Fuzzy Robot Control Project

Dhruv Mehta

Table of Contents

Table of Figures

Table of Tables

Introduction

Fuzzy logic is a method of converting inputs into linguistic variables which is more understandable for a human being. Each input is mapped into fuzzy sets and a fuzzy inference engine is made which consists of a fuzzy rule base that drives the system.

For the current project, a robot motion is controlled using fuzzy logic in MATLAB mainly using fuzzy logic toolbox. Robot meets the specified goals within the specified radius.

Mamdani fuzzy inference system has been used controlling the robot's movement.

Each input is fuzzified and mapped into 7 fuzzy sets and the corresponding outputs have also been mapped into 7 fuzzy sets.

1. Fuzzifying the inputs

Parameters or outputs being controlled are:

- Radial Acceleration
- Rotational Acceleration

Inputs to the system are:

- Distance scaled by the size of the arena
- Relative Angle between the bot and goal
- Radial Velocity
- Angular Velocity

Below shown is the basic schematic of the system:

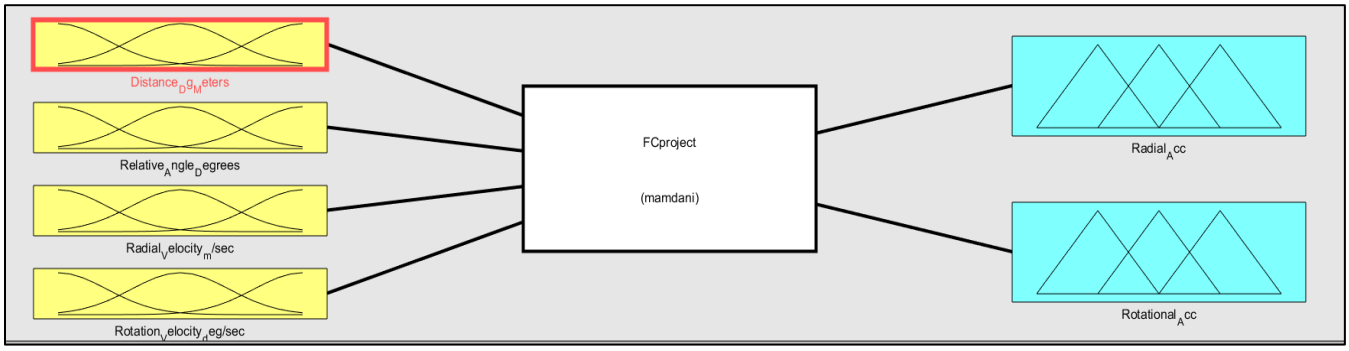


Figure 1: Schematic of Fuzzy Control System

1. Plotting fuzzified input for (ϕ_g, d_g, ϕ, $\dot{r}$) based on the errors

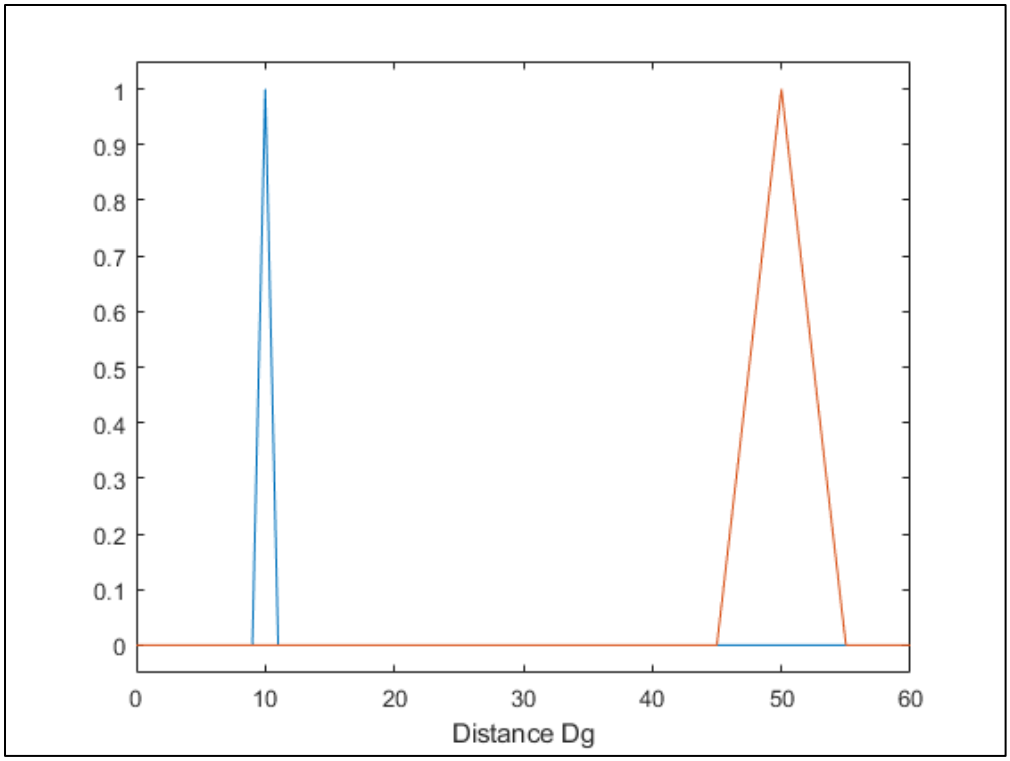


Figure 2: Fuzzified Input for Distance

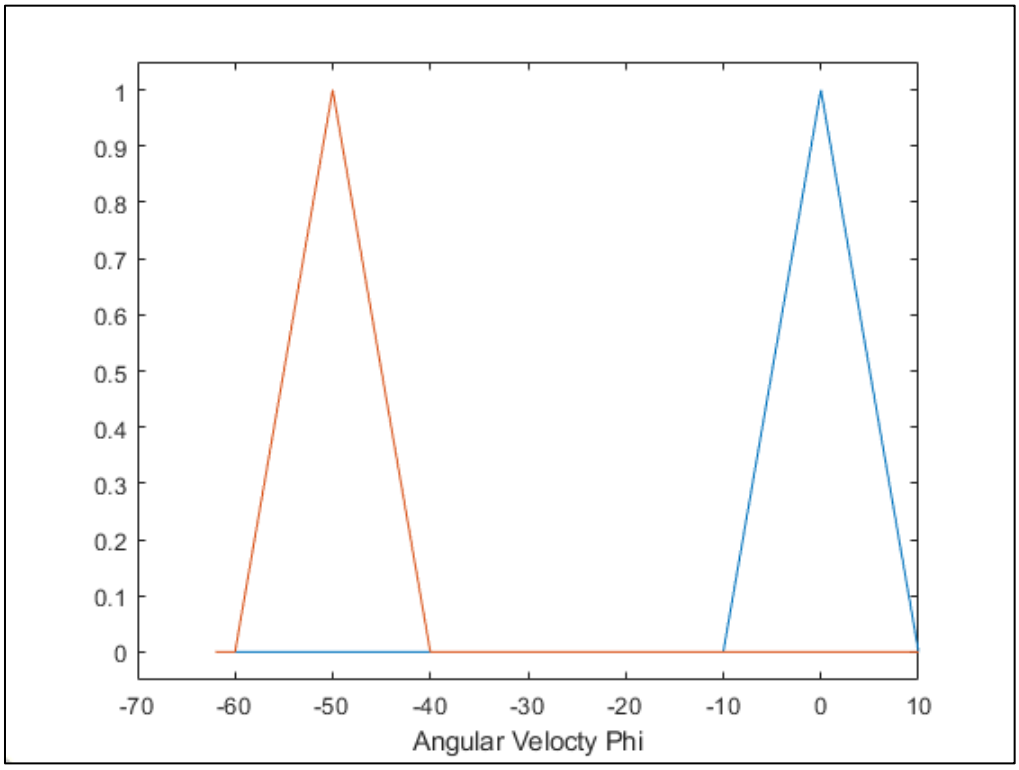


Figure 3: Fuzzified Input for Anglular Velocity

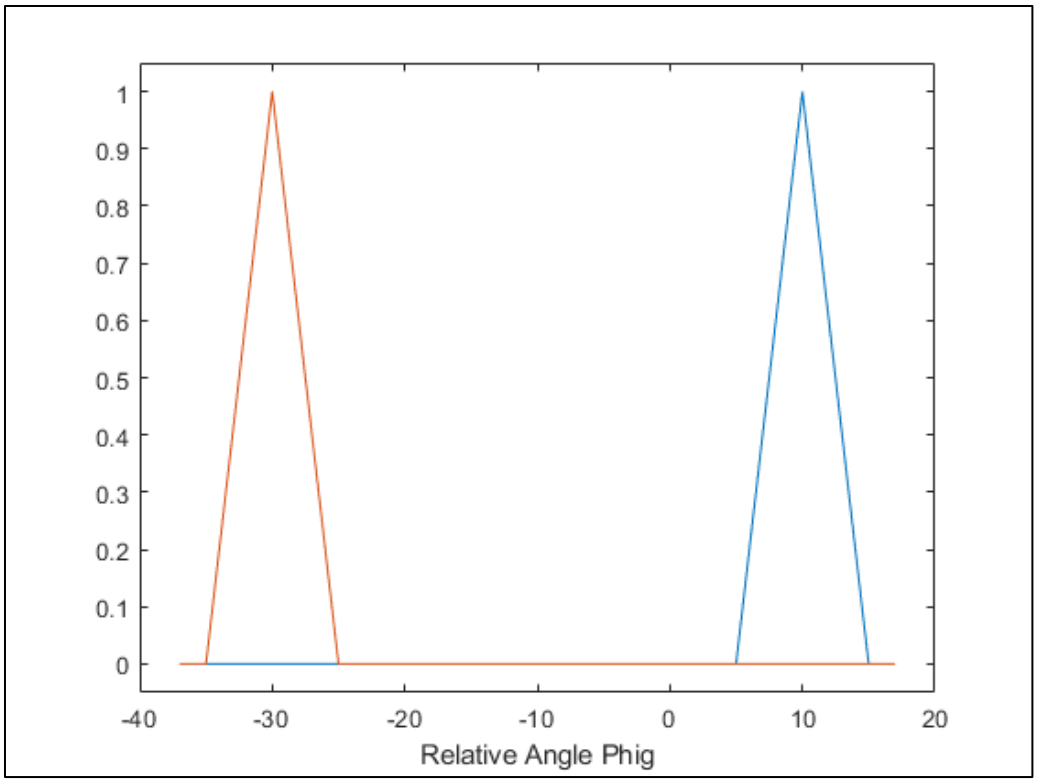


Figure 4: Fuzzified Input for Relative Angle

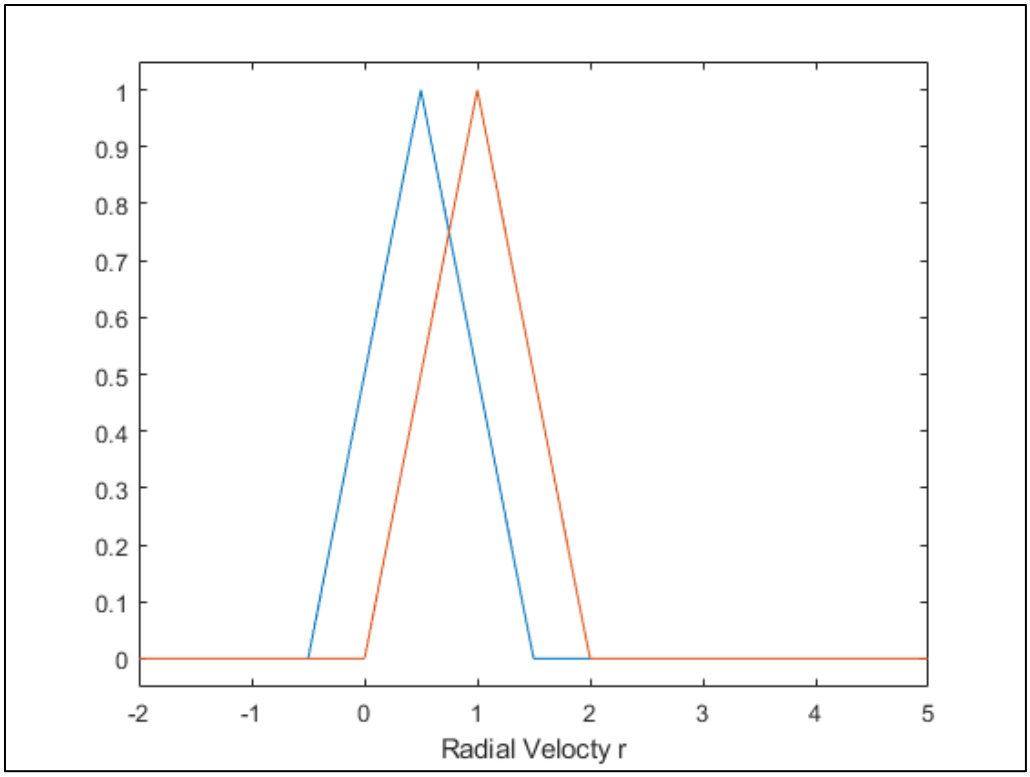


Figure 5: Fuzzified Input for Radial Velocity

1a) (ϕ_g, d_g, ϕ, $\dot{r}$) = (10,10,0,0.5) : Robot will travel a distance of about 12 m with velocity of about 0.8 m/s and firing strength if 0.7

1b) (ϕ_g, d_g, ϕ, $\dot{r}$) = (-30, 50, -50, 1) : Robot will travel a distance of about 50 m with a membership value of 1 as the distance is very large.

2. Mapping each Input Space to 7 Fuzzy Sets

Below shown are the plots of 4 input variables:

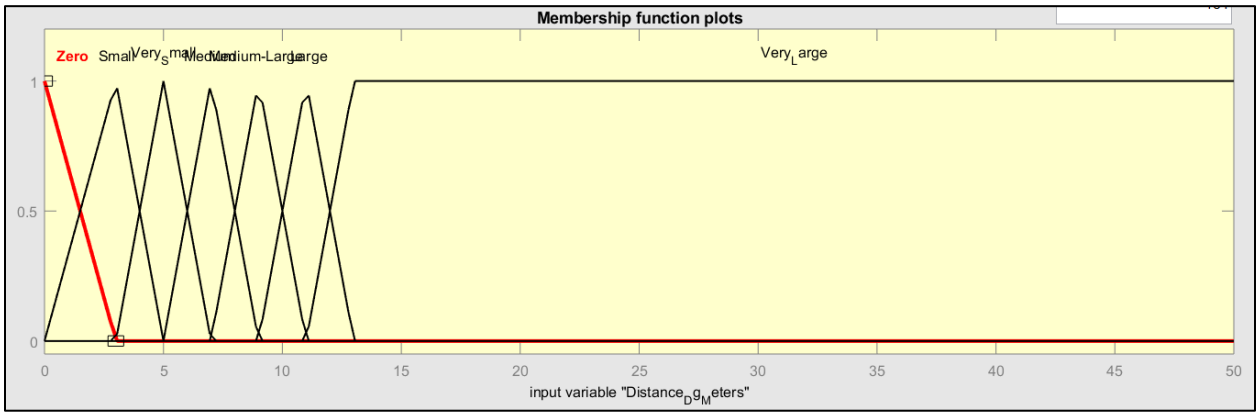


Figure 6: Distance Plot

The distance has been mapped into 7 fuzzy sets of Zero, Very Small, Small , Medium, Medium-Large, Large and Very Large.

Distance has been scaled based on the size of arena and most of the distance is within 15 m where the acceleration is zero for a more stable robot.

For relative angle:

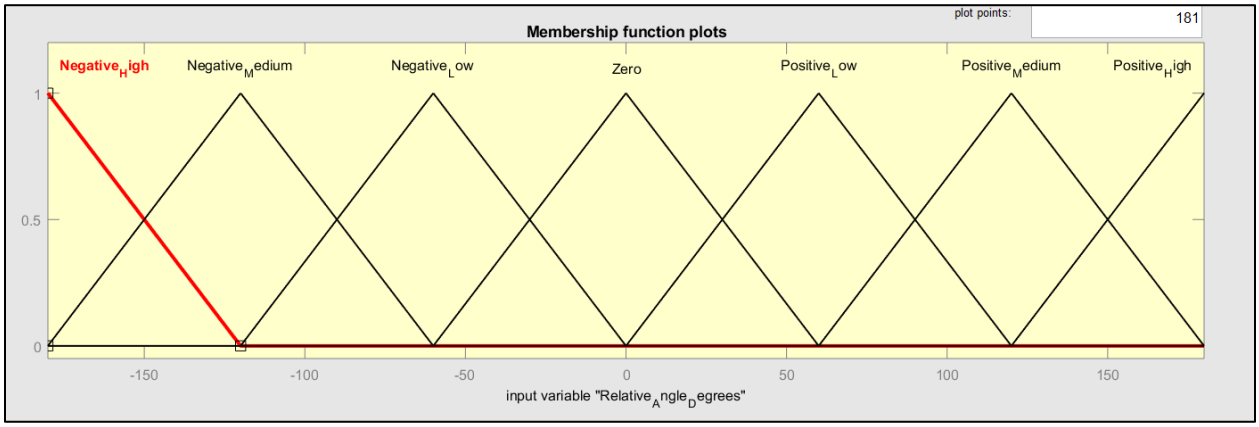


Figure 7: Relative Angle Plot

Relative angle has been mapped into 7 equally spaced intervals from -180^0 to 180^0. Negative High, Negative Medium, Negative Low, Zero, Positive Low, Positive Medium & Positive High.

For Radial Velocity:

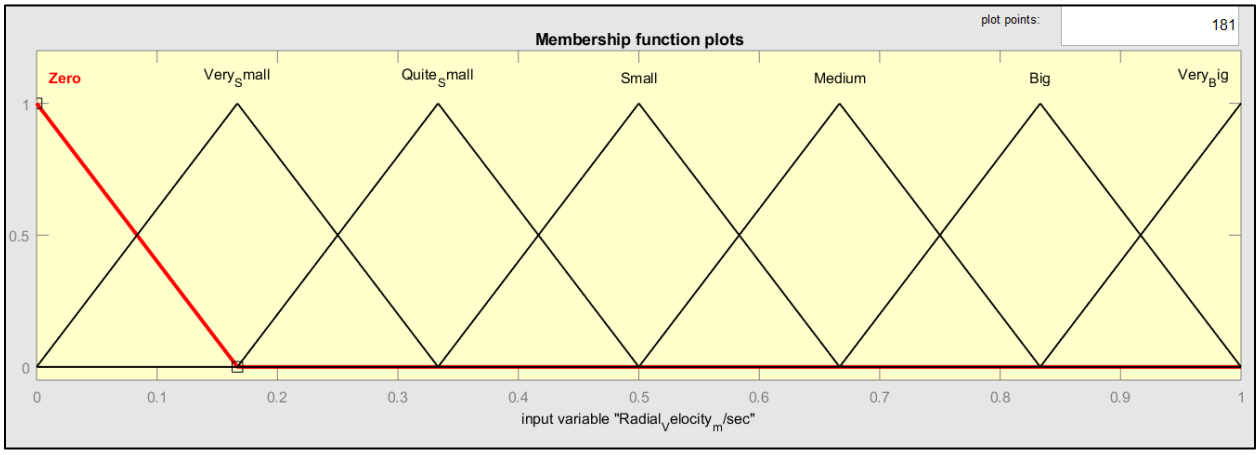


Figure 8: Radial Velocity Plot

Radial velocity has been plot into 7 equally divided membership functions from [0,1] m/s. Zero, Very Small, Quite Small, Small, Medium, Big & Very Big.

For Rotational Velocity:

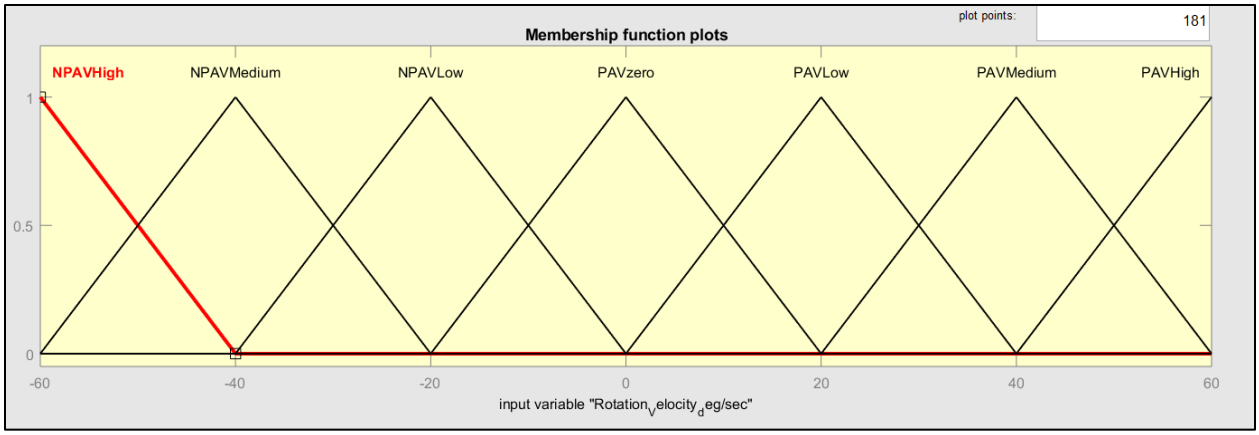


Figure 9: Rotational Velocity Plot

Rotational velocity has been plot into 7 equally divided membership functions from [-60, 60] deg/sec. NPAV High, NPAV Medium, NPAV Low, PAV Zero, PAV Low, PAV Medium and PAV High.

3. Mapping each Output Space to 7 Fuzzy Sets

Like the input parameters, each output has been mapped into 7 triangular membership functions.

Below shown are the respective plots.

For Radial Acceleration:

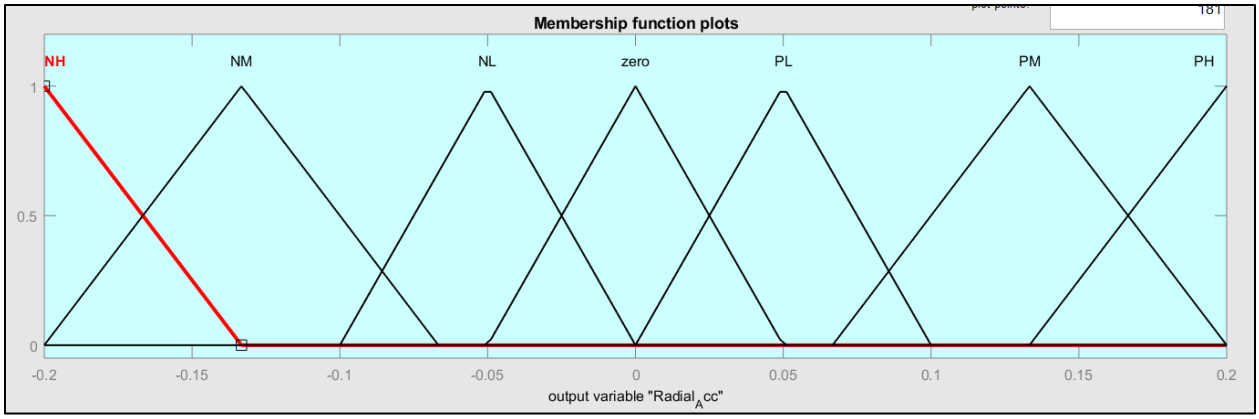
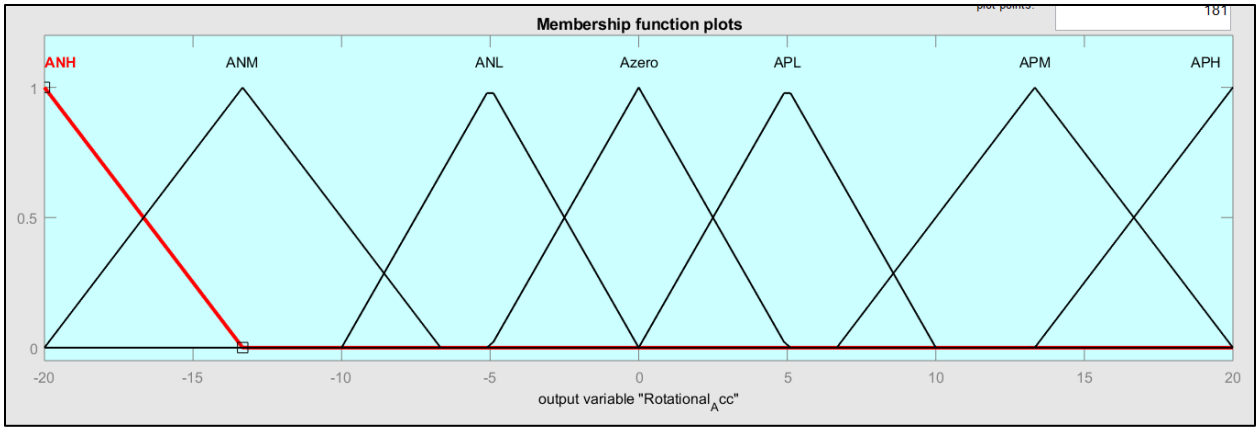


Figure 10: Radial Acceleration Plot

Radial acceleration has been divided into 7 membership functions which are not equally spaced. The zero, NL and PL are kept close for an optimized performance. The others are NH, NM, PM & PH each indicating negative and positive direction respectively.

For Rotational Acceleration:



Rotational acceleration has been divided similar to the radial acceleration functions with a range from 20 to 20 deg/sec^2. ANH, ANM, ANL, Azero, APL, APM and APH each indicating negative and positive direction respectively.

4. Fuzzy Rule Base or Fuzzy Inference Engine

The fuzzy rules are designed based on a stable robot movement behavior much before it reaches very close to goal. That is, having relative angle as 0 and applying negative acceleration in case of high velocities before it reaches the goal to make it stop right when it's about zero.

Some rules have been shown below:

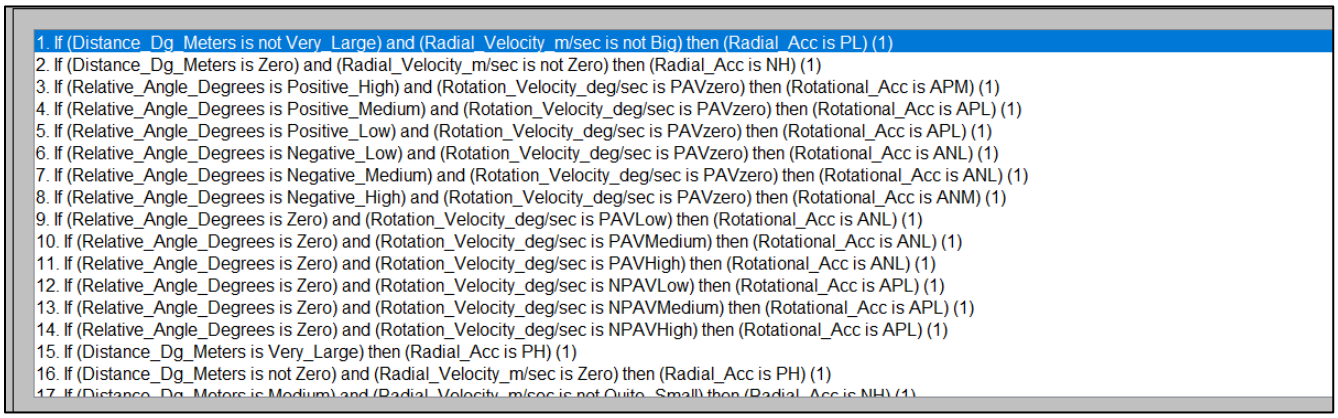


Figure 11: Fuzzy Rule Base

As shown, if the distance is very large then we have positive acceleration and once it reaches a lesser distance, the acceleration is zero travelling with constant velocity. For relative angle, it has a positive or negative angular acceleration based on the orientation of the robot but also has a negative acceleration applied in case of negative relative angle desired but angular velocity is positive, hence stabilizing the robot.

Total No. of Rules = 17

5. Defuzzifier

I've chosen centroid defuzzification method here. It basically aggregates the minimum of the antecedent and consequent, then calculates the center of the area and gives out a crisp output which is necessary for control.

- It is computationally inexpensive
- Less variation in outputs when input parameters are changed

Below shown is an example of relation based on surface plots.

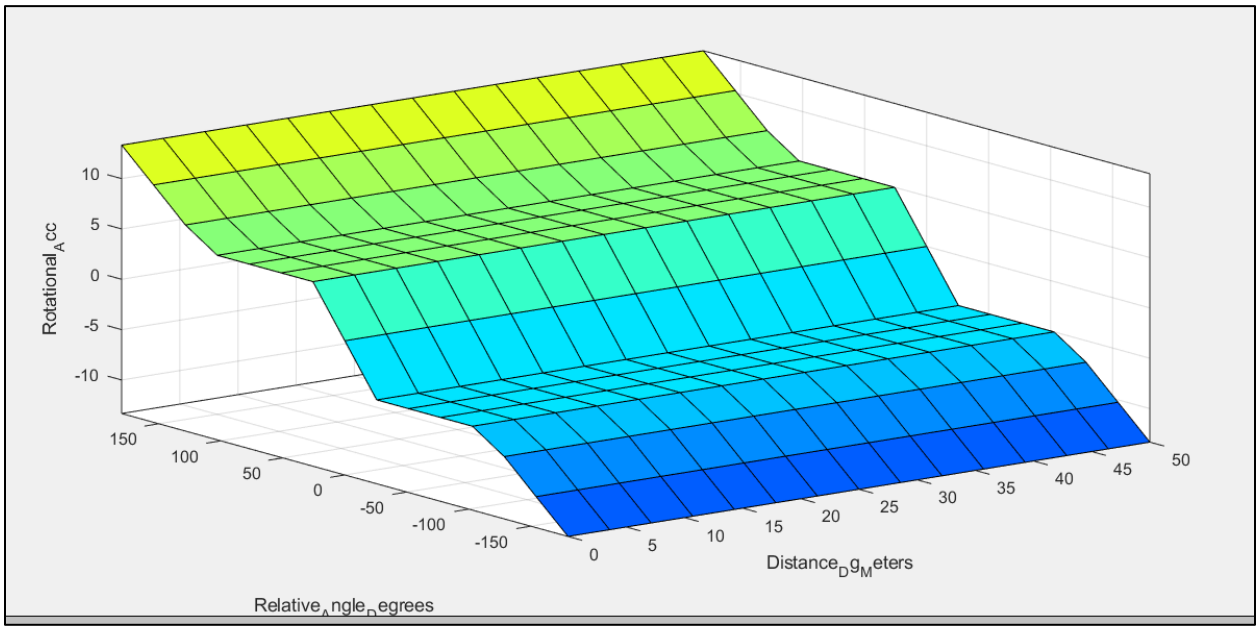


Figure 12: Surface Plot of 3 Inputs

As seen the system calculates a value based on the input of distance and relative angle giving the corresponding output.

Another example between 1 input and 1 output:

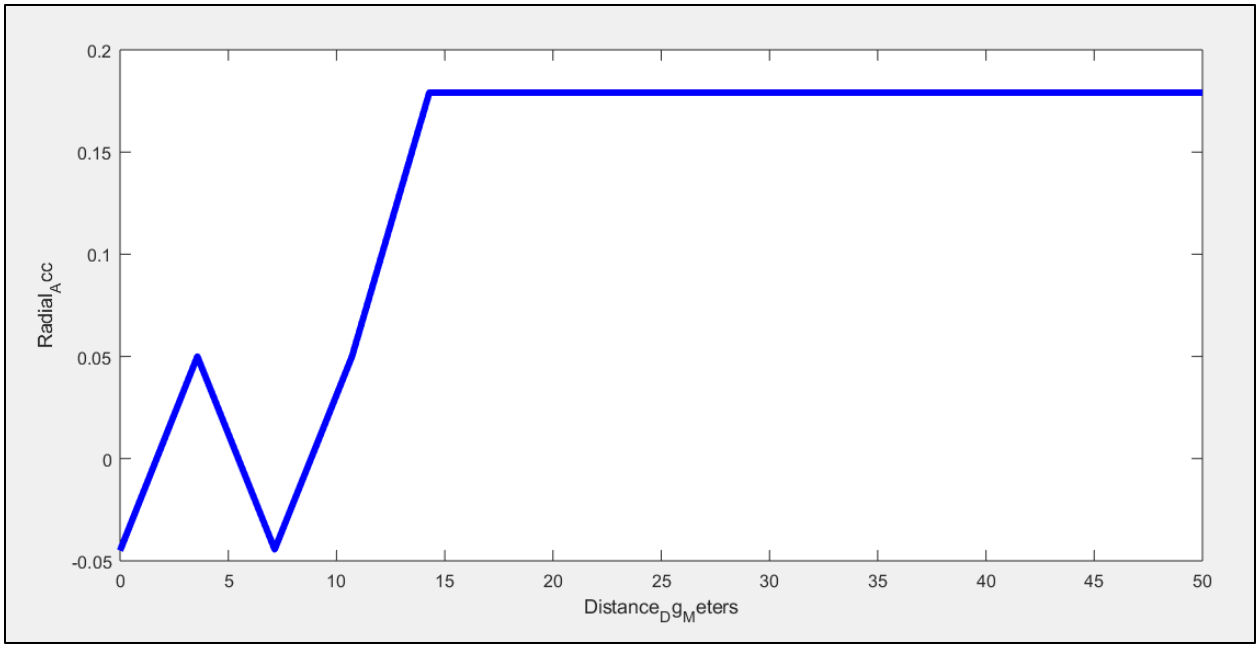


Figure 13: Radial Acceleration and Distance Plot

6. Robot Movement Plot

Below shown are the robot movement plots after implementing the fuzzy controller.

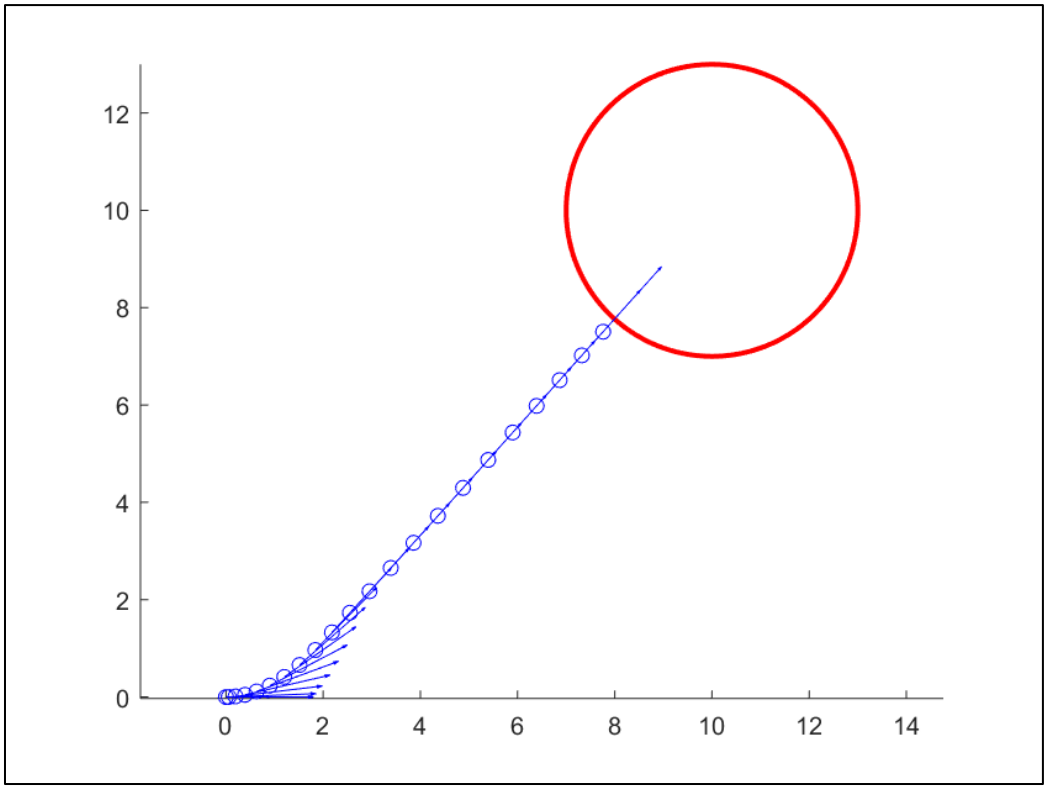


Figure 14: Scenario 1

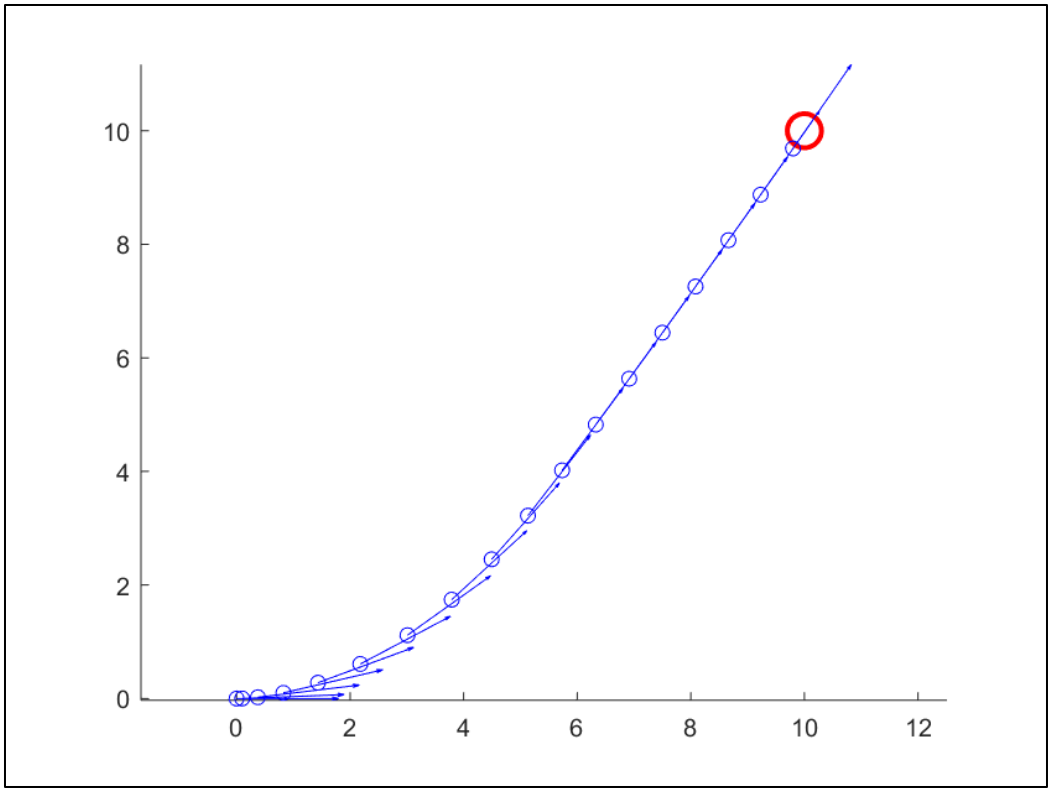


Figure 15: Scenario 2

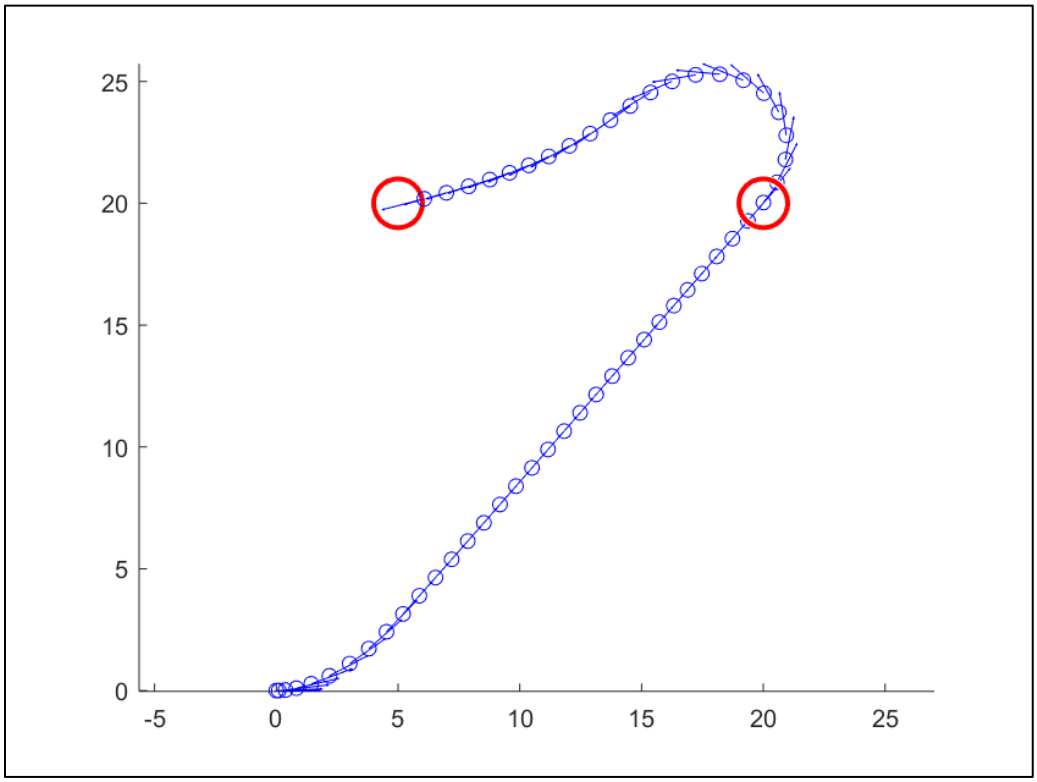


Figure 16: Scenario 3

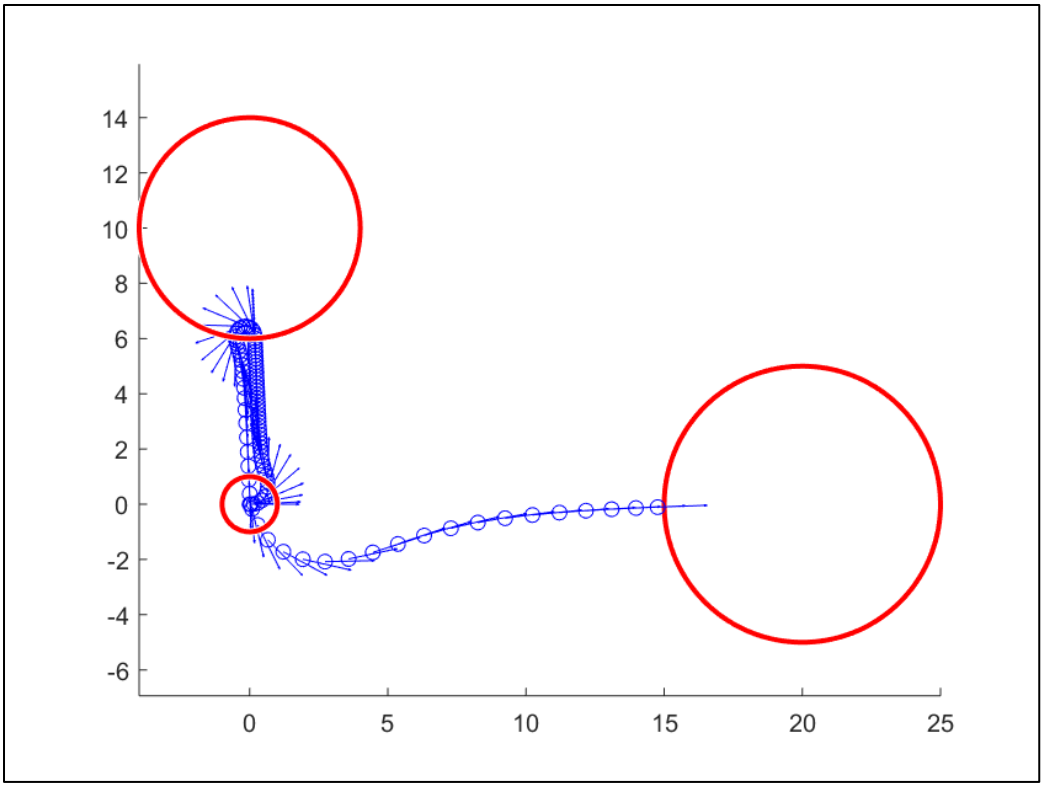


Figure 17: Scenario 4

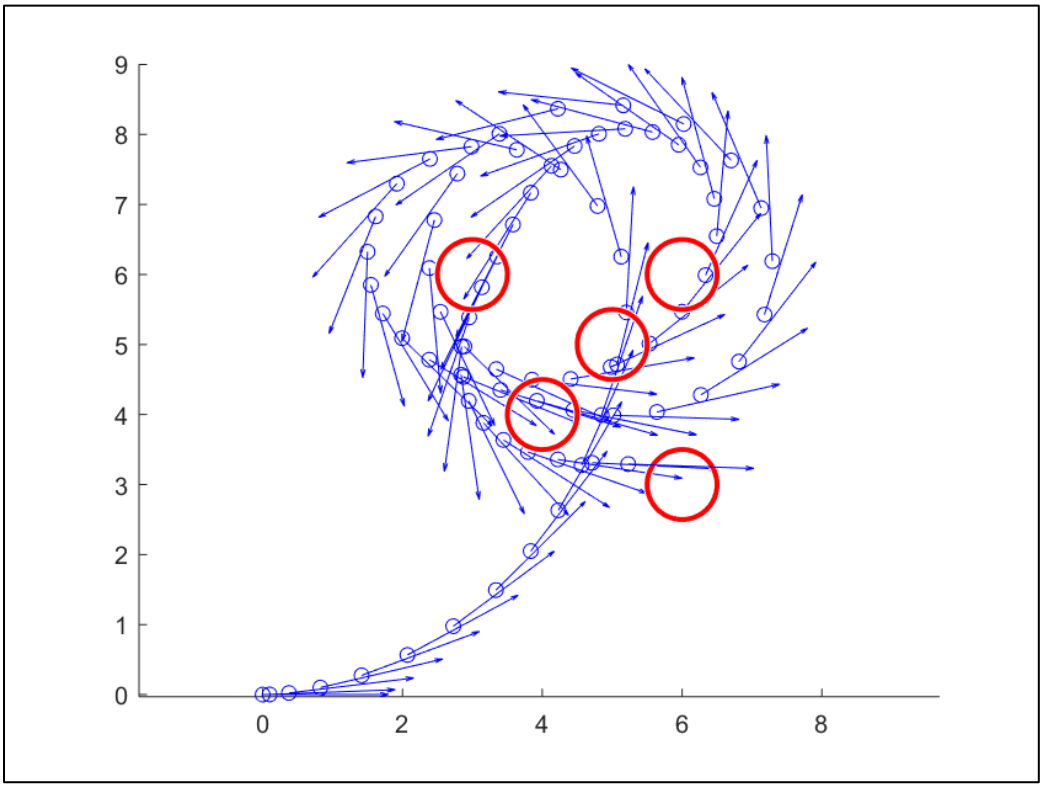


Figure 18: Scenario 5

Table 1: Comparison of Basic Controller and Designed Controller

Question Number	Goal and Radius	Basic Controller (Time Steps)	Designed Controller (Time Steps)
6a	(10,10) radius 3	166	217
6b	(10,10) radius 0.3	551	172
6c	(20,20) radius 1; (5,20) radius 1	622	533
6d	(0,10) radius 4; (0,0) radius 1; (20,0) radius 5	533	913
6e	(5,5) radius 0.5; (4,4) radius 0.5; (6,6) radius 0.5; (3,6) radius 0.5; (6,3) radius 0.5	1227	779

To get this robot working, I had to add a rule for stabilization that if the distance is Medium and Radial Velocity if Not Quite Small Then Radial Acceleration is Negative High and for Large Distances I had to change my radial acceleration from positive medium to positive low otherwise scenario 5 was not achieved as shown in the figure below.

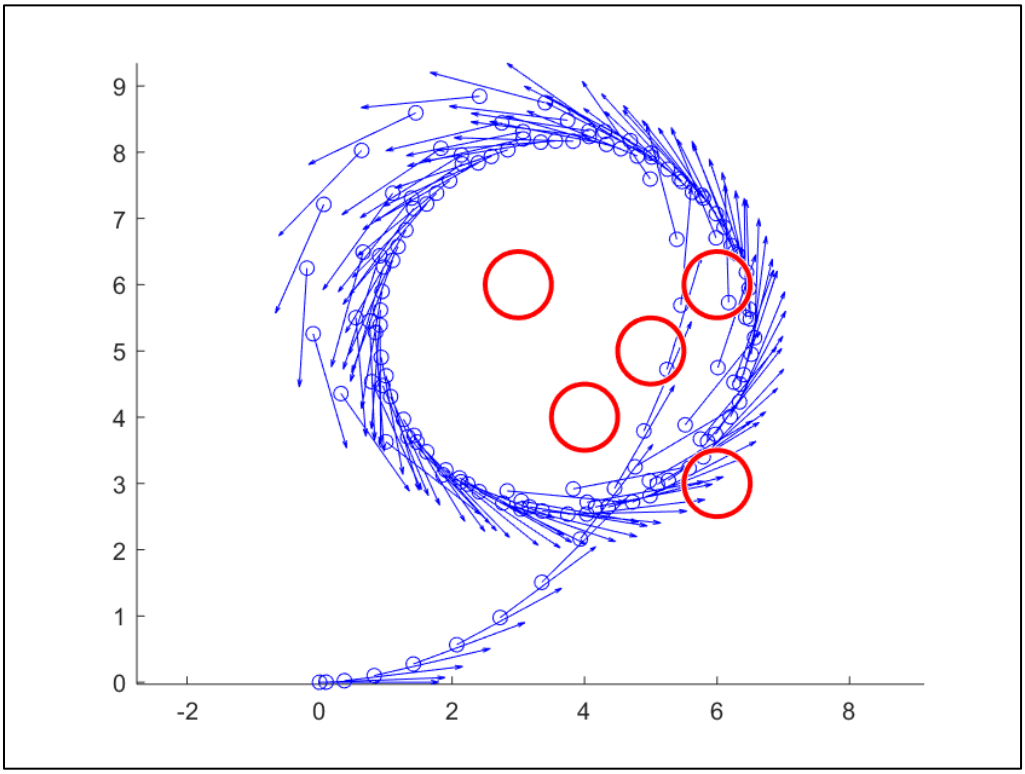


Figure 19: Spiral Behavior

As observed in the figure above, initially with the absence of just one rule, robot was just making spiral movements for scenario 5 which had to be optimized.

7. Optimization

To optimize the results, I would reduce the number of input membership functions to 5 as 7 fuzzy sets are not really required. Reducing the fuzzy set helps in reducing the rule base and, a significant reduction in time step is observed as shown below. Total no. of rules for optimized scenario = 13.

Table 2: Optimized Performance Comparison

Question Number	Goal and Radius	Designed Controller (Time Steps)	Optimized Controller Time Steps (5 MFs)
6a	(10,10) radius 3	217	167
6b	(10,10) radius 0.3	172	172
6c	(20,20) radius 1; (5,20) radius 1	533	506
6d	(0,10) radius 4; (0,0) radius 1; (20,0) radius 5	915	469
6e	(5,5) radius 0.5; (4,4) radius 0.5; (6,6) radius 0.5; (3,6) radius 0.5; (6,3) radius 0.5	779	543

This controller does not work with 3 membership functions, it gives a bizarre output causing a very unstable behavior as shown in the figure below for Scenario 1 and Scenario 3. Scenario 1 works out but takes 1444-time steps and a very poor motion.

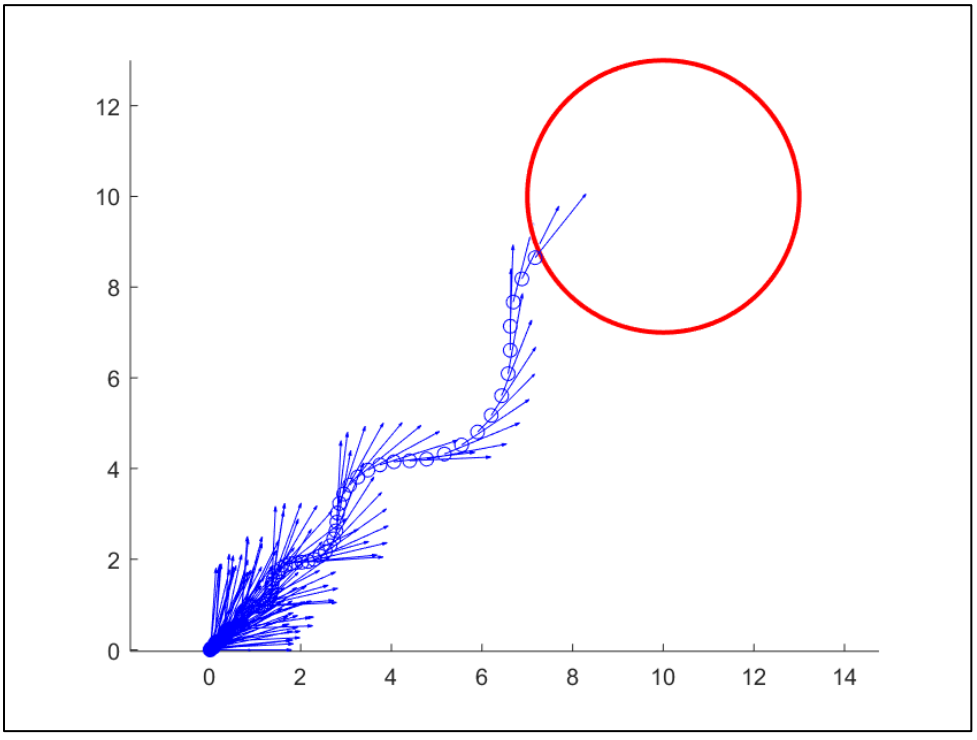


Figure 20: Unstable Output for Scenario 1. Three MFs

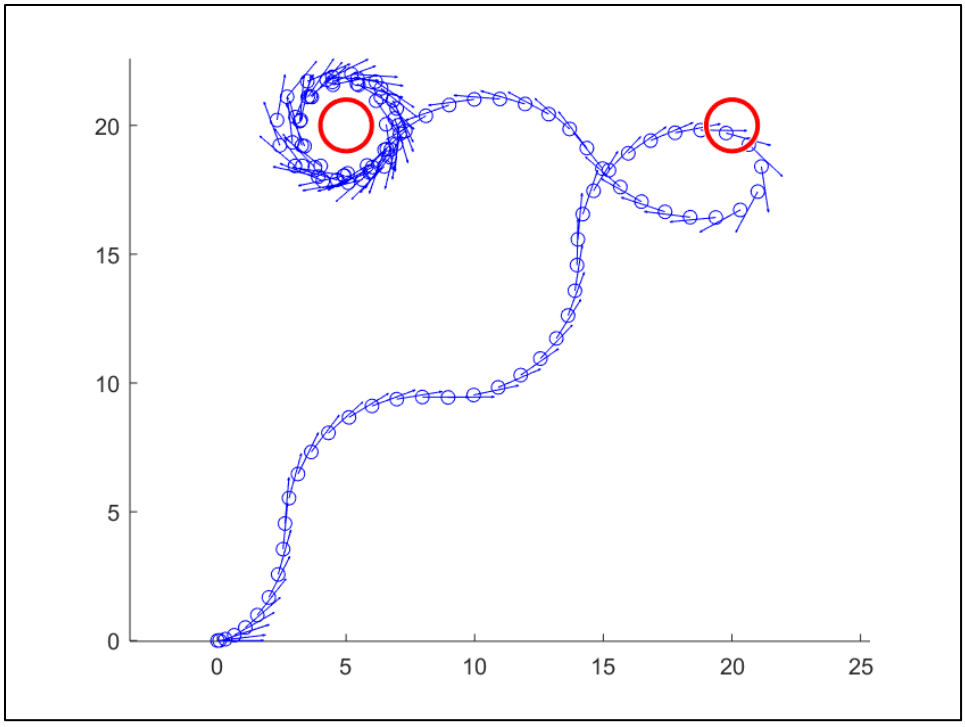


Figure 21: Unstable Output for Scenario 3. Three MFs

Conclusion

Fuzzy logic can be a great tool to create control systems that use only linguistic variables. It can be very useful for operators in factories or people who can't understand scientific data. What I learnt from this project is to go step by step i.e. first to stabilize the angle, run the fuzzy logic and then work on distance and radial accelerations. Trying to make fuzzy rule base by using everything didn't work.

Also, using all the 5 parameters in a single rule did not cause my robot to move and a special care should be taken for conflicting rules when designing such control systems. I initially made 65 rules and got them down to 13 by realizing the use of not rule which can be a very powerful tool to combine some rules to one. The input sequences also made a difference and hence, the provided MATLAB code should also be examined. Overall, a great learning experience.