ASSIGNMENT REPORT- 2

(*Task : Acceleration control of a vehicle using Fuzzy Logic*)

Submitted by

(Mogalraj Kushal Dath)

Registration Number : 12100559

Course Name : Soft Computing Laboratory

Course Code : INT518

Section Number : K21ML

GitHub Link : GIT_LINK

Submitted to

(Dr . Prakash Kumar Sarangi)

School of Computer Science and Engineering

LOVELY PROFESSIONAL UNIVERSITY

Abstract :

The expanding pace of road accident is disturbing and any vehicle without a viable stopping mechanism is inclined to accident with clearly disastrous impact following. This is because of human blunders in driving which includes response time delays and distraction. Automatic braking system will be created to keep the vehicle steerable and stable and furthermore prevent wheel lock and impact with an obstacle. The objective of this study is to develop a fuzzy Inference system for an antilock braking system.

Introduction :

The purpose of this program is Automatic acceleration and braking adapted system based on the situation of the vehicle.

We want to constitute fuzzy rule base before the beginning of the design. These definitions are made up –

1. **Input signal of the rule base :**
   Distance (Between Vehicle and Obstacle) : small, medium, large
   Current speed : slow, medium, fast (linguistic variables)

2. **Output signal of the FIS :**
   Acceleration of the vehicle : neg_large, neg_small, zero, pos_small, pos_large

<u>We need to definite the Rules to rule base which are following :</u>

Rule 1 : IF distance = small and speed = fast THEN Acceleration = neg_large

Rule 2 : IF distance = small and speed = medium THEN Acceleration = neg_small

Rule 3 : IF distance = medium and speed = fast THEN Acceleration = neg_small

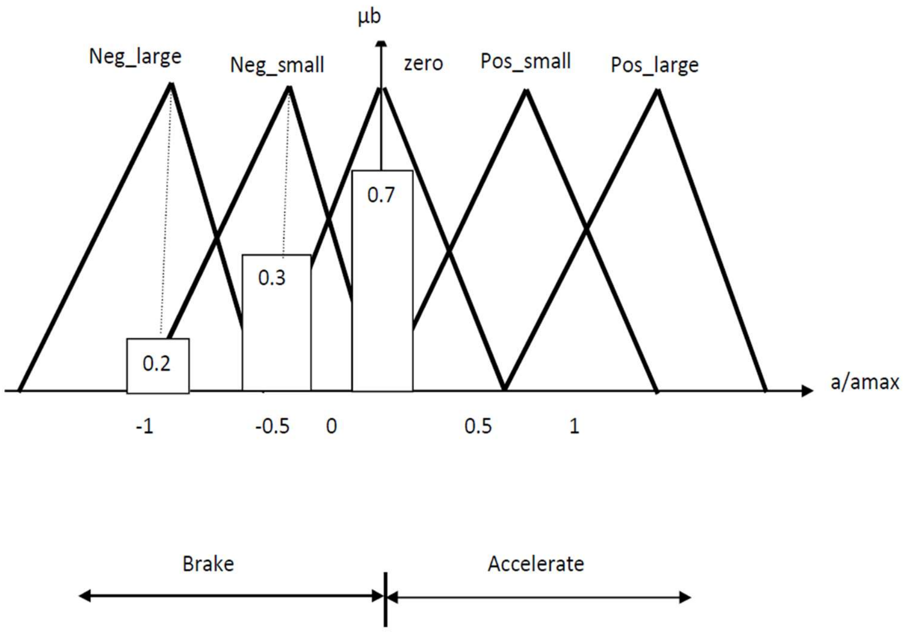
Rule 4 : IF distance = medium and speed = medium THEN Acceleration = zero

Rule 5 : IF distance = medium and speed = slow THEN Acceleration = pos_small

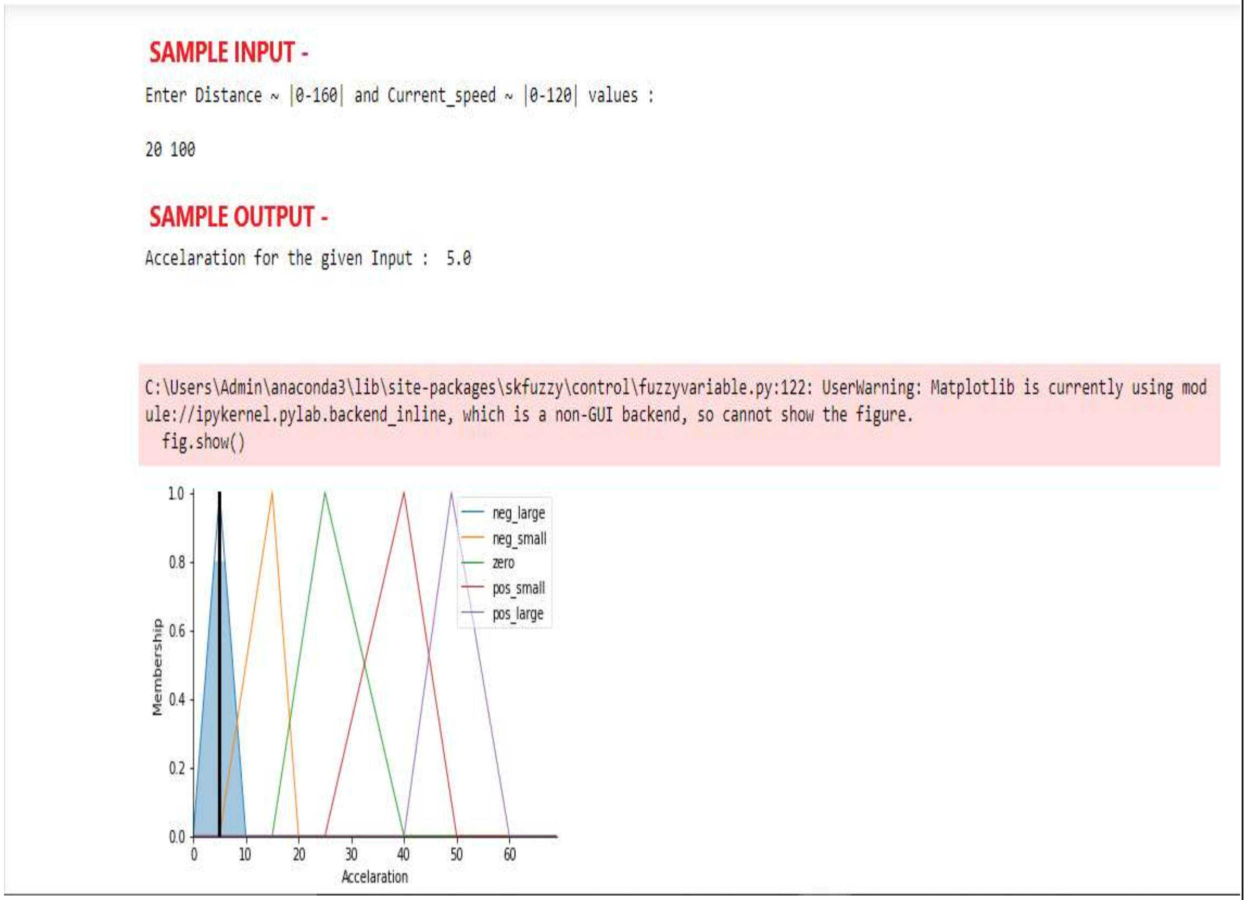
Rule 6 : IF distance = large and speed = medium THEN Acceleration = pos_small

Rule 7 : IF distance = large and speed = slow THEN Acceleration = pos_large

Describing the Output :



Sample Input and Output of the Program :



SAMPLE INPUT -

```
Enter Distance ~ |0-160| and Current_speed ~ |0-120| values :

20 100
```

SAMPLE OUTPUT -

```
Accelaration for the given Input :  5.0
```

```
C:\Users\Admin\anaconda3\lib\site-packages\skfuzzy\control\fuzzyvariable.py:122: UserWarning: Matplotlib is currently using mod
ule://ipykernel.pylab.backend_inline, which is a non-GUI backend, so cannot show the figure.
  fig.show()
```

Program :

1. **Importing Libraries :**

```
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl
import matplotlib.pyplot as plt
```

2. **Determining the Input and Output Signal of the Rule Base :**

```
Distance          = ctrl.Antecedent(np.arange(0,160,1),'Distance')
Current_Speed  = ctrl.Antecedent(np.arange(0,120,1),'Current Speed')
Accelaration     = ctrl.Consequent(np.arange(0,70,1),'Accelaration')
```

3. **User-defined Membership Function :**

```
Distance['small']     = fuzz.trimf(Distance.universe,[0,25,50])
Distance['medium'] = fuzz.trimf(Distance.universe,[40,80,120])
Distance['large']      = fuzz.trimf(Distance.universe,[85,120,160])
Distance.view()

Current_Speed['slow']       = fuzz.trimf(Current_Speed.universe,[0,15,30])
Current_Speed['medium'] = fuzz.trimf(Current_Speed.universe,[20,50,85])
Current_Speed['fast']        = fuzz.trimf(Current_Speed.universe,[70,100,120])
Current_Speed.view()

Accelaration['neg_large']  = fuzz.trimf(Accelaration.universe,[0,5,10])
Accelaration['neg_small']  = fuzz.trimf(Accelaration.universe,[5,15,20])
Accelaration['zero']          = fuzz.trimf(Accelaration.universe,[15,25,40])
Accelaration['pos_small']  = fuzz.trimf(Accelaration.universe,[25,40,50])
Accelaration['pos_large']  = fuzz.trimf(Accelaration.universe,[40,49,60])
Accelaration.view()
```

4. **Set of Rules For Knowledge Base** :

```
rule1 = ctrl.Rule(Distance['small']  & Current_Speed['fast'],
Accelaration['neg_large'])

rule2 = ctrl.Rule(Distance['small']  & Current_Speed['medium'],
Accelaration['neg_small'])

rule3 = ctrl.Rule(Distance['medium'] & Current_Speed['fast'],
Accelaration['neg_small'])

rule4 = ctrl.Rule(Distance['medium'] & Current_Speed['medium'],
Accelaration['zero'])

rule5 = ctrl.Rule(Distance['medium'] & Current_Speed['slow'],
Accelaration['pos_small'])

rule6 = ctrl.Rule(Distance['large']  & Current_Speed['medium'],
Accelaration['pos_small'])

rule7 = ctrl.Rule(Distance['large']  & Current_Speed['slow'],
Accelaration['pos_large'])
```

5. **Creating Fuzzy Inference System Based on the above Rules** :

```
Accelaration_controller =
ctrl.ControlSystem([rule1,rule2,rule3,rule4,rule5,rule6,rule7])

CSS = ctrl.ControlSystemSimulation(Accelaration_controller)
```

6. **Plotting Output Based on User-Input** :

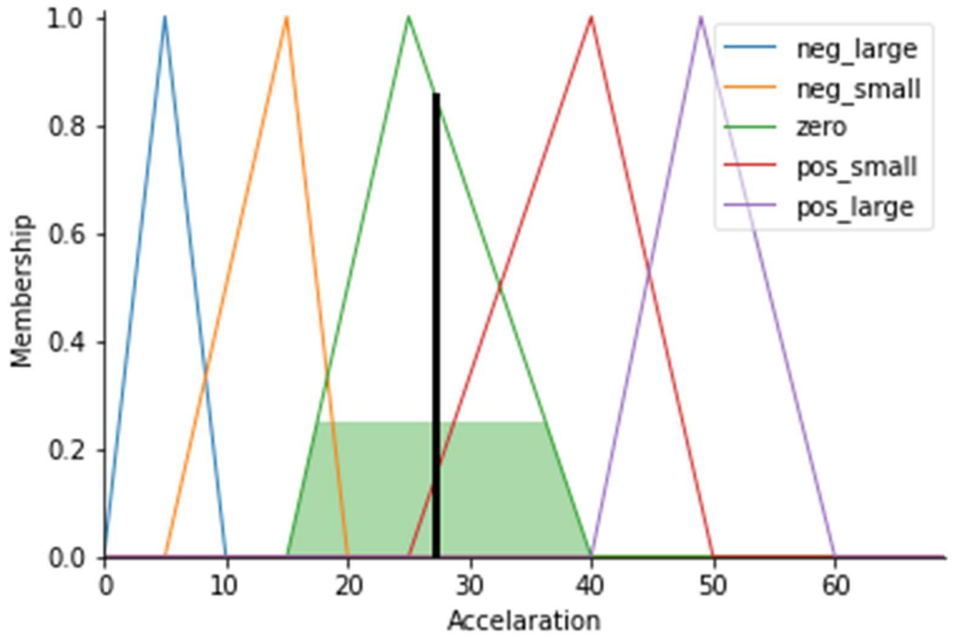
```
while(True):
   print("Enter Distance ~ |0-160| and Current_speed ~ |0-120| values : \n")
   n=list(map(int,input().split()))
   CSS.input['Distance']=n[0]
   CSS.input['Current Speed']=n[1]
   CSS.compute()
   print("\n\n")
   print("Accelaration for the given Input : ",CSS.output['Accelaration'])
   print("\n\n")
   Accelaration.view(sim=CSS)
   break
```

7. **Output of the Program** :

```
Enter Distance ~ |0-160| and Current_speed ~ |0-120| values :

50 70

Accelaration for the given Input :  27.20238095238095
```



Conclusion :

This Program focuses on modelling of an Automatic Braking system using fuzzy logic or can say Accelaration control over Fuzzy Inference System. In future if we implement this fuzzy logic in hardware system of the vehicle then it is intended to solve the problem where drivers fail to apply brakes on time of an emergency and can reduce speed automatically before colliding with the obstacles. This study will enlighten students on the need for automation of vehicle to avoid human errors.