BUILDING A FUZZY LOGIC EXPERT SYSTEM

Autonomous Golf Cart Navigation

SHAOOR JAN
EXPERT SYSTEM
CS 59000

1 CONTENTS

2 INTRODUCTION

We want to design a fuzzy logic system that would navigate a golf cart around a golf course. There are seven major steps involved in building a fuzzy logic expert system:

- **Step 1: Defining the problem**
- **Step 2: Defining the linguistic variables**
- **Step 3: Define the fuzzy set**
- **Step 4: Define fuzzy rules**
- **Step 5: Build the system**
- **Step 6: Test the system**
- **Step 7: Tune the system**

3 STEPS FOR BUILDING FUZZY LOGIC EXPERT SYSTEM

3.1 DEFINING THE PROBLEM

First, we need to contact an expert to gather the knowledge needed for building the system. To define the problem, we can ask our experts of the approach he would take to solve the problem at hand. Due to the nature of the problem, I was able to provide the expert opinion on my own and did not have to contact someone else.

I defined the problem as to navigate the cart in an efficient and safe fashion from some initial stationary position to the location of the golf ball. Efficiency means the cart should minimize both the distance traveled and travel time. Safety means that any obstacle in the path must be avoided. To achieve both these tasks we must give control of speed and direction of the cart to the expert system. Figure 1 shows the navigation problem.

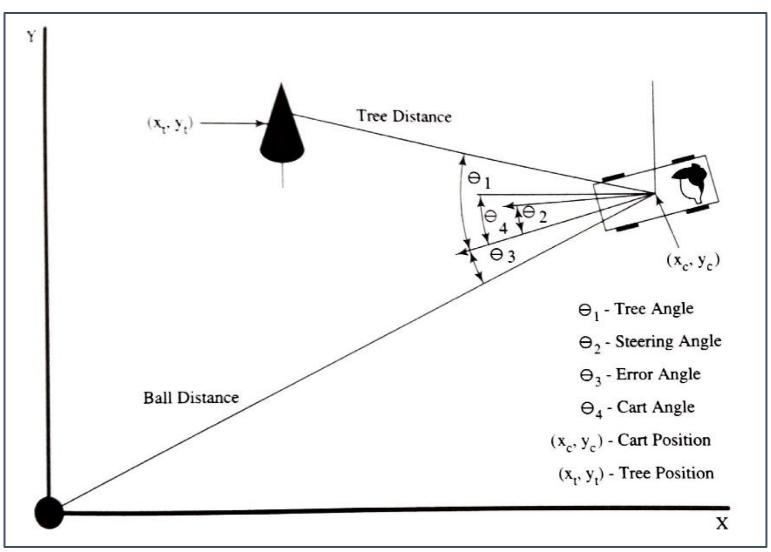


Figure 1 Cart navigation geometry

This becomes an error nulling problem. The cart must initially steer toward the ball by nullifying the error between the angular direction of the cart and the direction toward the ball. The must also reaches some maximum allowed speed, should slow down and eventually stop when it approaches the ball. We choose this distance to be 3 yards. When there is an obstacle in the path, the cart should steer around it cautiously, then pick up speed and again steer toward the ball.

3.2 Define Linguistic Variables

The next step is to define the linguistic variables for our problem. We do this by consulting with an expert. We want to know what variables are needed to represent our universe of discourse and fuzzy sets that we need to define.

From step 1 we know that our system must deal with three basic problems:

1. Control steering of cart to direct it toward the ball
2. Control the cart's speed
3. Control steering to avoid trees

Based on expert opinion I defined the following linguistic variables and their range:

LINGUISTIC VARIABLES	RANGE
ERROR ANGLE	-180 to 180 degree
TREE ANGLE	-180 to 180 degree
STEERING ANGLE	-45 to 45 degree
SPEED	0 to 5 yd/s
ACCELERATION	-2 to 1 yd/s^2
BALL DISTANCE	0 to 600 yards
TREE DISTANCE	0 to 1000 yards

3.3 DEFINE FUZZY SETS

Then we need to define fuzzy sets for each universe. Based on expert opinion the following are the linguistic variables and their corresponding adjectives.

ERROR ANGLE	TREE ANGLE	STEERING ANGLE	SPEED	ACCELERATION	BALL DISTANCE	TREE DISTANCE
Large Negative	Large Negative	Hard Right	Zero	Brake Hard	Brake Hard	Close
Small Negative	Small Negative	Slight Right	Real Slow	Brake Light	Real Close	
Zero	Zero	Zero	Slow	Coast	Close	
Small Positive	Small Positive	Slight Left	Medium	Zero	Medium	

Large Positive	Large Positive	Hard Left	Fast	Slight Acceleration	Far
				Floor It	

Then we need to ask our expert questions that will allow us to define the fuzzy sets for each adjective. We might ask questions like what speed is considered slow or we can ask that to what extent does our expert thinks a given value is slow. In this manner we defined the following fuzzy sets for this problem.

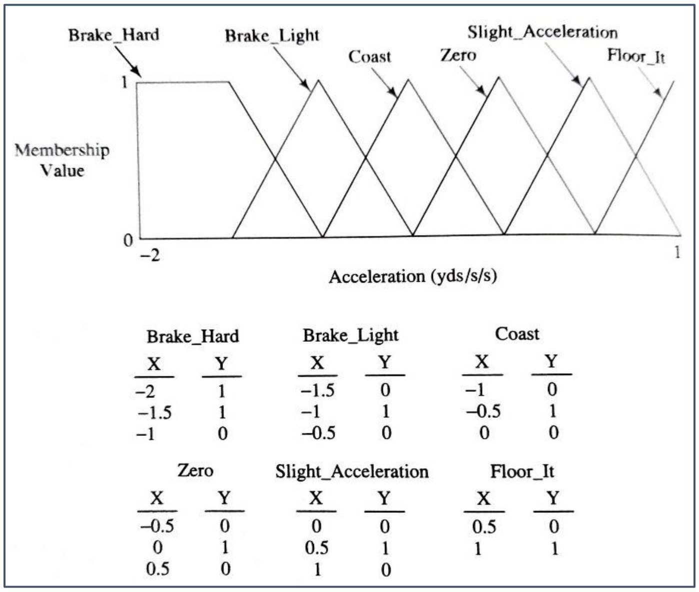


Figure 2 Fuzzy sets on acceleration

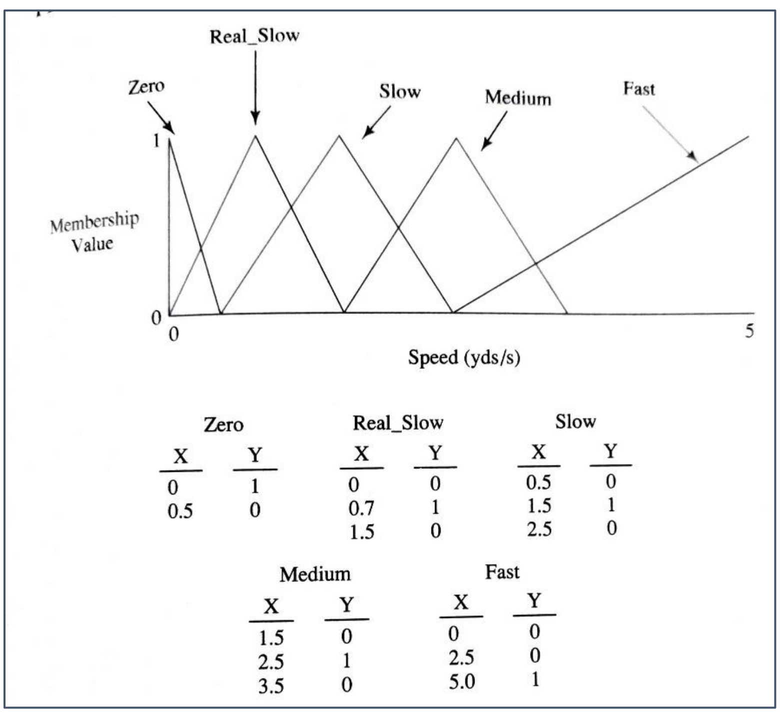
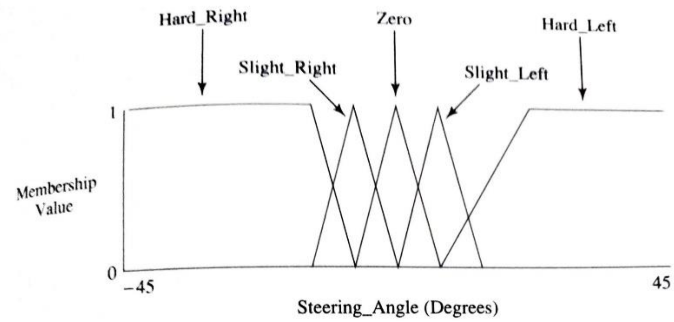


Figure 3 Fuzzy sets on speed



Hard_Left

X	Y
7.5	0
22.5	1
45	1

Slight_Left

X	Y
0	0
7.5	1
15	0

Zero

X	Y
−7.5	0
0	1
7.5	0

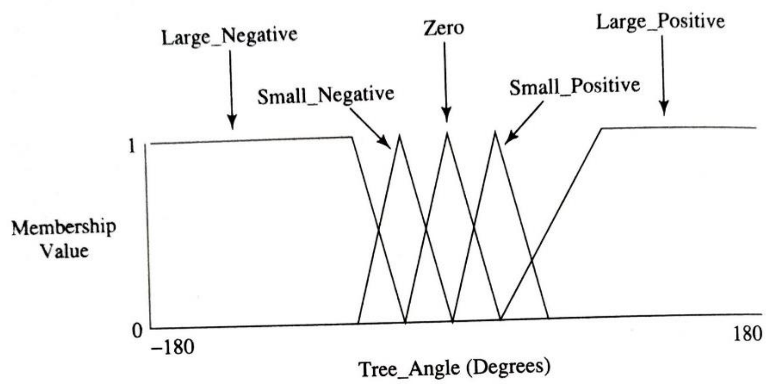
Slight_Right

X	Y
−15	0
−7.5	1
0	0

Hard_Right

X	Y
−45	1
−15	1
−7.5	0

Figure 4 Fuzzy sets on steering angle



Large_Positive

X	Y
30	0
90	1
180	1

Small_Positive

X	Y
0	0
30	1
60	0

Zero

X	Y
−30	0
0	1
30	0

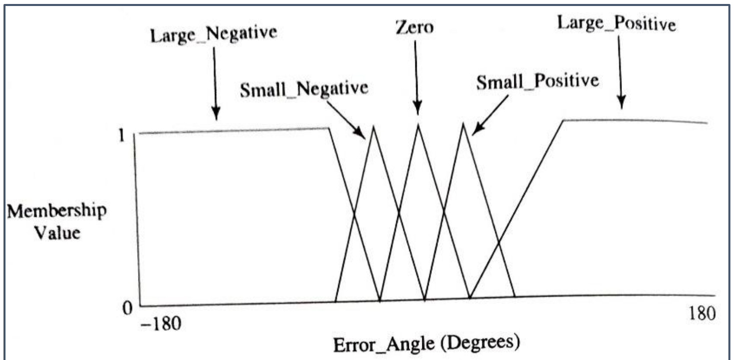
Small_Negative

X	Y
−60	0
−30	1
0	0

Large_Negative

X	Y
−180	1
−60	1
−30	0

Figure 5 Fuzzy sets on tree angle



Large_Positive

X	Y
30	0
90	1
180	1

Small_Positive

X	Y
0	0
30	1
60	0

Zero

X	Y
−30	0
0	1
30	0

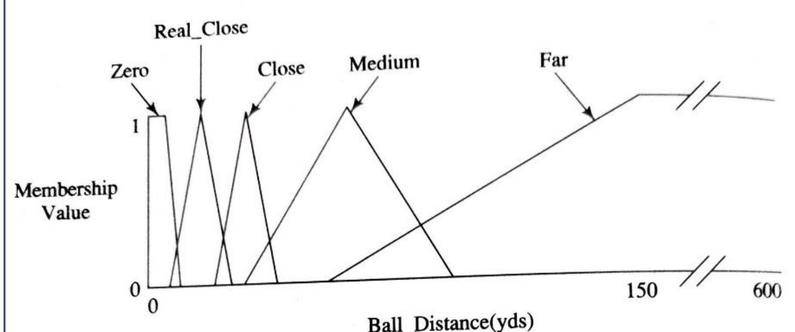
Small_Negative

X	Y
−60	0
−30	1
0	0

Large_Positive

X	Y
−180	1
−60	1
−30	0

Figure 6 Fuzzy sets on error angle



Zero

X	Y
0	1
3	1
9	0

Real_Close

X	Y
3	0
12	1
18	0

Close

X	Y
12	0
24	1
36	0

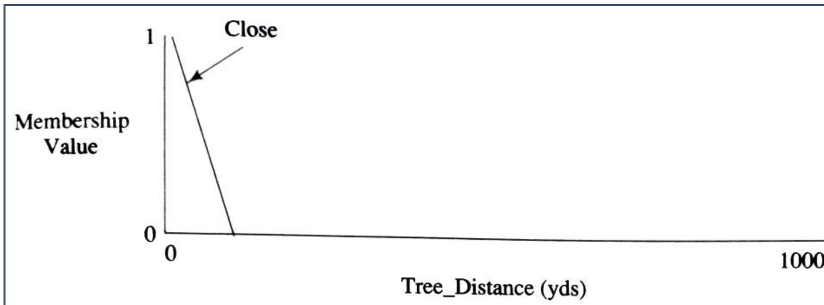
Medium

X	Y
24	0
60	1
96	0

Far

X	Y
60	0
150	1
600	1

Figure 7 Fuzzy set for ball distance



Close

X	Y
0	1
50	0
1000	0

Figure 8 Fuzzy Set for tree distance

3.4 Defining fuzzy rules

Next, we need to define fuzzy rules. For this we need to consult our expert again and ask him about the approach he would take to solve the three basic problems. Based on this discussion we need to come up with rules. Following are the rules used for this problem:

```
// ==== RULES FOR STEERING ====
RULEBLOCK S
    AND : MIN;
    ACT : PROD;
    ACCU : MAX;

    // maintaining steering direction
    RULE 1 : IF error_angle IS zero
             AND tree_dist IS NOT somewhat_close
             AND tree_angle IS NOT somewhat_zero
             THEN steering_angle IS zero;

    // change steering direction slightly right
    RULE 2 : IF error_angle IS small_positive
             AND tree_dist IS NOT somewhat_close
             AND tree_angle IS NOT somewhat_zero
             THEN steering_angle IS slight_right;

    // change steering direction slightly left
    RULE 3 : IF error_angle IS small_negative
             AND tree_dist IS NOT somewhat_close
             AND tree_angle IS NOT somewhat_zero
             THEN steering_angle IS slight_left;

    // change steering direction slightly right
    RULE 4 : IF error_angle is large_positive
             AND speed IS fast
             THEN steering_angle IS slight_right;

    // change steering direction hard right
    RULE 5 : IF error_angle IS large_positive
             AND speed IS NOT fast
             THEN steering_angle IS hard_right;

    // change steering direction slightly left
    RULE 6 : IF error_angle IS large_negative
             AND speed IS fast
             THEN steering_angle IS slight_left;

    // change steering direction hard left
    RULE 7 : IF error_angle IS large_negative
             AND speed IS NOT fast
             THEN steering_angle IS hard_left;

END_RULEBLOCK
```

```
// ==== RULES FOR ACCELERATION ====
RULEBLOCK A
    AND : MIN;
    ACT : PROD;
```

```
ACCU : MAX;

// brake lightly
RULE 1 : IF error_angle IS large_positive
         AND speed IS fast
         THEN acceleration IS brake_light;

// brake lightly
RULE 2 : IF error_angle IS large_negative
         AND speed IS fast
         THEN acceleration IS brake_light;

// floor it
RULE 3 : IF ball_dist IS far
         AND speed IS NOT very_fast
         THEN acceleration IS floor_it;

// set acceleration to zero
RULE 4 : IF ball_dist IS far
         AND speed IS very_fast
         THEN acceleration IS zero;

// slight acceleration
RULE 5 : IF ball_dist IS medium
         AND speed IS NOT fast
         THEN acceleration IS slight_acceleration;

// set acceleration to zero
RULE 6 : IF ball_dist IS medium
         AND speed IS fast
         THEN acceleration IS zero;

// brake lightly
RULE 7 : IF ball_dist IS close
         AND speed IS fast
         THEN acceleration IS brake_light;

// slight acceleration
RULE 8 : IF ball_dist IS close
         AND speed IS zero
         THEN acceleration IS slight_acceleration;

// brake hard
RULE 9 : IF ball_dist IS real_close
         AND speed IS fast
         THEN acceleration IS brake_hard;

// brake lightly
RULE 10 : IF ball_dist IS real_close
          AND speed IS medium
          THEN acceleration IS brake_light;

// coast
RULE 11 : IF ball_dist IS real_close
          AND speed IS slow
          THEN acceleration IS coast;
```

```
    // set acceleration to zero
    RULE 12 : IF ball_dist IS real_close
              AND speed IS real_slow
              THEN acceleration IS zero;

    // slight acceleration
    RULE 13 : IF ball_dist IS real_close
              AND speed IS zero
              THEN acceleration IS slight_acceleration;

    // brake hard
    RULE 14 : IF ball_dist IS zero
              AND speed IS NOT zero
              THEN acceleration IS brake_hard;

    // coast
    RULE 15 : IF ball_dist IS close
              AND speed IS medium
              THEN acceleration IS coast;

    // set acceleration to zero
    RULE 16 : IF ball_dist IS close
              AND speed IS slow
              THEN acceleration IS zero;

END_RULEBLOCK
```

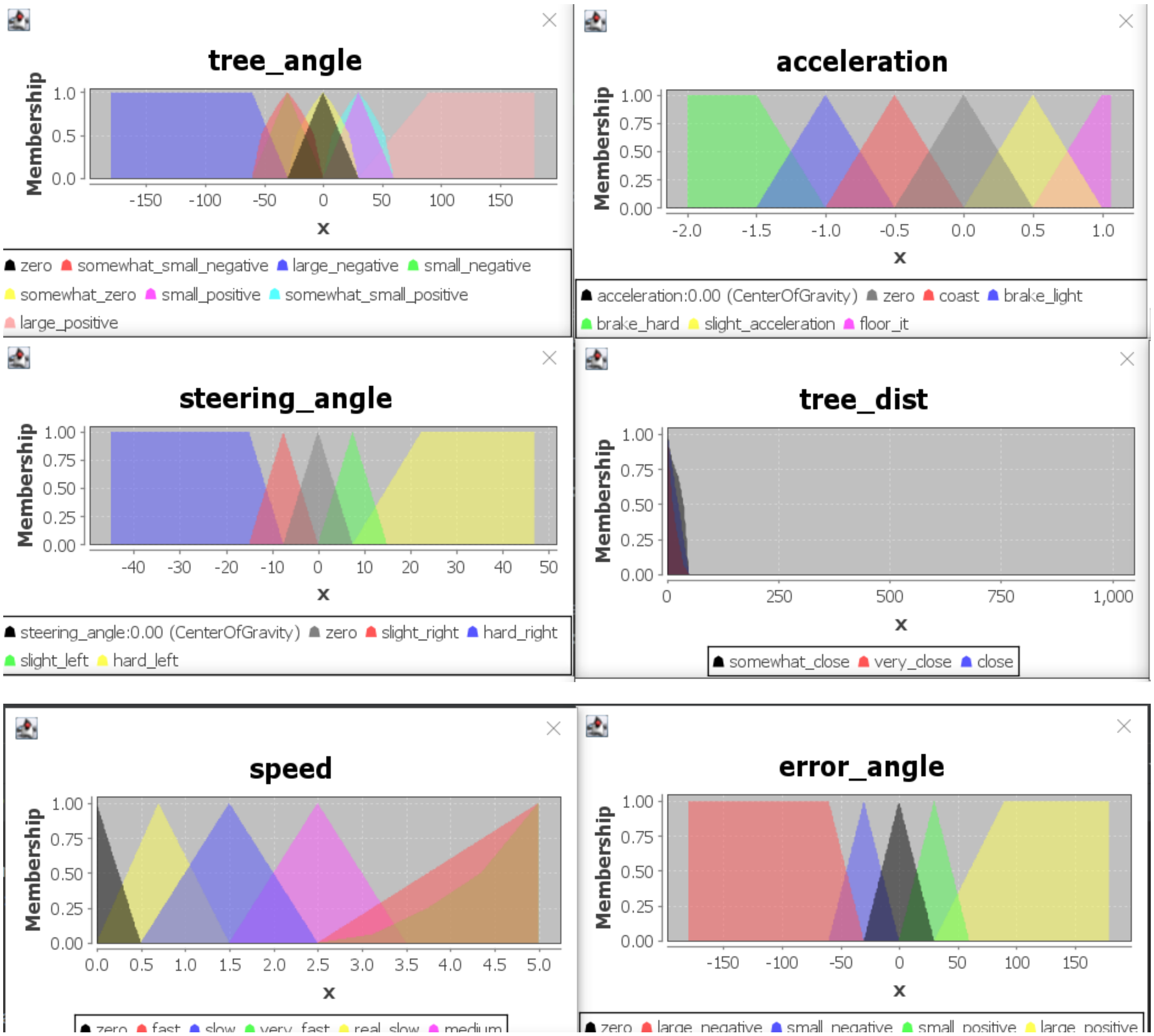
```
// ==== RULES TO AVOID TREE ====
RULEBLOCK T
    AND : MIN;
    ACT : PROD;
    ACCU : MAX;

    // turning slightly left to avoid tree
    RULE 1 : IF tree_dist IS somewhat_close
             AND tree_angle IS somewhat_zero
             AND tree_angle IS somewhat_small_positive
             THEN steering_angle IS slight_left;

    // turn slightly right to avoid tree
    RULE 2 : IF tree_dist IS somewhat_close
             AND tree_angle IS somewhat_zero
             AND tree_angle IS somewhat_small_negative
             THEN steering_angle IS slight_right;

    // brake hard to avoid tree
    RULE 3 : IF tree_dist IS very_close
             AND tree_angle IS zero
             THEN acceleration IS brake_hard;

END_RULEBLOCK
```

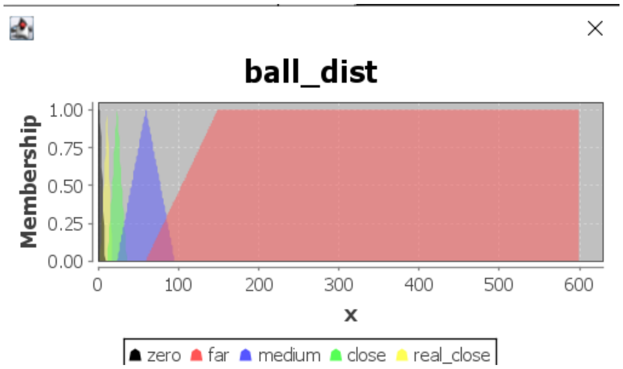
3.5 BUILDING SYSTEM

To build the system I have used java programming language. Besides the default libraries I used jFuzzyLogic and jFreeChart libraries. jFuzzyLogic takes care of the low-level implementation of fuzzy logic. jFreeChart is used for plotting graphs.

I have included project files along with my submission.

Following are the fuzzy sets as used in the project:





3.6 Test System

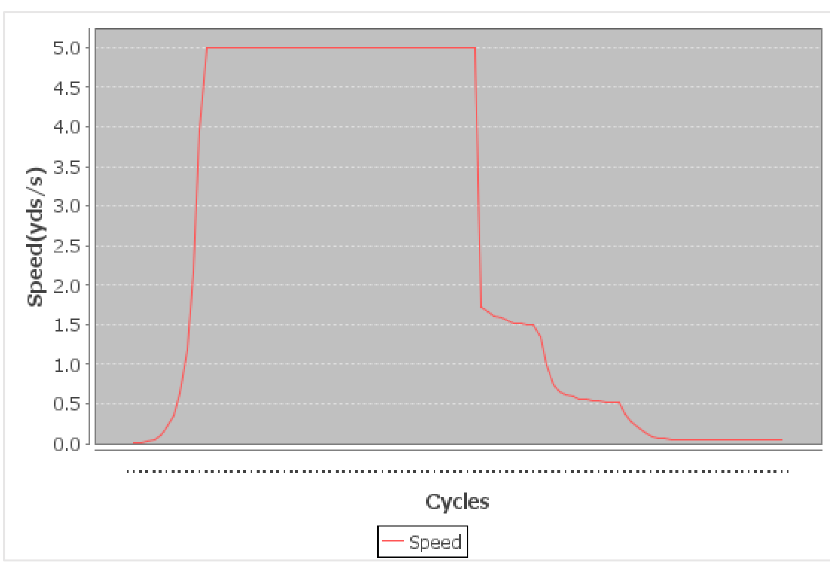
After building the system we need to test the system to see if it meets the specifications defined during step 1. I have performed two tests. In the first test I considered an initial position in which the tree is not in path of the cart. In second case tree is in the path of the cart.

3.6.1 Test case 1: No tree in path

Initial conditions used:

ERROR ANGLE: -45 DEGREE
BALL DISTANCE: 250 YARDS
TREE DISTANCE: 1000 YARDS
TREE ANGLE: 10 DEGREE
SPEED: 0.005 YDS/S
ACCELERATION: 0 YDS/S^2
STEERING ANGLE: 0 DEGREE

Following is speed of the cart for the first 100 cycles:



We can see that the cart starts from a stationary condition and quickly accelerated to the maximum possible speed. It stays at that speed and start deaccelerating when it gets close to the ball. It comes to a stop when gets close to the ball.

The response is as we expected. Now let us stop the simulation at a point and observe the acceleration.

> ERROR ANGLE: 0 DEGREE
> BALL DISTANCE: 28.5 YARDS
> TREE DISTANCE: 1000 YARDS
> TREE ANGLE: 10 DEGREE
> SPEED: 4.5 YDS/S
> ACCELERATION: 0.5 YDS/S^2
> STEERING ANGLE: 0 DEGREE

At these conditions, the following three rules will be fired.

```
// slight acceleration
RULE 5 : IF ball_dist IS medium
         AND speed IS NOT fast
         THEN acceleration IS slight_acceleration;

// set acceleration to zero
RULE 6 : IF ball_dist IS medium
         AND speed IS fast
         THEN acceleration IS zero;

// brake lightly
RULE 7 : IF ball_dist IS close
         AND speed IS fast
         THEN acceleration IS brake_light;
```

The following figure shows the expected response:

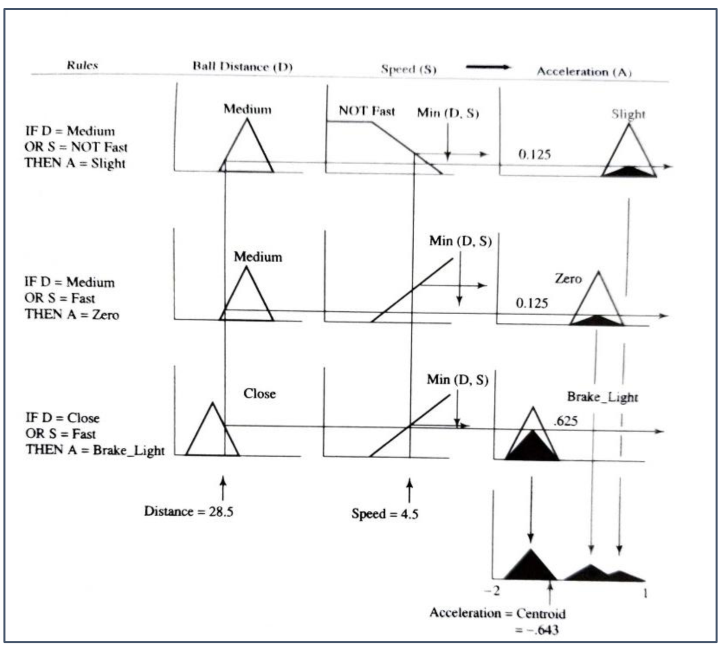


Figure 9 Fuzzy inference sample

Result I got:

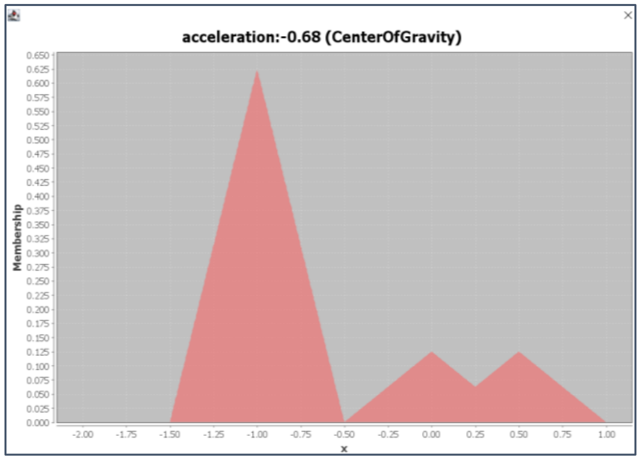


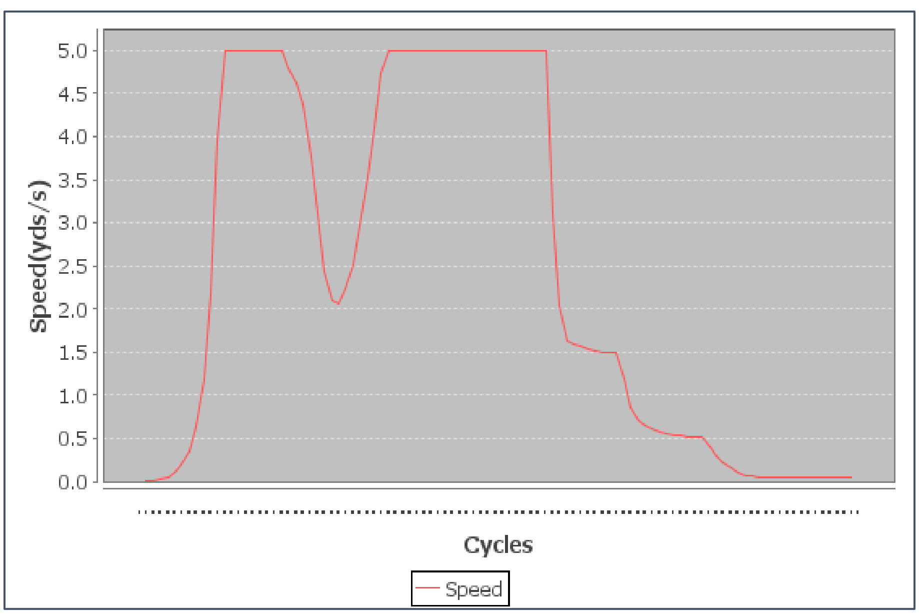
Figure 10 Fuzzy inference smaple

3.6.2 Test case 2: Tree in path

Next, I considered the case where a tree is in the path of the cart. Initial conditions are:

- ERROR ANGLE: 0 DEGREE
- BALL DISTANCE: 250 YARDS
- TREE DISTANCE: 100 YARDS
- TREE ANGLE: 10 DEGREE
- SPEED: 0.005 YDS/S
- ACCELERATION: 0 YDS/S^2
- STEERING ANGLE: 0 DEGREE

Following is speed of the cart for the first 100 cycles:



3.7 TUNE SYSTEM

After testing our system, we need to compare its response with the initial expectations. In most cases the time spent on developing fuzzy sets and rules is small as compared to the time spent testing the system. The initial system gives us a reasonably accurate answer that we then must tweak to get the desired response. This is one of the advantages of fuzzy logic. By using common sense, we can build a reasonable enough system that provides a quick and reasonable result.

4 REFERENCES

- Expert Systems: design and development by Durkin
- http://jfuzzylogic.sourceforge.net/html/pdf/Cingolani_Alcala-Fdez_jFuzzyLogic_2013_IJCIS.pdf
- https://en.wikipedia.org/wiki/International_Electrotechnical_Commission
- http://jfuzzylogic.sourceforge.net/html/manual.html#details
- https://en.wikipedia.org/wiki/Fuzzy_Control_Language