Fuzzy Systems and Computational Intelligence: Car Control

Dimitrios Karatis 10775, *Electrical and Computer Engineering, AUTH, G_CarControl.pdf*

***Abstract*—This project focuses on the design and simulation of a fuzzy logic controller (FLC) for autonomous car control. The system aims to regulate the vehicle's trajectory and orientation toward a desired position while handling nonlinearities and uncertainties. The FLC is designed with input variables corresponding to the vehicle's heading and position error, and its rule base is constructed using Mamdani inference, max aggregation, and min–max composition. Defuzzification is performed through the center of area (COA) method. Simulation results carried out in MATLAB/Simulink demonstrate the effectiveness of the proposed FLC in guiding the vehicle to the target position, even in the presence of disturbances, validating the suitability of fuzzy logic for real-world car navigation tasks.**

***Index Terms*—fuzzy logic controller, car control, intelligent systems, nonlinear dynamics, Mamdani inference, trajectory tracking.**

Design of a Fuzzy Logic Controller for Car Dynamics

The car control problem is formulated as reaching a desired position (x_d, y_d) with a specified orientation θ_d while maintaining smooth and stable motion. The control variables are defined as:

- $\underline{d_V}$: current vertical distance from the points of collision (known),
- $\underline{d_H}$: current horizontal distance from the points of collision (known),
- $\underline{\theta}$: current orientation of the cars' velocity (known),
- $\underline{\Delta\theta}$: current change in orientation of the cars' velocity. (unknown)

The fuzzy controller uses d_V, d_H, and θ as inputs, while generating the command $\Delta\theta$ as the output. The membership functions for d_V and d_H are defined over $[0, 1]$ with linguistic variables {Small, Medium, Large}, while for θ and $\Delta\theta$ are defined over $[-180°, +180°]$ and $[-130°, +130°]$, respectively, with linguistic variables {Negative, Zero, Positive}.

The rule base consists of rules of the form:

IF d_V is Small AND d_H is Small AND $\Delta\theta$ is Negative THEN θ is Positive.

This document was prepared by Dimitrios Karatis as part of the second assignment for the Simulation and Modeling of Dynamic Systems course at Aristotle University of Thessaloniki.

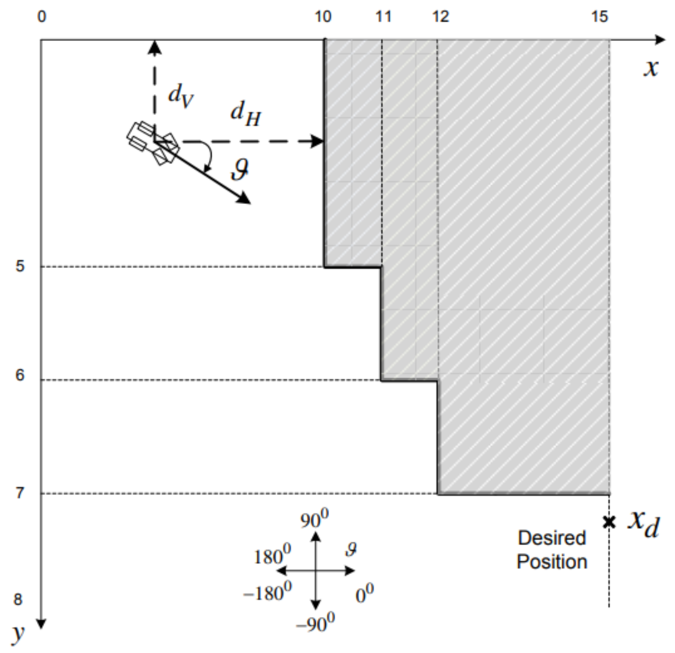


Fig. 1: The control problem

The fuzzy inference mechanism is implemented using the Mamdani approach, max aggregation, and min–max composition. Defuzzification is carried out using the center of area (COA) method. The system is tested in MATLAB using the FIS Editor, with initial conditions $x_{init} = 9.1$, $y_{init} = 4.3$ and orientation $\theta = \{0°, 45°, 90°\}$.

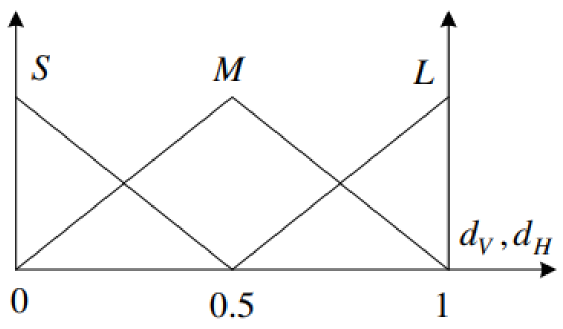


Fig. 2: Linguistic variable for the vertical (d_V) and horizontal (d_H) distance from the points of collision

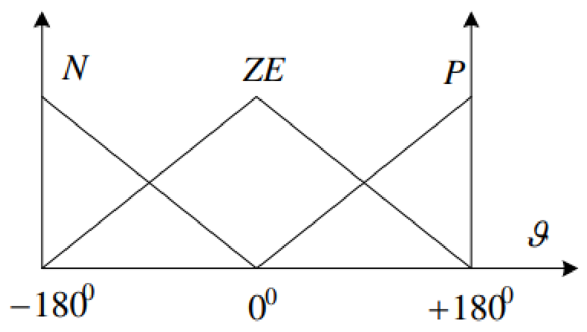


Fig. 3: Linguistic variables for the orientation of the car's velocity (θ)

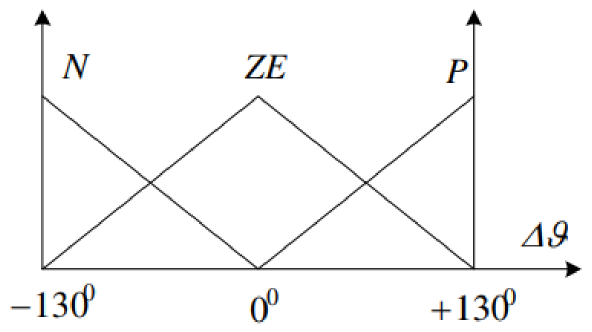
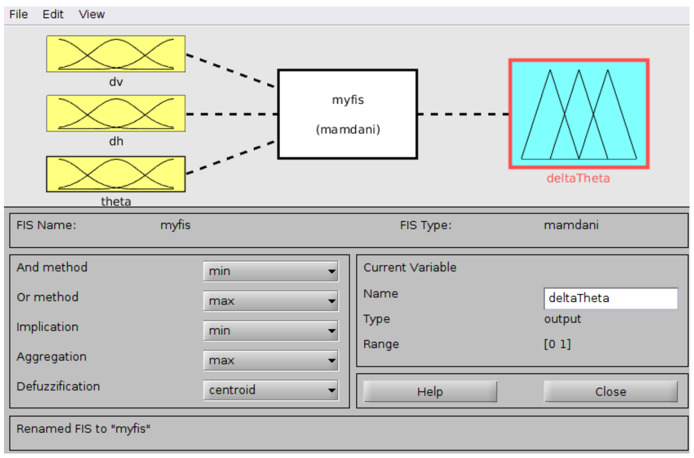


Fig. 4: Linguistic variable for the change in orientation of the car's velocity ($\Delta\theta$)

I. Initial Fuzzy Logic Controller

The first step is to construct the `Fuzzy Inference System` *(FIS)*. For this purpose, I used the `Fuzzy Logic Designer` tool provided by `MATLAB`. The figures below illustrate the process of creating the *FIS*.



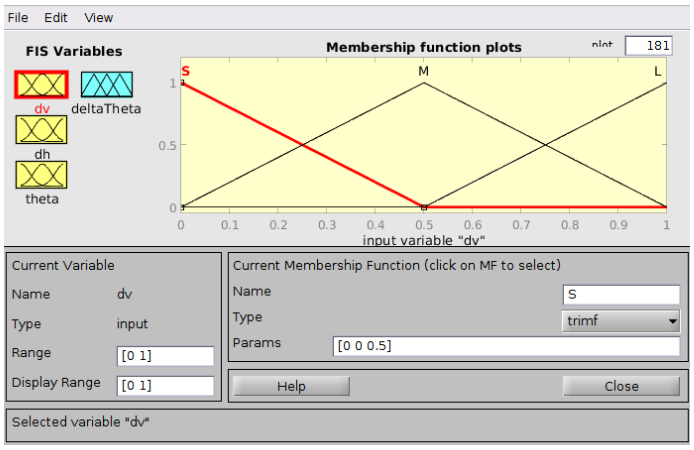
FIS Name: myfis
FIS Type: mamdani

And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: centroid

Current Variable
Name: deltaTheta
Type: output
Range: [0 1]

Renamed FIS to "myfis"

Fig. 5: The FIS, created with the help of *FuzzyLogicDesigner*

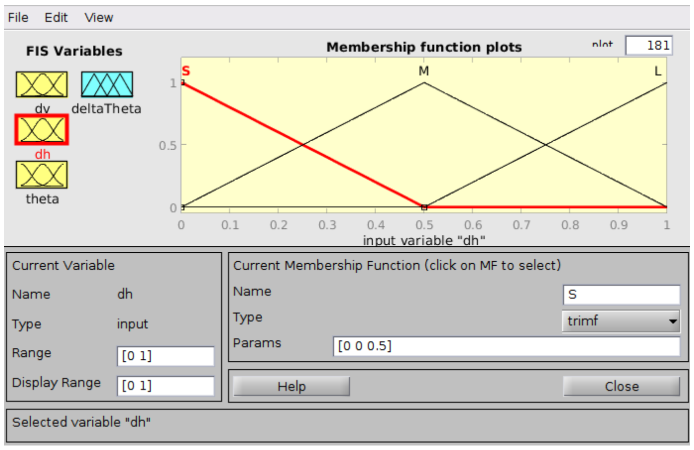


Current Variable
Name: dv
Type: input
Range: [0 1]
Display Range: [0 1]

Current Membership Function (click on MF to select)
Name: S
Type: trimf
Params: [0 0 0.5]

Selected variable "dv"

Fig. 6: Membership function of the input variable d_V

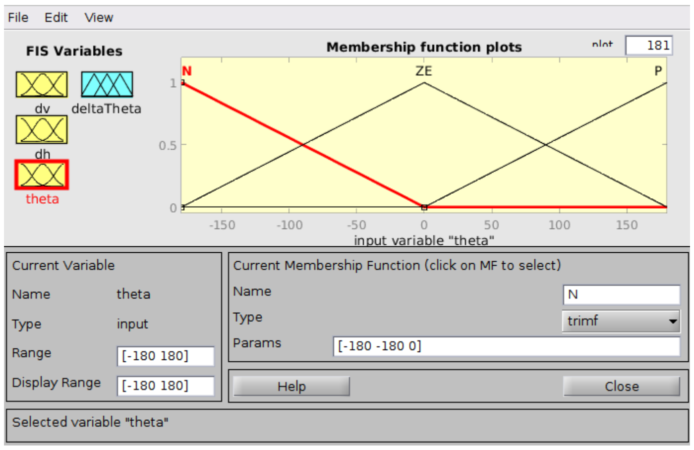


Current Variable
Name: dh
Type: input
Range: [0 1]
Display Range: [0 1]

Current Membership Function (click on MF to select)
Name: S
Type: trimf
Params: [0 0 0.5]

Selected variable "dh"

Fig. 7: Membership function of the input variable d_H

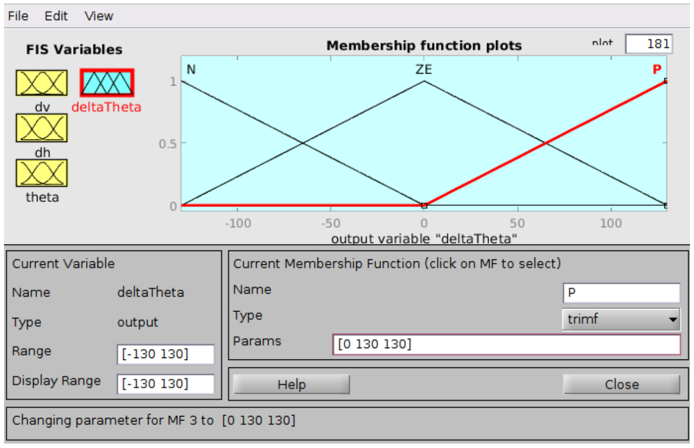


Current Variable
Name: theta
Type: input
Range: [-180 180]
Display Range: [-180 180]

Current Membership Function (click on MF to select)
Name: N
Type: trimf
Params: [-180 -180 0]

Selected variable "theta"

Fig. 8: Membership function of the input variable θ



Current Variable

Name	deltaTheta
Type	output
Range	[-130 130]
Display Range	[-130 130]

Current Membership Function (click on MF to select)

Name	P
Type	trimf
Params	[0 130 130]

Changing parameter for MF 3 to [0 130 130]

Fig. 9: Membership function of the output variable $\Delta\theta$

A. *Fuzzy Rule Base*

The complete fuzzy rule base will include one rule for each possible combination of the inputs, and given that each of the three (3) inputs is described by three (3) linguistic values, we will need $3 \times 3 \times 3 = 27$ fuzzy rules for the complete base. However, we make an important observation:

The vertical distance is not necessary (i.e., it does not need to be used) for every combination of the horizontal distance and the current angle θ, but only when the vehicle is inside the corners of the obstacle ($x \in [10, 11]$ & $y \in [5, 6]$, $x \in [11, 12]$ & $y \in [6, 7]$) as well as on the final straight line ($x \geq 12$ & $y > 7$), so that in the end it makes a straight trajectory approaching the final target.

Thus, by essentially using the verical distance d_V only when the vehicle is inside the corners of the obstacle ($x \in [10, 11]$ & $y \in [5, 6]$, $x \in [11, 12]$ & $y \in [6, 7]$) or when it reaches the final line to the target ($x \geq 12$ & $y > 7$) we would have the following base of fuzzy rules:

1) If (d_V is S) and (d_H is S) and (θ is N) then ($\Delta\theta$ is ZE)
2) If (d_V is S) and (d_H is S) and (θ is ZE) then ($\Delta\theta$ is N)
3) If (d_V is S) and (d_H is S) and (θ is P) then ($\Delta\theta$ is N)
4) If (d_V is S) and (d_H is M) and (θ is N) then ($\Delta\theta$ is P)
5) If (d_V is S) and (d_H is M) and (θ is ZE) then ($\Delta\theta$ is N)
6) If (d_V is S) and (d_H is M) and (θ is P) then ($\Delta\theta$ is N)
7) If (d_V is S) and (d_H is L) and (θ is N) then ($\Delta\theta$ is P)
8) If (d_V is S) and (d_H is L) and (θ is ZE) then ($\Delta\theta$ is ZE)
9) If (d_V is S) and (d_H is L) and (θ is P) then ($\Delta\theta$ is N)
10) If (d_V is M) and (d_H is S) and (θ is N) then ($\Delta\theta$ is ZE)
11) If (d_V is M) and (d_H is S) and (θ is ZE) then ($\Delta\theta$ is N)
12) If (d_V is M) and (d_H is S) and (θ is P) then ($\Delta\theta$ is N)
13) If (d_V is M) and (d_H is M) and (θ is N) then ($\Delta\theta$ is P)
14) If (d_V is M) and (d_H is M) and (θ is ZE) then ($\Delta\theta$ is ZE)
15) If (d_V is M) and (d_H is M) and (θ is P) then ($\Delta\theta$ is N)
16) If (d_V is M) and (d_H is L) and (θ is N) then ($\Delta\theta$ is P)
17) If (d_V is M) and (d_H is L) and (θ is ZE) then ($\Delta\theta$ is ZE)
18) If (d_V is M) and (d_H is L) and (θ is P) then ($\Delta\theta$ is N)
19) If (d_V is L) and (d_H is S) and (θ is N) then ($\Delta\theta$ is ZE)
20) If (d_V is L) and (d_H is S) and (θ is ZE) then ($\Delta\theta$ is N)
21) If (d_V is L) and (d_H is S) and (θ is P) then ($\Delta\theta$ is N)
22) If (d_V is L) and (d_H is M) and (θ is N) then ($\Delta\theta$ is P)
23) If (d_V is L) and (d_H is M) and (θ is ZE) then ($\Delta\theta$ is ZE)
24) If (d_V is L) and (d_H is M) and (θ is P) then ($\Delta\theta$ is N)
25) If (d_V is L) and (d_H is L) and (θ is N) then ($\Delta\theta$ is P)
26) If (d_V is L) and (d_H is L) and (θ is ZE) then ($\Delta\theta$ is ZE)
27) If (d_V is L) and (d_H is L) and (θ is P) then ($\Delta\theta$ is N)

The linguistic variables are interpreted as follows:

- $d_H = S$: When d_H is small, we need to be careful not to hit any obstacles while also going the right way.
- $d_H = M$ or $d_H = L$: When d_H is medium or large, I want the vehicle to start approaching the obstacle vertically, so it can track it better.

When the distance d_H is small ($d_H = S$) and the angle θ is positive (P), meaning that the vehicle is headed upward, then the change in angle $\Delta\theta$ must be negative (N) in order to change the direction of velocity, and also to not hit any obstacle. If $\theta = 0$ (ZE), then again $\Delta\theta$ must be negative (N) so that the vehicle continues moving downward. In the case where θ is negative (N), then $\Delta\theta$ should be zero (ZE) in order to maintain the same negative direction.

If $d_H = M$ or $d_H = L$, then the vehicle is relatively far from the obstacle. In these cases, the goal is to approach the obstacle first and only then start turning when d_H beacomes small ($d_H = S$). Therefore, if θ is positive (P), we require $\Delta\theta$ to be negative (N) so that the heading moves closer to $\theta = 0$. Conversely, if θ is negative (N), then $\Delta\theta$ must become positive (P) for the exact same reason. Finally, if $\theta = 0$ (ZE), then we want $\Delta\theta = 0$ (ZE) in order to remain in the same direction, which is essentially perpendicular to the obstacle.

The variable d_V influences two main situations:

1) It is used to ensure that the vehicle does not get too close to vertical obstacles ($\rightarrow$ Rule 5).
2) It ensures that, once the vehicle reaches the final straight line, it follows it ($\rightarrow$ Rule 8).

In general, these rules are designed so that the vehicle adapts its motion near the walls, essentially following them quite closely. This is why d_V was not used extensively (almost all rules with the same d_H are identical regardless of d_V except for Rule 5). Otherwise, we could minimize the distance the vehicle needs to travel by directing it toward the hypotenuse of the triangles, which would require much more frequent use of d_V. However, such an approach was not implemented in the present analysis.

The above fuzzy rule base is can be found in the `myfis.fis` *file.*

II. FLC Simulation

A. Distance Sensors Design

As mentioned above, the distance sensors read within the range θ from 0 to 1 m, while the actual range of horizontal distance values is from 0 to 15 m and the vertical distance values from 0 to 7 m. For this reason, the sensors that read the above distances are saturated for distances greater than 1 m.

Thus, when a sensor gives a distance of 1, it means a distance ≥ 1 m.

The function `distance_sensors(x, y)` computes the vertical distance d_V and the horizontal distance d_H from obstacles on a 2D map. The algorithm divides the map into four regions according to the value of x. For each region, d_V is defined as the difference between a reference boundary and the coordinate y, while d_H is determined as the difference between a boundary and the coordinate x. Finally, both distances are bounded above by 1, i.e.

$$d_V = \min(d_V, 1), \quad d_H = \min(d_H, 1),$$

which ensures that neither distance exceeds the maximum allowed values.

Lastly, in this project, the vertical axis y of the map was defined as $y < 0$ (rather than $y > 0$ in the original formulation), so that the vertical distance always satisfies $d_V \geq 0$, based on Fig. 1. Meaning that all of the positive y coordinates of the original formulation are now negative, and so the point $x_{init} = 9.1$, $y_{init} = 4.3$ actually becomes $x_{init} = 9.1$, $y_{init} = -4.3$.

Detailed code can found in the `distance_sensors.m` file.

B. Control Simulation Design

The simulation evaluates the trajectory of a car controlled by a fuzzy logic controller (FLC). For each initial orientation θ in the set of angles $\{\theta_i\}$, the car starts from the initial position $(x_{\text{init}}, y_{\text{init}})$. At each time step, the distances to obstacles (d_V, d_H) are computed via the distance sensor function and combined with the current orientation θ as inputs to the fuzzy inference system `carFIS`. The FLC outputs a correction $\Delta\theta$, which is used to update the heading:

$$\theta \leftarrow \theta + \Delta\theta.$$

The car then moves forward with a step size u:

$$x \leftarrow x + u\cos(\theta), \quad y \leftarrow y + u\sin(\theta).$$

The process repeats until the car either leaves the map boundaries

$$0 \leq x \leq 15, \quad -8 \leq y \leq 0,$$

or reaches a neighborhood of the target position $(x_{\text{desired}}, y_{\text{desired}}) = (15, -7.2)$ within a distance less than the specified threshold.

The control simulation of the FLC can be found in the `car_controller.m` file.

C. Results

The results for the above rule base with the Membership Functions (MF) proposed by the project's formulation (Figures 2-4) for the different values of the angle $\theta = \{0°, 45°, 90°\}$ are:

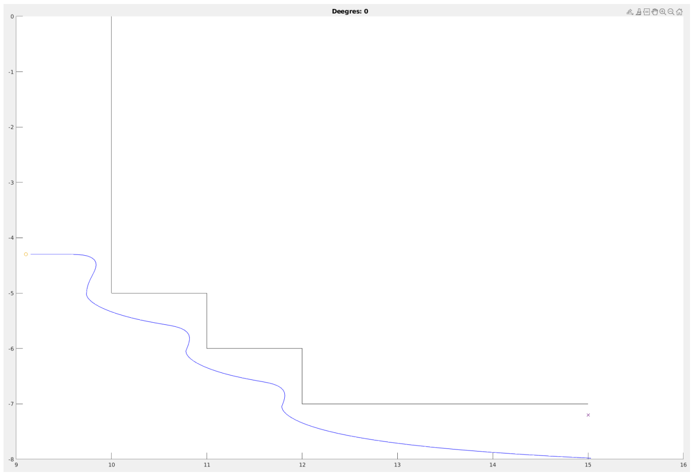


Fig. 10: Results when using the proposed MF for $\theta = 0°$

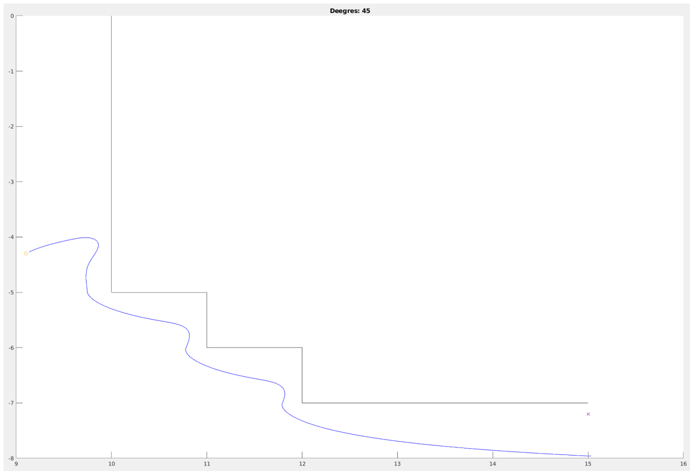


Fig. 11: Results when using the proposed MF for $\theta = 45°$

From the simulation results, it seems that in all cases the vehicle follows the edges of the map quite well, avoiding collisions while maintaining a reasonable distance from the boundaries. However, the car does not succeed in reaching the desired target point exactly. In particular, while the final x-coordinate is very close to the desired value x_{desired}, the y-coordinate consistently shows a larger deviation.

For example, with initial orientations of 0°, 45°, and 90°, the final positions were

$$(15.032074, -7.981067), \quad (15.033120, -7.961499),$$

$$(15.030353, -7.961212),$$

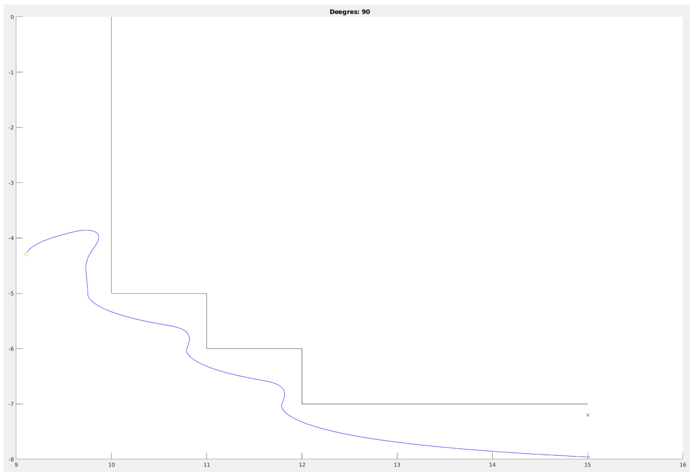


Fig. 12: Results when using the proposed MF for $\theta = 90°$

respectively, while the desired target was

$$(15.00, -7.20).$$

Among these cases, $\theta = 0°$ resulted in the largest deviation along the y-axis at -7.981067, while $\theta = 90°$ achieved the smallest deviation along the y-axis at -7.961212. In terms of the x-axis, $\theta = 90°$ gave the smallest error with $x = 15.030353$, whereas $\theta = 45°$ produced the largest error with $x = 15.033120$.

This indicates that the controller is able to maintain accurate horizontal alignment but has more difficulty in precisely converging vertically to the target position.

For this reason, the Membership Functions of the Fuzzy Logic Controller (FLC) were modified. The tuning process was carried out by `trial and error`, testing different configurations of MFs until a more satisfactory performance was achieved.

III. Final Fuzzy Logic Controller

Initially, the fuzzy logic controller used simple triangular membership functions for all inputs and output, **with broad overlapping ranges**. Specifically, the vertical distance (d_V) and horizontal distance (d_H) were each divided into three triangular MFs spanning $[0, 0, 0.5]$, $[0, 0.5, 1]$, and $[0.5, 1, 1]$, while the heading angle (θ) and the output $\Delta\theta$ used wide triangular functions covering nearly the entire input range. While this configuration allowed the vehicle to navigate the map and avoid obstacles, the simulation results showed significant deviation from the desired target, especially in the vertical direction.

To improve the vehicle's precision in reaching the desired position, the membership functions were **adjusted to be sharper and slightly shifted**. The vertical and horizontal distances (d_V and d_H) were redefined to have narrower triangular ranges, increasing the controller's sensitivity to small deviations. The heading angle (θ) and output ($\Delta\theta$) were changed to include trapezoidal shapes at the extremes, providing stronger corrective action when the heading was significantly misaligned. This "crispier" MF design allows the fuzzy logic controller to generate more precise steering adjustments, especially near the desired target. Below we can see the new Membership Functions:

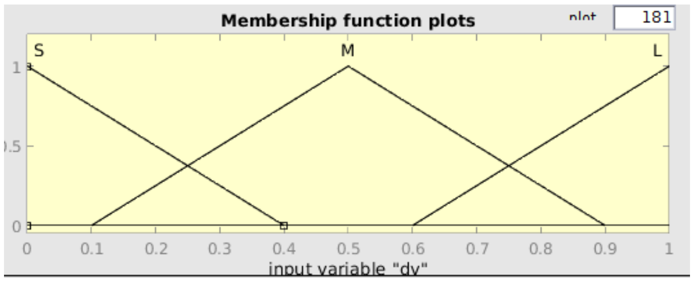


Fig. 13: New Membership function of the input variable d_V

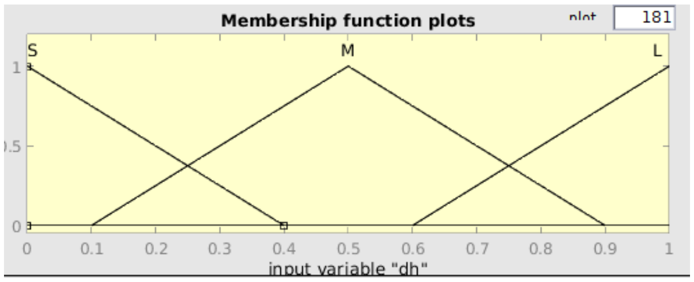


Fig. 14: New Membership function of the input variable d_H

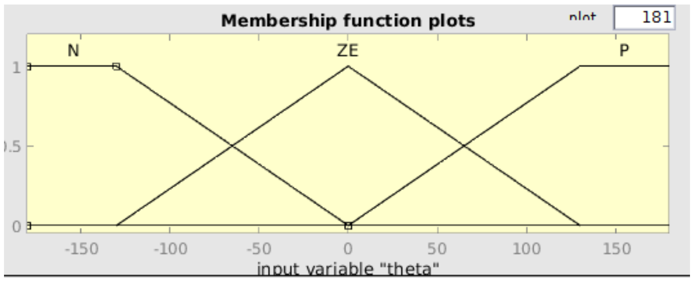


Fig. 15: New Membership function of the input variable θ

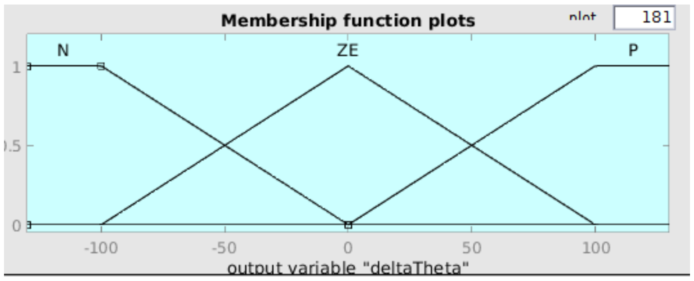
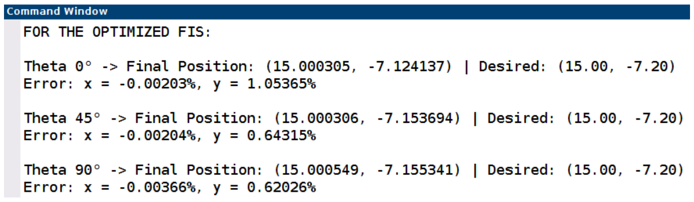


Fig. 16: New Membership function of the output variable $\Delta\theta$

The new optimized Fuzzy Logic Interface (FIS) can be found in the `myfis_optimized.fis` file.

As a result of these optimizations, the vehicle reached positions much closer to the target. For the same initial orientations of 0°, 45°, and 90°, the final x-coordinates were essentially perfect (errors less than 0.004%), and the y-coordinates improved dramatically, with errors between 0.62% and 1.05%.



```
FOR THE OPTIMIZED FIS:

Theta 0° -> Final Position: (15.000305, -7.124137) | Desired: (15.00, -7.20)
Error: x = -0.00203%, y = 1.05365%

Theta 45° -> Final Position: (15.000306, -7.153694) | Desired: (15.00, -7.20)
Error: x = -0.00204%, y = 0.64315%

Theta 90° -> Final Position: (15.000549, -7.155341) | Desired: (15.00, -7.20)
Error: x = -0.00366%, y = 0.62026%
```

Fig. 17: Results when using the optimized MFs

TABLE I: Comparison of Final Positions and Errors: Initial vs Optimized FIS

Theta	FIS Type	Final X	Final Y	Desired X	Desired Y	Error (X%, Y%)
0°	Initial	15.032074	-7.981067	15.00	-7.20	-0.21383%, -10.84815%
0°	Optimized	15.000305	-7.124137	15.00	-7.20	-0.00203%, 1.05365%
45°	Initial	15.033120	-7.961499	15.00	-7.20	-0.22080%, -10.57637%
45°	Optimized	15.000306	-7.153694	15.00	-7.20	-0.00204%, 0.64315%
90°	Initial	15.030353	-7.961212	15.00	-7.20	-0.20235%, -10.57239%
90°	Optimized	15.000549	-7.155341	15.00	-7.20	-0.00366%, 0.62026%

In addition, the vehicle appears to track the borders of the obstacles more effectively. Compared to the previous FIS, it now approaches the obstacles more closely while maintaining a safe distance, demonstrating better spatial awareness. This indicates that the optimized Membership Functions not only enhanced accuracy in reaching the desired target but also improved the controller's ability to navigate the environment effectively.

Overall, the car achieves a more precise trajectory, balancing both **target convergence** and **obstacle avoidance**. Below we can see the results of this modification in action:

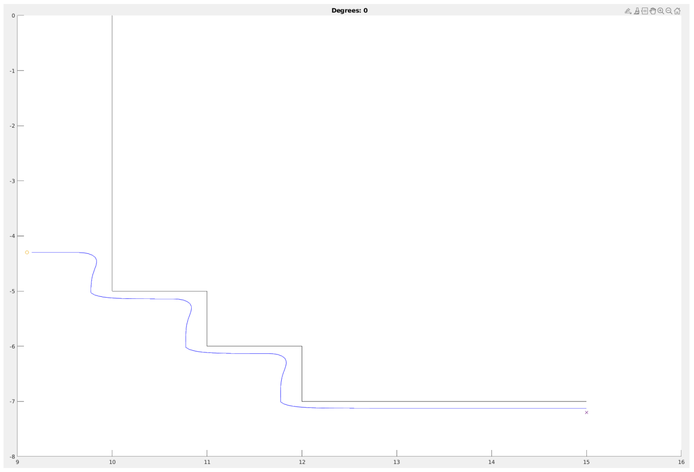


Fig. 18: Optimized results when using the proposed MF for $\theta = 0°$

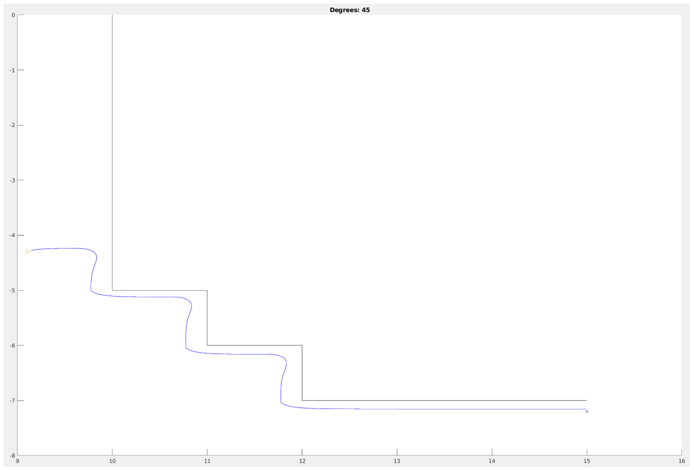


Fig. 19: Optimized results when using the proposed MF for $\theta = 45°$

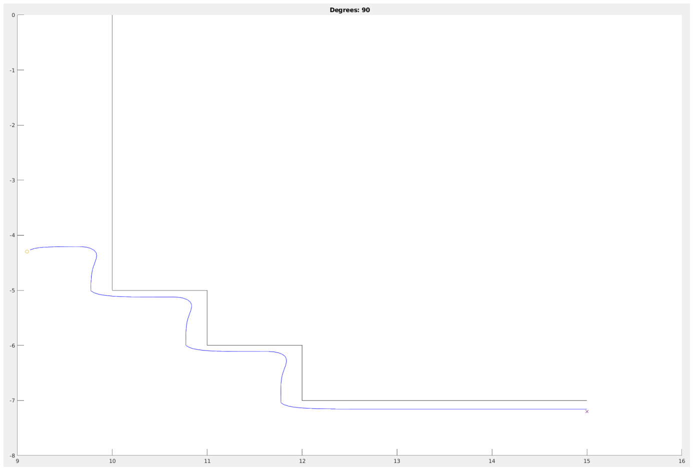


Fig. 20: Optimized results when using the proposed MF for $\theta = 90°$

The control simulation of the new optimized FLC can be found in the `car_controller_optimized.m` *file.*