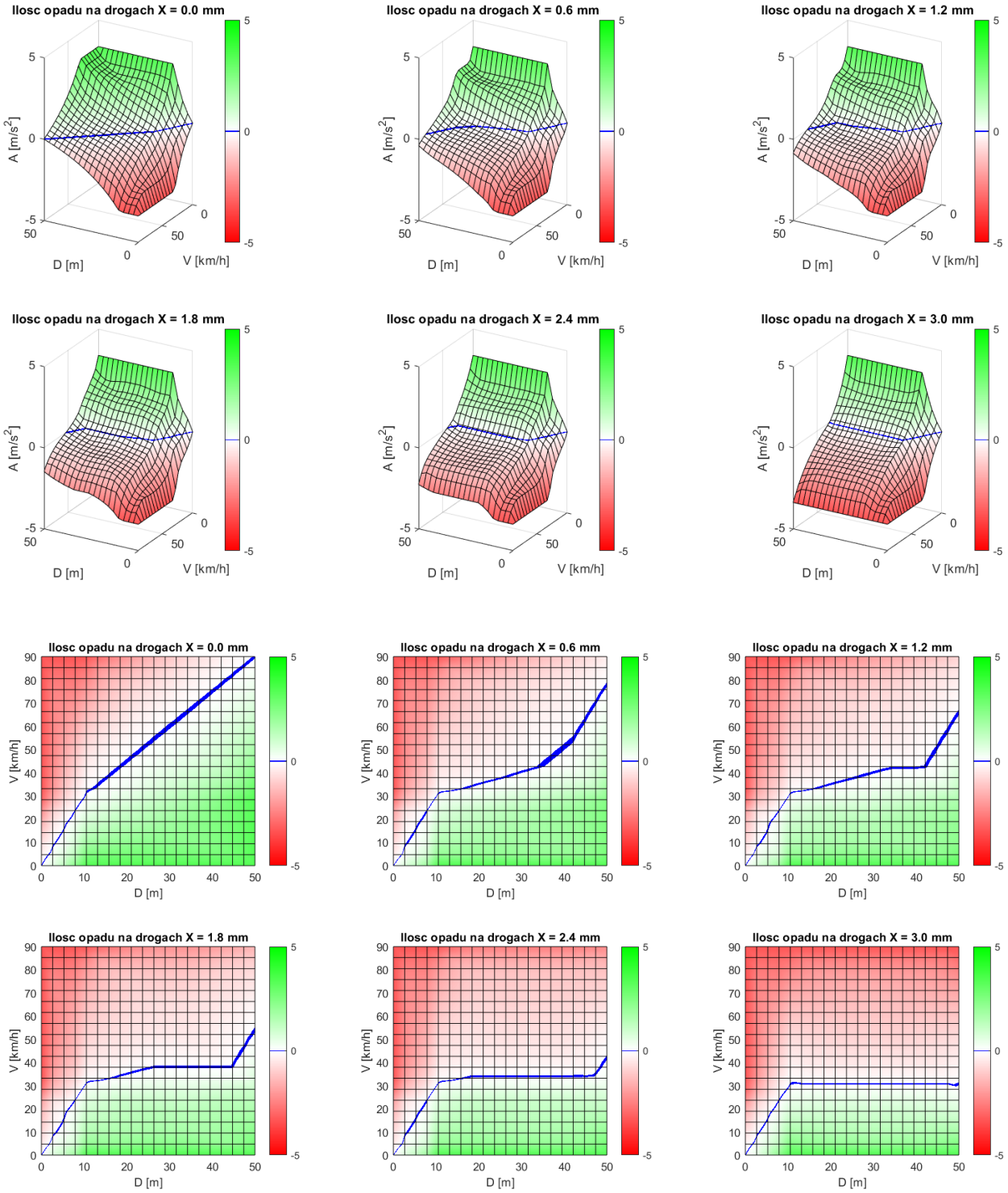
```
    for i = 1:length(D)
        for j = 1:length(V)
            A(i, j) = evalfis(fuzzy, [D(i) V(j) X(x)]);
        end
    end
```

Utworzenie wykresu powierzchni wartości przyspieszenia dla bieżącej wartości opadu:

```
    figure(fig1);
    subplot(2, 3, x);
    s = surf(D_grid, V_grid, A');
    s.EdgeColor = 'black';
    s.FaceColor = 'interp';
    colorbar;
    caxis([-5 5]);
    colormap(cmap);
    xlabel('D [m]');
    xlim([D_min D_max]);
    ylabel('V [km/h]');
    ylim([V_min V_max]);
    zlabel('A [m/s^2]');
    zlim([-5 5]);
    title(sprintf("Ilość opadu na drogach X = %.1f mm", X(x)));
    view(-150, 15);
```

Utworzenie drugiego wykresu powierzchni wartości przyspieszenia dla bieżącej wartości opadu:

```
    figure(fig2);
    subplot(2, 3, x);
    s = surf(D_grid, V_grid, A');
    s.EdgeColor = 'black';
    s.FaceColor = 'interp';
    colorbar;
    caxis([-5 5]);
    colormap(cmap);
    xlabel('D [m]');
    xlim([D_min D_max]);
    ylabel('V [km/h]');
    ylim([V_min V_max]);
    zlabel('A [m/s^2]');
    zlim([-5 5]);
    title(sprintf("Ilość opadu na drogach X = %.1f mm", X(x)));
    view(0, 90);
end
```



Rysunek 7: Wygenerowane wykresy przy użyciu własnego skryptu

Wnioski

System rozmyty działa zgodnie z określonymi regułami, prezentując różne poziomy przyspieszenia w zależności od zmienności odległości, prędkości oraz ilości opadu na drogach. Wartości przyspieszenia są zgodne z oczekiwaniami w różnych warunkach, co sugeruje poprawność działania modelu w tym kontekście. Interpretacja grafów funkcji przynależności oraz wyników ewaluacji pozwala na zrozumienie zależności między wejściami a wyjściem w systemie sterowania rozmytym.

Jakie są zalety i wady sterowania rozmytym?

Zalety sterowania rozmytego to:

- Jest to podejście oparte na intuicji, które może być łatwiejsze do zrozumienia i zastosowania niż klasyczne metody sterowania.
- Może być używane do sterowania systemami, które są nieliniowe lub niepewne.
- Jest elastyczne i może być dostosowane do różnych zastosowań.

Wady sterowania rozmytego to:

- Może być mniej dokładne niż klasyczne metody sterowania.
- Jest trudniejsze do optymalizacji.
- Wymaga więcej danych do szkolenia niż klasyczne metody sterowania.

Podsumowanie

Zastosowanie sterownika rozmytego w MATLABie pozwoliło na stworzenie modelu sterowania, który reaguje na zmiany odległości, prędkości i ilości opadu na drogach, określając przyspieszenie samochodu. Otrzymane wyniki sugerują, że system rozmyty dobrze odwzorowuje rzeczywistość w tym kontekście.

Ważne uwagi

Ocena jakości modelu sterowania rozmytego wymagałaby bardziej szczegółowej analizy i weryfikacji w warunkach rzeczywistych. W dalszych badaniach można by rozszerzyć zakres testowanych przypadków oraz przeprowadzić weryfikację na dodatkowych danych.

Spis plików

Nazwa pliku	**Opis**
DAVID ALI - FIS - Sprawozdanie.docx	To sprawozdanie w formacie Word.
DAVID ALI - FIS - Sprawozdanie.pdf	To sprawozdanie w formacie pdf.
fuzzy.fis	Plik konfiguracyjny, który zawiera opis struktury i parametrów systemu sterowania rozmytego.
script.m	Script MATLAB zawierający kod do załadowania FIS, wyświetlania wykresów dotyczących FIS, oraz wyniki na kilku przykładach.