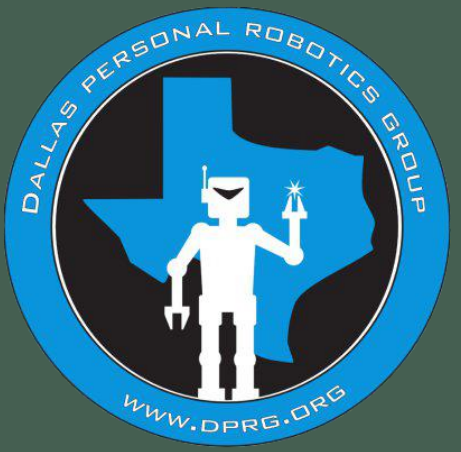


Intro to Fuzzy Logic

Presented by
Doug Paradis

Copyright © Doug Paradis • www.dprg.org

Overview:

- What is Fuzzy Logic?
- Why would we use Fuzzy Logic?
- The “Tip” problem – the “Hello world” for Fuzzy Logic.
- Fuzzy Logic with Arduino.
- Demo: A Fuzzy Logic line following controller.

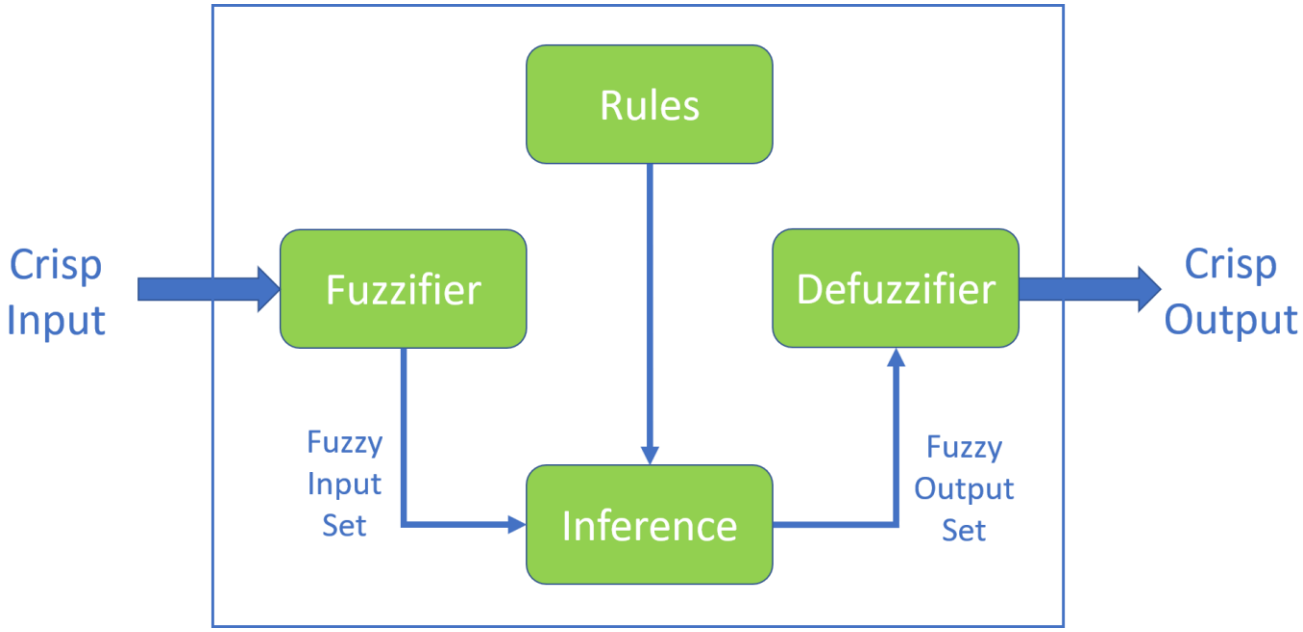


Image source:Based on “Artificial Intelligence – Fuzzy Logic Systems”
https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_fuzzy_logic_systems.htm
Copyright © tutorialspoint.com

What is Fuzzy Logic?

- First of all Fuzzy Logic is not a Fuzzy (i.e., a non precise technique). It has strong mathematical underpinnings. Formalized by Lotfi Zadeh in 1965, it has been applied to a multitude of tasks.

- It is a method that resembles human reasoning. It allows for intermediate possibilities between digital values similar to how our brain takes in information.

- For example, take this recipe:
  - Cut two slices of bread **medium thick**.
  - Turn the heat on the griddle on **high**.
  - Grill the slices on one side until **golden brown**.
  - Turn the slices over and add a **generous helping** of cheese.
  - Replace and grill until the top of the cheese is **slightly brown**.
  - Remove, sprinkle on a **small amount** of black pepper, and eat.

Example taken from Programming Game AI by Example – Mat Buckland

dep - 07/14/2018 Copyright © Doug Paradis • www.dprg.org

Why would we use Fuzzy Logic?

- Simplicity and flexibility
- Can handle problems with imprecise and incomplete data
- Can model nonlinear functions of arbitrary complexity
- Cheaper to develop
- Cover a wider range of operating conditions
- More readily customizable in natural language terms
- No need to re-train system when new data or rules are added to the system (besides rule conflict check).

Source: https://www.quora.com/What-are-the-advantages-of-fuzzy-logic

dep - 07/14/2018 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem

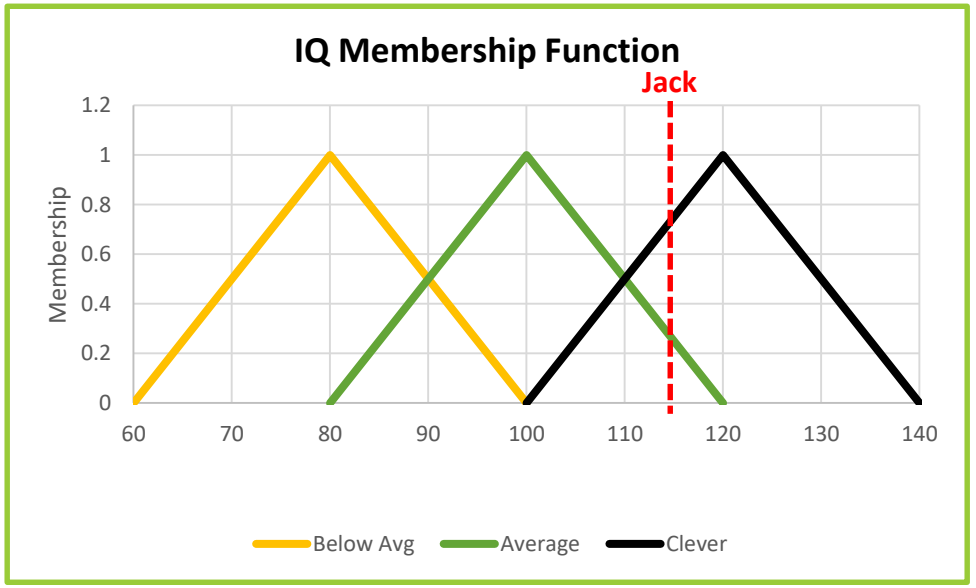
- **Problem:** How much to tip in a restaurant?
- **Step 1:** Define **F**uzzy **L**inguistic **V**ariables (FLV’s)
  - **Inputs:**
    - Service – Poor, Good, Excellent
    - Food Quality – Rancid, Delicious
  - **Output:** Tip – Cheap, Average, Generous
- The FLV’s are:
  - Service and Food Quality
  - Tip
- The Fuzzy sets are:
  - Poor, Good, Excellent
  - Rancid, Delicious
  - Cheap, Average, Generous

dep - 07/14/2018 Copyright © Doug Paradis • www.dprg.org

Example Membership Function

Step 2: Build Membership Functions for the fuzzy sets.

Jack has an IQ of 115. How would you describe his IQ?

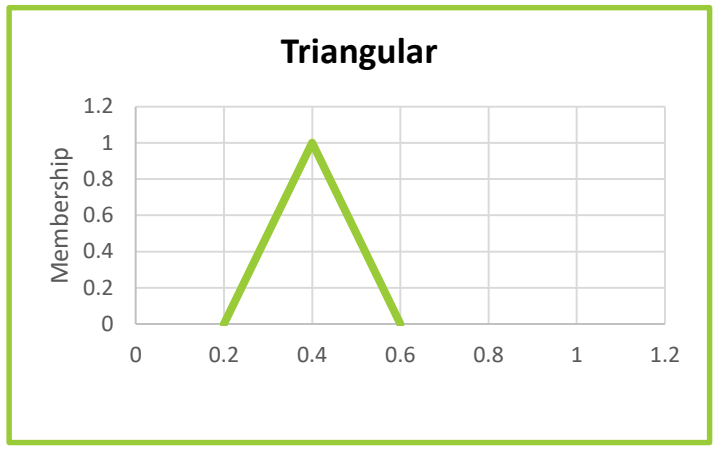


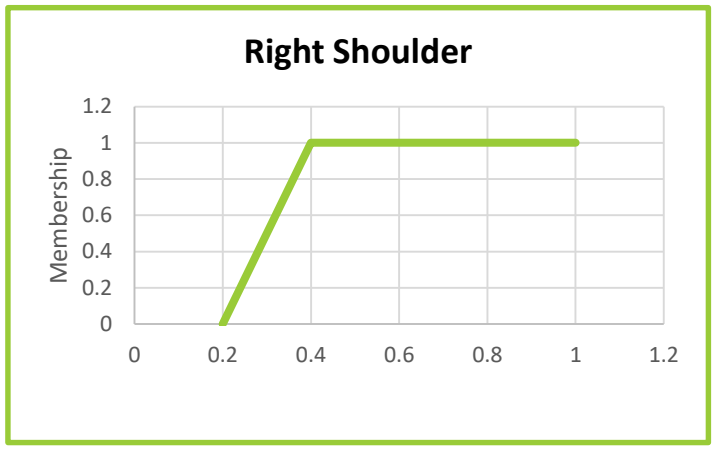
Example Membership Function for IQ

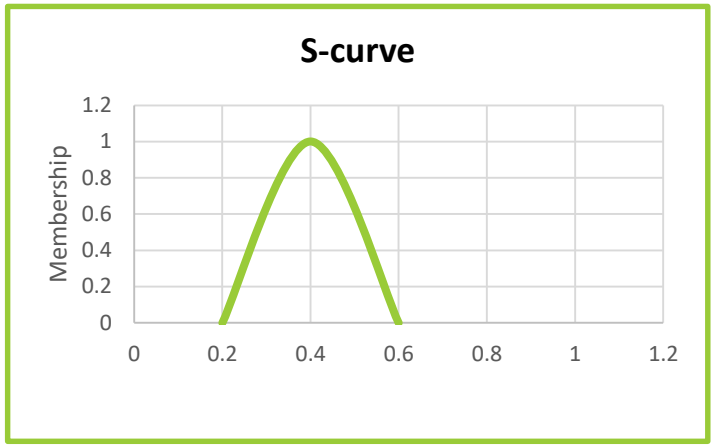
Example taken from Programming Game AI by Example – Mat Buckland

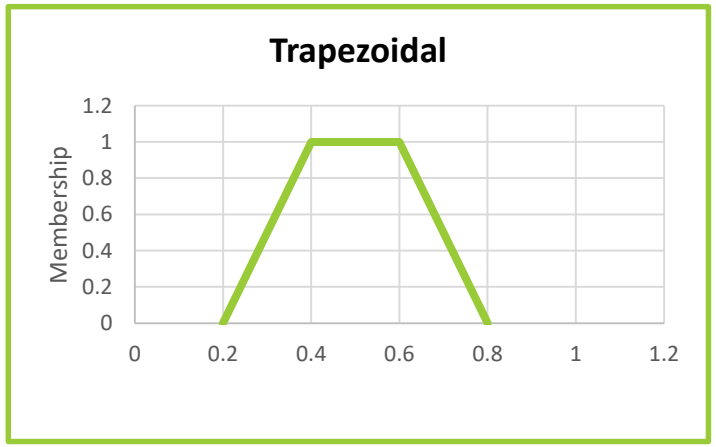
 Copyright © Doug Paradis • www.dprg.org

Typical Membership Function Shapes

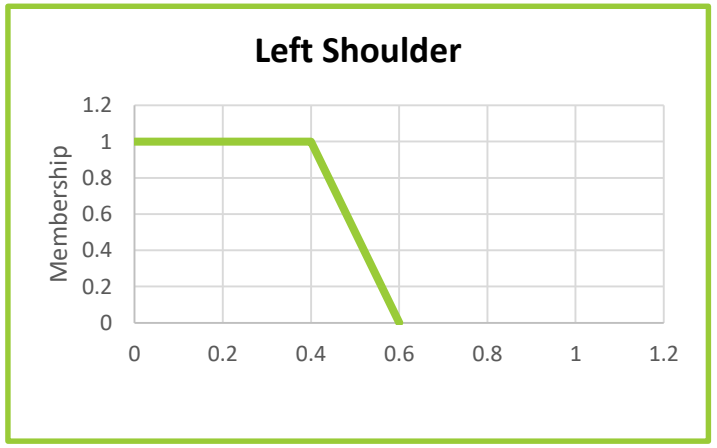


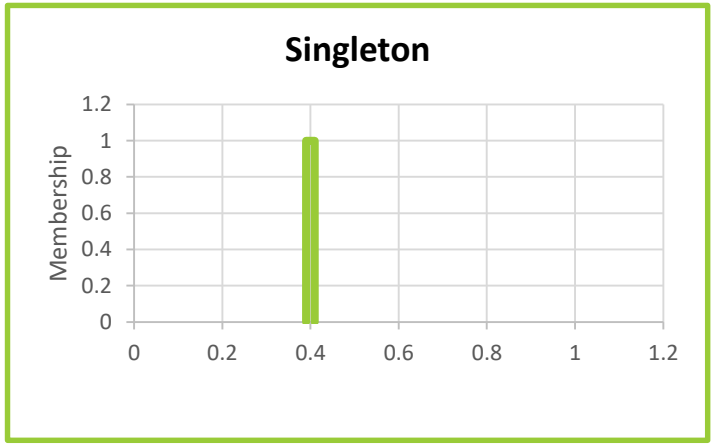






No-No's





 Copyright © Doug Paradis • www.dprg.org

Fuzzy Set Operators

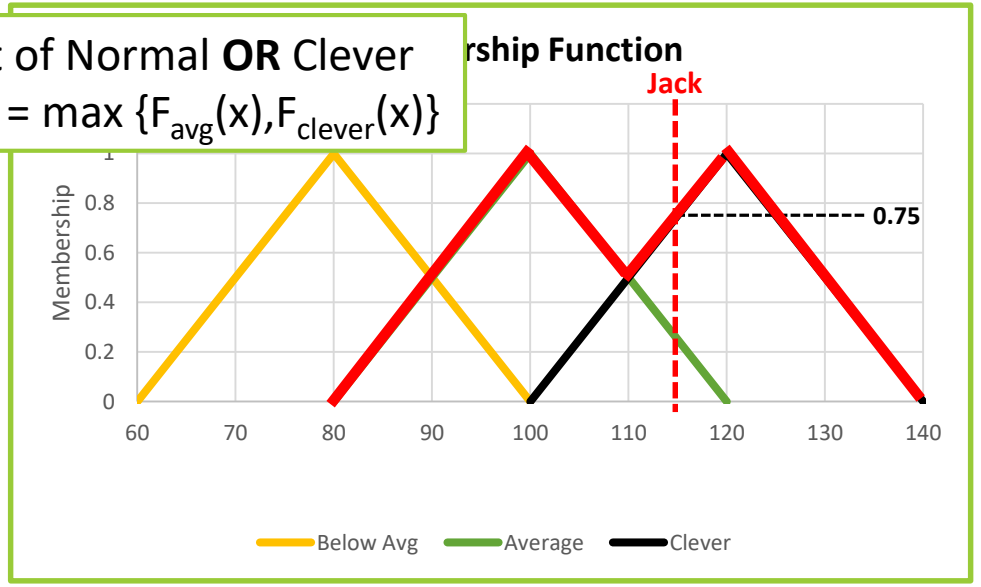
Jack has an IQ of 115.

Jack is 0.75 Clever and 0.25 Normal

Note: there are other operators (such as NOT), but we will show AND and OR.

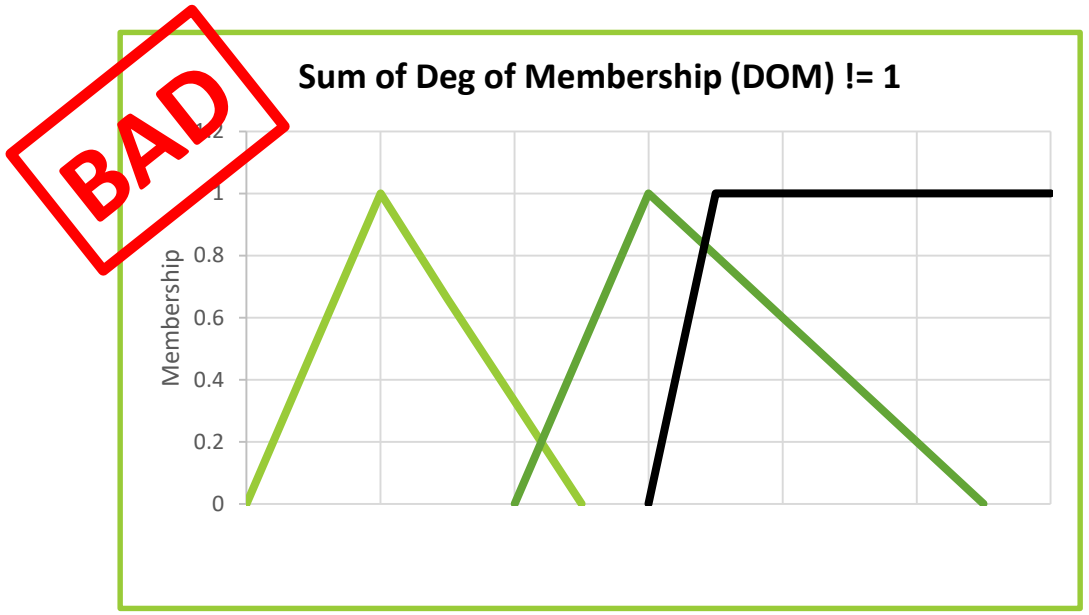
Result of Normal **AND** Clever is $F(x) = \min \{F_{avg}(x), F_{clever}(x)\}$

Result of Normal **OR** Clever is $F(x) = \max \{F_{avg}(x), F_{clever}(x)\}$

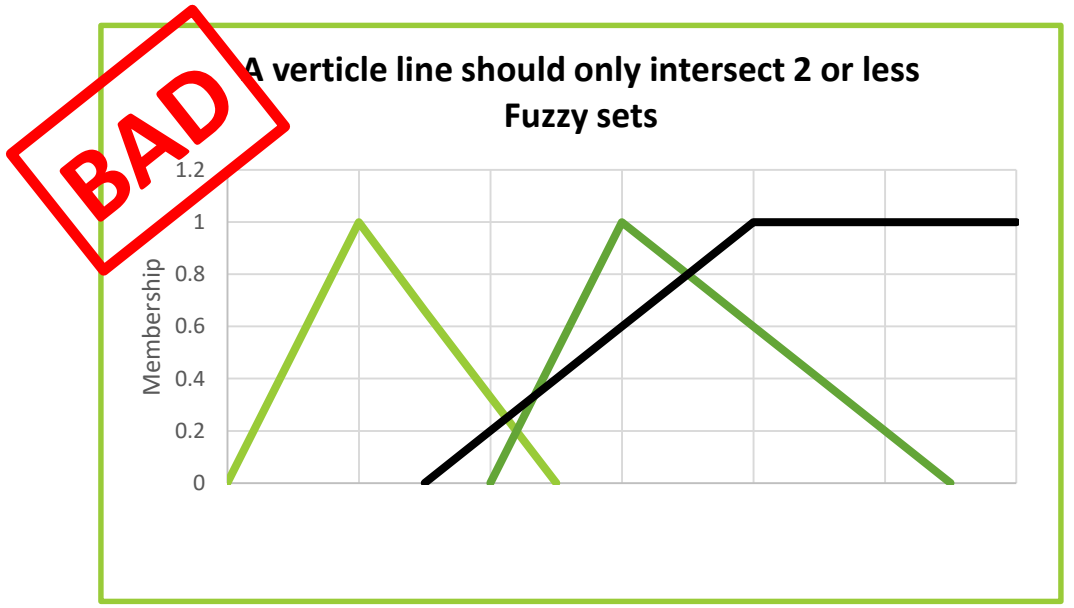


 Copyright © Doug Paradis • www.dprg.org

Membership Function No-No's



Any vertical line drawn through the FLV should sum to ~ 1.

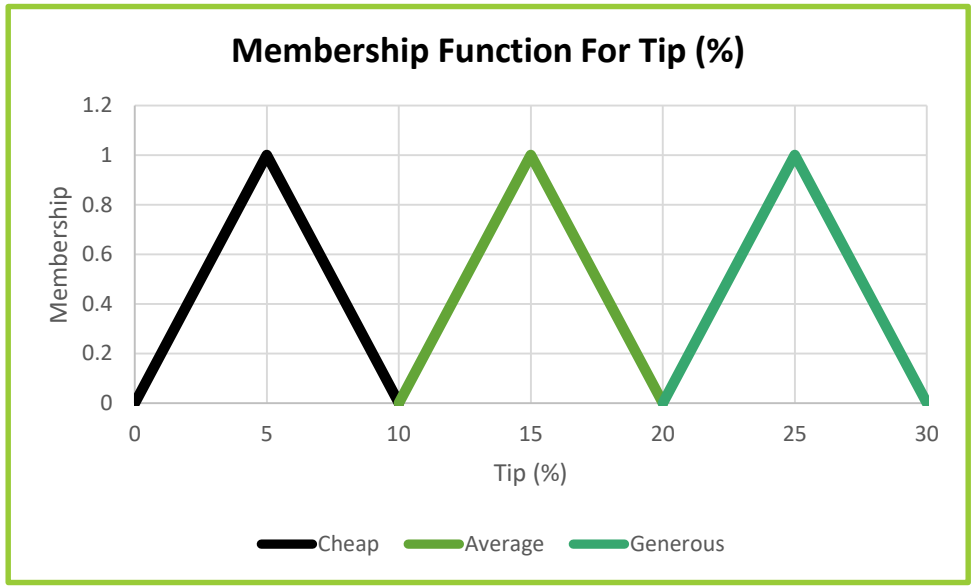


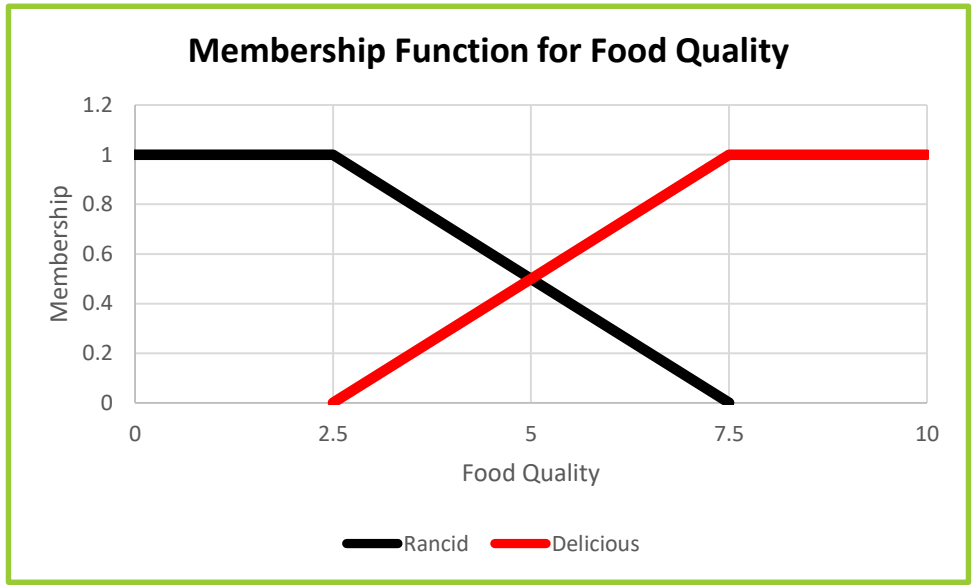
Any vertical line drawn through the FLV should only intersect 2 or less Fuzzy sets.

 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

- **Problem:** How much to tip in a restaurant?
- **Step 1:** Define linguistic variables
  - **Inputs:**
    - Service – Poor, Good, Excellent
    - Food Quality – Rancid, Delicious
  - **Output:** Tip – Cheap, Average, Generous
- **Step 2:** Define Membership Functions

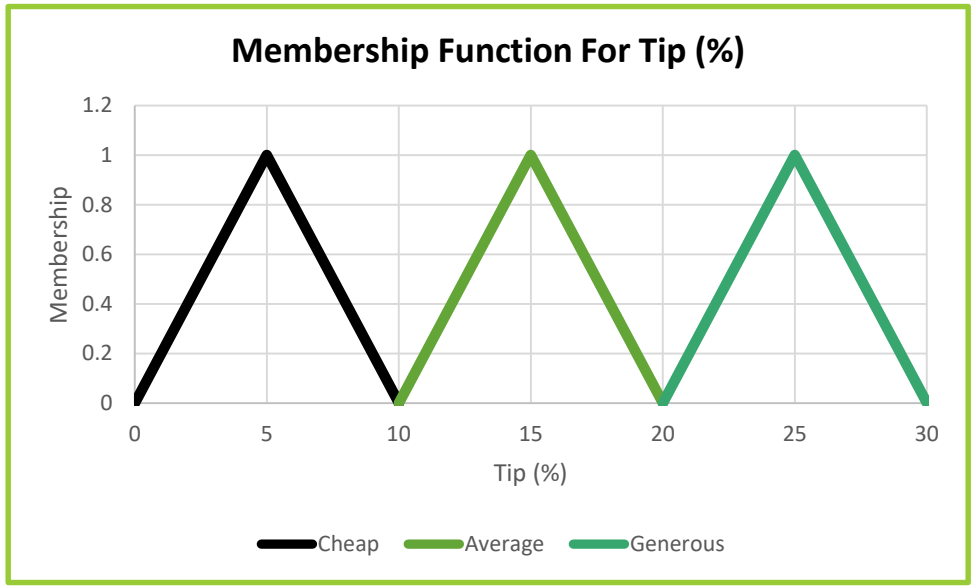


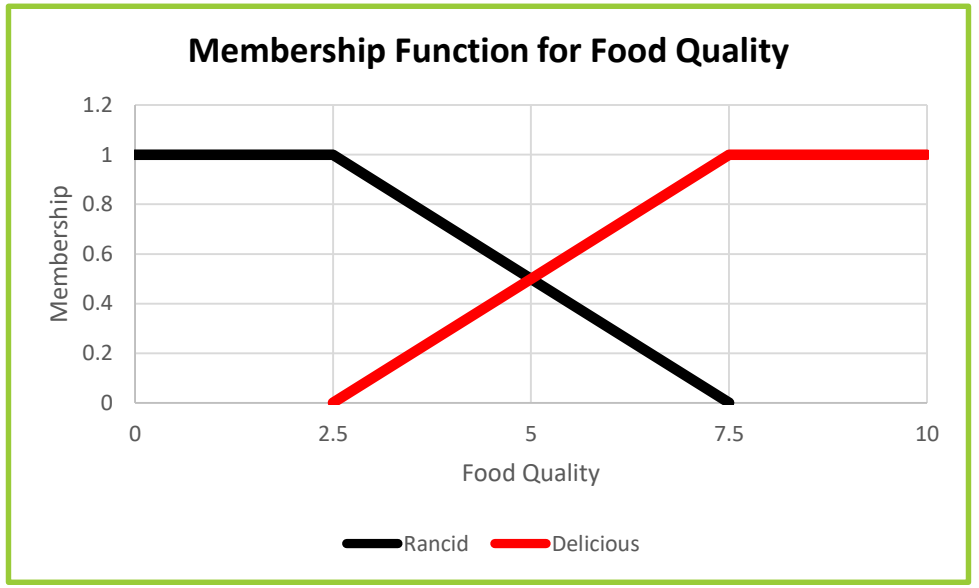


 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

- **Step 3:** Construct a Rule set.
  - Rule 1: If service is poor or food is rancid then tip is cheap.
  - Rule 2: If service is good then tip is average.
  - Rule 3: If service is excellent or food is delicious, then tip is generous.
  - Antecedents are in orange and Consequents are in purple.





 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

- **Step 4: Fuzzy Inference**
  - For each rule:
    - For each antecedent, calculate the degree of membership of the input data.
    - Calculate the rule’s inferred conclusion using the DOM of the input data.
  - Combine all the inferred conclusions into a single conclusion (a fuzzy set).
  - Finally, defuzzify the conclusion into a crisp number.
  - **We will use the following set of inputs to determine the tip.**
    - **Food Quality was rated 4.**
    - **Service was rated 7.**

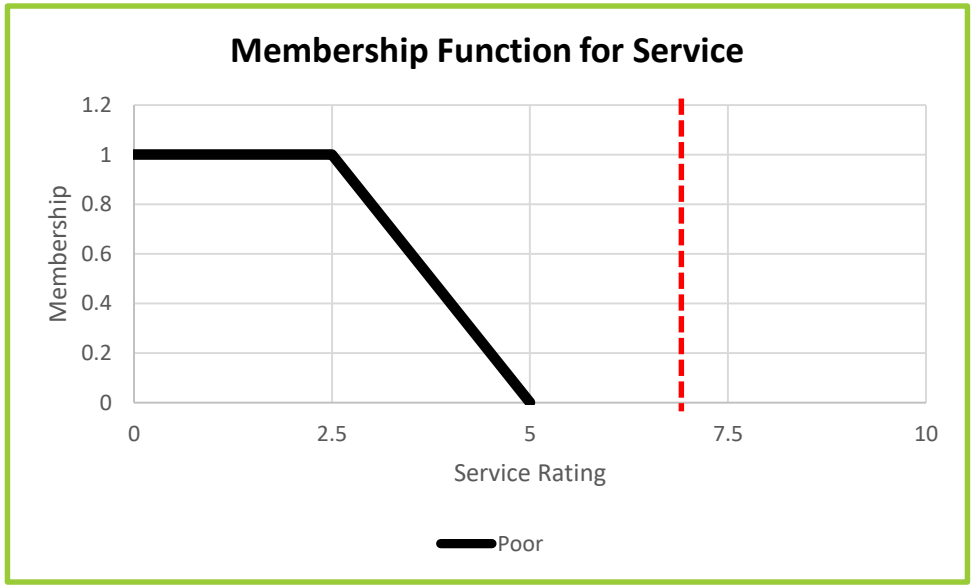
 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

- **Food Quality was rated 4.**
- **Service was rated 7.**

Rule 1: If service is poor or food is rancid then tip is cheap.

Inferred conclusion for Rule 1 is tip is cheap (0.65).



 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

- **Food Quality was rated 4.**
- **Service was rated 7.**

Rule 2: If service is good then tip is average.

Inferred conclusion for Rule 2 is tip is average (0.2).

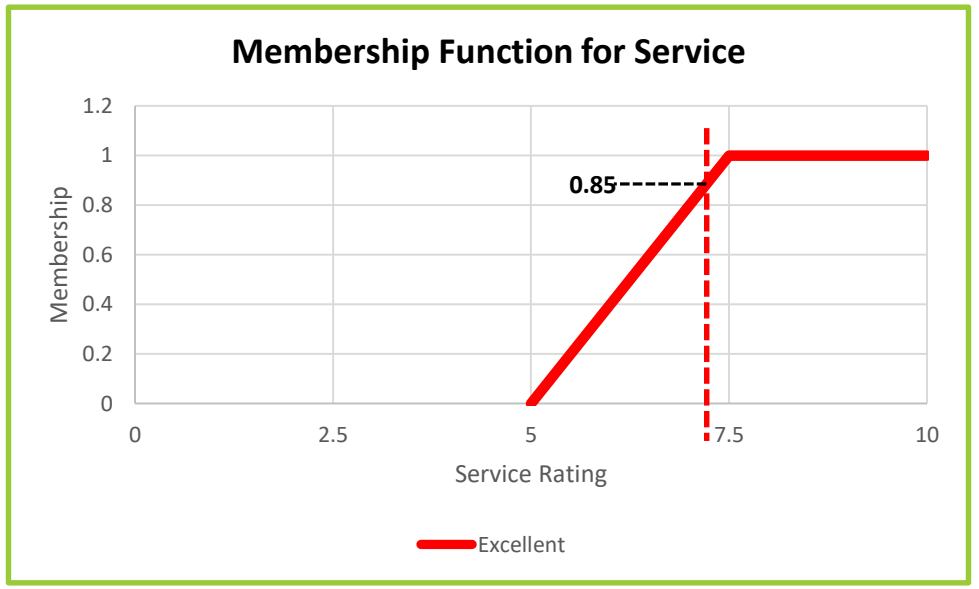
 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

- **Food Quality was rated 4.**
- **Service was rated 7.**

Rule 3: If service is excellent or food is delicious, then tip is generous.

Inferred conclusion for Rule 2 is tip is generous (0.85).



 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

- **Food Quality was rated 4.**
- **Service was rated 7.**

Fuzzy Associative Matrix

Service/Food Quality	Rancid	Delicious
Poor	Cheap (0.65)	Cheap (0)
Good	Cheap(0.65)	Average(0.2)
Excellent	Cheap(0.65) / Generous(0.85)	Generous (0.85)

Where we have multiple confidences we will take the maximum.

dep - 07/14/2018 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

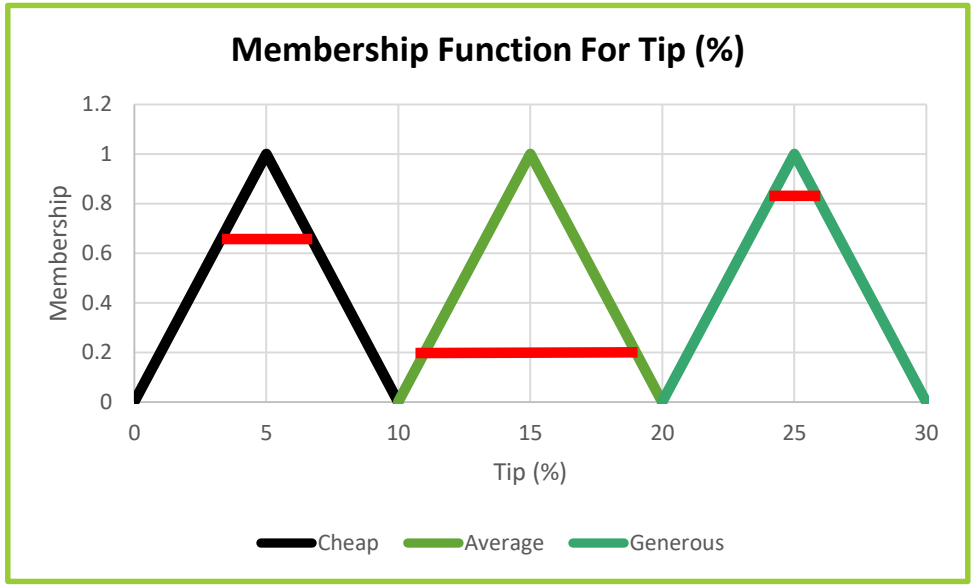
- **Food Quality was rated 4.**
- **Service was rated 7.**

Summarizing Confidences

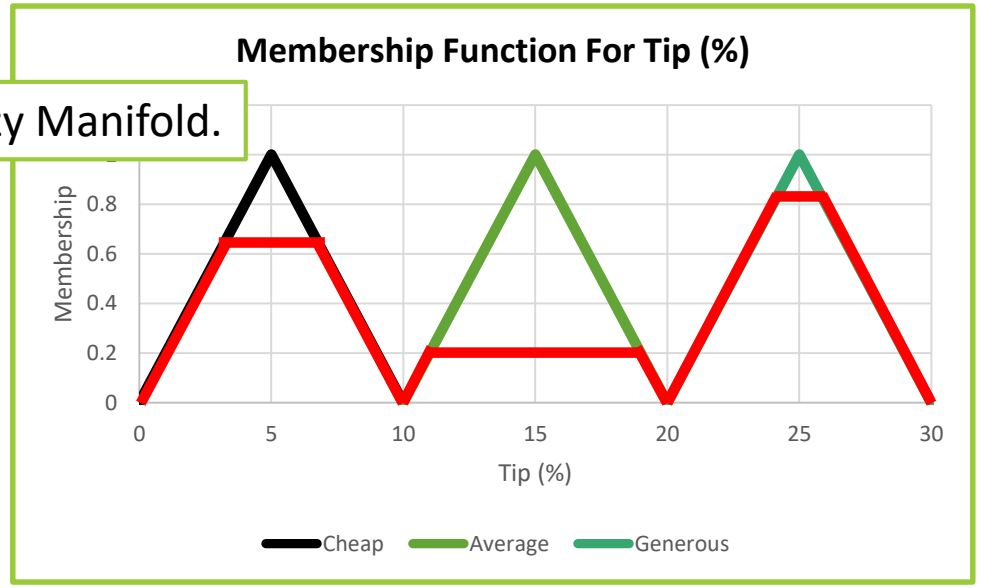
Consequent	Confidence
Cheap	0.65
Average	0.2
Generous	0.85

Clip each consequent to the level of confidence and combine them to form a Fuzzy Manifold.

We are now ready for **Defuzzification**. This is where we take the Fuzzy set that makes up the Fuzzy Manifold and turn it back into a crisp output.



Fuzzy Manifold.



 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

- **Food Quality was rated 4.**
- **Service was rated 7.**

Defuzzification is the opposite of fuzzification. There are several methods that can be used.

- **Mean of Maximum (MOM)** – only takes into account the highest confidence.
- **Centroid** – Takes more calculations but is the most accurate
- **Average of Maxima (MaxAv)** – A good compromise between amount of computing and accuracy.
- And many others…

We will use the **Average of Maxima** since we are doing this by hand.

Average of Maxima

$$\text{Crisp Value} = \frac{\Sigma \text{ representative value x confidence}}{\Sigma \text{ confidence}}$$

Set	Representative Value	Confidence
Cheap	5	0.65
Average	15	0.2
Generous	25	0.85

For **Triangular** sets the representative value is simply the center point. For **Trapezoidal** sets the representative value is average of the values at the beginning and end of the Plateau.

dep - 07/14/2018 Copyright © Doug Paradis • www.dprg.org

The “Tip” problem - cont

- **Food Quality was rated 4.**
- **Service was rated 7.**

$$\text{Tip} = \frac{5 \times 0.65 + 15 \times 0.2 + 25 \times 0.85}{0.65 + 0.2 + 0.85}$$

$$\text{Tip} = \frac{27.5}{1.7}$$

Tip = 16.18 %

Average of Maxima

$$\text{Crisp Value} = \frac{\Sigma \text{ representative value} \times \text{confidence}}{\Sigma \text{ confidence}}$$

Set	Representative Value	Confidence
Cheap	5	0.65
Average	15	0.2
Generous	25	0.85

For **Triangular** sets the representative value is simply the center point. For **Trapezoidal** sets the representative value is average of the values at the beginning and end of the Plateau.

dep - 07/14/2018 Copyright © Doug Paradis • www.dprg.org

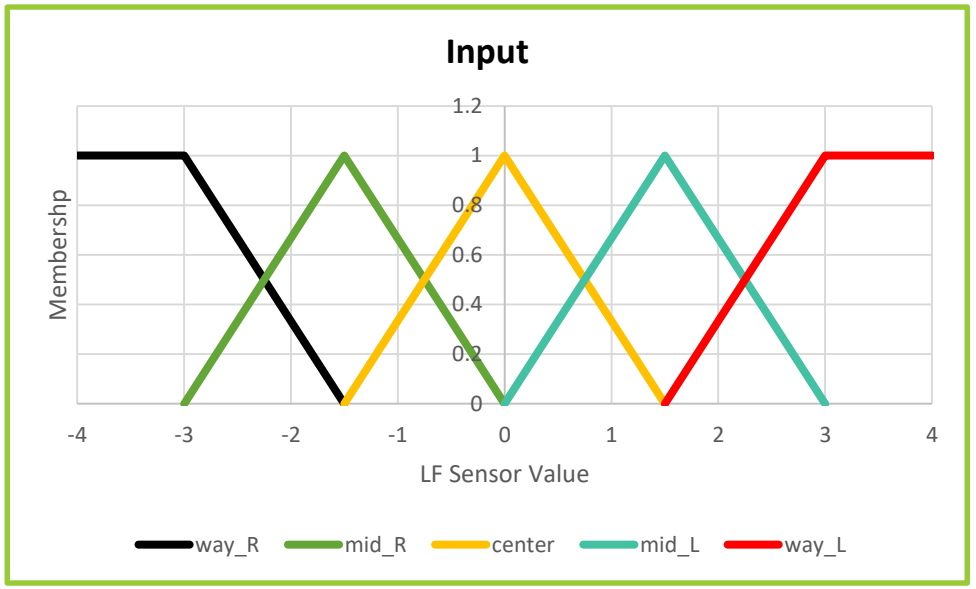
Arduino – “There is a library for that…”

- There is a Arduino library for Fuzzy logic named **eFLL**. It can be found at https://github.com/zerokol/eFLL.
- It was programmed by a group of students in Brazil, so some of the comments are in Portuguese.
- Written in C++/C, the library only uses the standard C language library "stdlib.h", so eFLL is a library designed not only to Arduino, but any Embedded System.
- The library contains two Arduino examples and two general programming examples. Between the two examples and Google Translate the library is easy to use.
- For inference and composition it use (MAX-MIN) and (Mamdani Minimum) and (CENTER OF AREA) for defuzzification in a continuous universe.
- Last updated on 2017-08-21.

 Copyright © Doug Paradis • www.dprg.org

Demo: A Fuzzy Logic line following controller.

```
/*****************************************************************
*  fuzzy_logic_funcs.cpp
*  version: 20180710           Doug Paradis
*
*****************************************************************/

#include "fuzzy_logic_funcs.h"

// Instantiating an object from the eFF fuzzy logic library
Fuzzy* fuzzy = new Fuzzy();


void init_Fuzzy_Logic ()
{

    // Setup Membership function for input (LF sensor value)
FuzzySet* far_to_L = new FuzzySet(-4, -4, -3, -2);
FuzzySet* leftish = new FuzzySet(-3, -1.5, -1.5, 0);
FuzzySet* center = new FuzzySet(-1.75, 0, 0, 1.75);
FuzzySet* rightish = new FuzzySet(0, 1.5, 1.5, 3);
FuzzySet* far_to_R = new FuzzySet(2, 3, 4, 4);

// Define Linguistic variables and terms for input
// FuzzyInput
FuzzyInput* LF_sen_val = new FuzzyInput(1);
LF_sen_val->addFuzzySet(far_to_L);
LF_sen_val->addFuzzySet(leftish);
LF_sen_val->addFuzzySet(center);
LF_sen_val->addFuzzySet(rightish);
LF_sen_val->addFuzzySet(far_to_R);

fuzzy->addFuzzyInput(LF_sen_val);
```



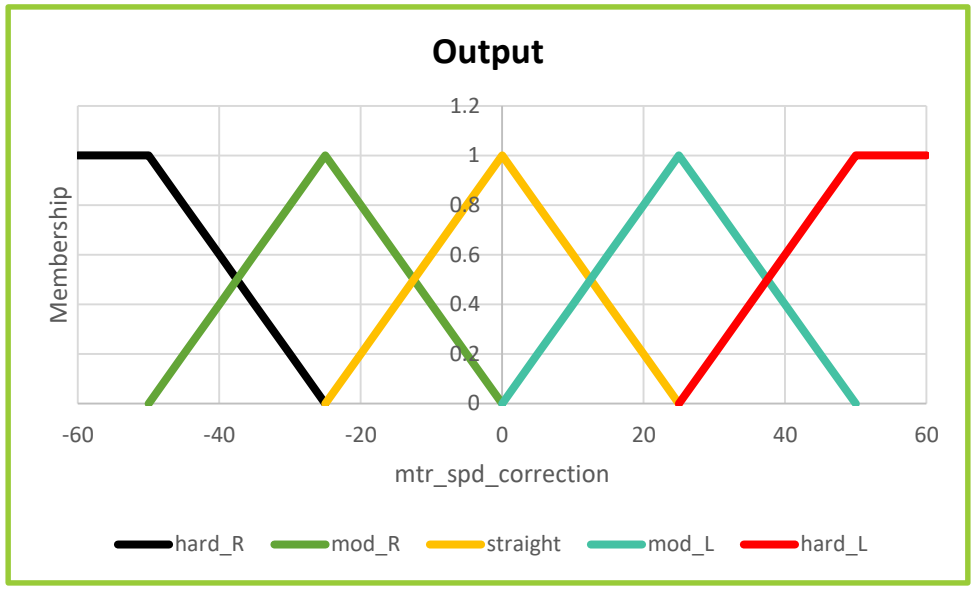
Fuzzy Linguistic Variables

- far_to_L
- leftish
- center
- rightish
- far_to_R

 Copyright © Doug Paradis • www.dprg.org

Demo: A Fuzzy Logic line following controller.

```
//Setup Membership functions for outputs (L and R motor speed settings)
// FuzzyOutput
FuzzyOutput* mtr_spd_correction = new FuzzyOutput(1);

FuzzySet* hard_R = new FuzzySet(-60,-60,-50,-25);
mtr_spd_correction->addFuzzySet(hard_R);

FuzzySet* moderate_R = new FuzzySet(-50,-25,-25,0);
mtr_spd_correction->addFuzzySet(moderate_R);

FuzzySet* straight_ahead = new FuzzySet(-25, 0, 0, 25);
mtr_spd_correction->addFuzzySet(straight_ahead);

FuzzySet* moderate_L = new FuzzySet(0, 25, 25, 50);
mtr_spd_correction->addFuzzySet(moderate_L);

FuzzySet* hard_L = new FuzzySet(25,50,60,60);
mtr_spd_correction->addFuzzySet(hard_L);

fuzzy->addFuzzyOutput(mtr_spd_correction);
```



Fuzzy Linguistic Variables

- hard_R
- moderate_R
- straight_ahead
- moderate_L
- hard_L

 Copyright © Doug Paradis • www.dprg.org

Demo: A Fuzzy Logic line following controller.

```
// Building Fuzzy Rules
        // Rule 1
        FuzzyRuleAntecedent* if_LF_sen_val_is_far_to_L = new FuzzyRuleAntecedent();
        if_LF_sen_val_is_far_to_L->joinSingle(far_to_L);
        FuzzyRuleConsequent* then_turn_hard_R = new FuzzyRuleConsequent();
        then_turn_hard_R->addOutput(hard_R);

        FuzzyRule* fuzzyRule01 = new FuzzyRule(1, if_LF_sen_val_is_far_to_L, then_turn_hard_R);
        fuzzy->addFuzzyRule(fuzzyRule01);
        //Rule 2
        FuzzyRuleAntecedent* if_LF_sen_val_is_leftish = new FuzzyRuleAntecedent();
        if_LF_sen_val_is_leftish->joinSingle(leftish);
        FuzzyRuleConsequent* then_turn_moderate_R = new FuzzyRuleConsequent();
        then_turn_moderate_R->addOutput(moderate_R);

        FuzzyRule* fuzzyRule02 = new FuzzyRule(2, if_LF_sen_val_is_leftish, then_turn_moderate_R);
        fuzzy->addFuzzyRule(fuzzyRule02);
        //Rule 3
        FuzzyRuleAntecedent* if_LF_sen_val_is_center = new FuzzyRuleAntecedent();
        if_LF_sen_val_is_center->joinSingle(center);
        FuzzyRuleConsequent* then_go_straight_ahead = new FuzzyRuleConsequent();
        then_go_straight_ahead->addOutput(straight_ahead);

        FuzzyRule* fuzzyRule03 = new FuzzyRule(3, if_LF_sen_val_is_center, then_go_straight_ahead);
        fuzzy->addFuzzyRule(fuzzyRule03);
        //Rule 4
        FuzzyRuleAntecedent* if_LF_sen_val_is_rightish = new FuzzyRuleAntecedent();
        if_LF_sen_val_is_rightish->joinSingle(rightish);
        FuzzyRuleConsequent* then_turn_moderate_L = new FuzzyRuleConsequent();
        then_turn_moderate_L->addOutput(moderate_L);

        FuzzyRule* fuzzyRule04 = new FuzzyRule(4, if_LF_sen_val_is_rightish, then_turn_moderate_L);
        fuzzy->addFuzzyRule(fuzzyRule04);
        // Rule 5
        FuzzyRuleAntecedent* if_LF_sen_val_is_far_to_R = new FuzzyRuleAntecedent();
        if_LF_sen_val_is_far_to_R->joinSingle(far_to_R);
        FuzzyRuleConsequent* then_turn_hard_L = new FuzzyRuleConsequent();
        then_turn_hard_L->addOutput(hard_L);

        FuzzyRule* fuzzyRule05 = new FuzzyRule(5, if_LF_sen_val_is_far_to_R, then_turn_hard_L);
        fuzzy->addFuzzyRule(fuzzyRule05);
}
```

Rules:

- If LF_sen_val is far_to_L then turn hard_R
- If LF_sen_val is leftish then turn moderate_R
- If LF_sen_val is center then go straight_ahead
- If LF_sen_val is rightish then go moderate_L
- If LF_sen_val is far_to_R then turn hard_L

 Copyright © Doug Paradis • www.dprg.org

Demo: A Fuzzy Logic line following controller.

```
void loop()
{
     // variables
     int16_t correction = 0;

     // get info from LF sensor array
     read_LF_sensor(ptr_LF_sensor);
     calc_LF_sensor_val(ptr_LF_sensor);

     // get correction value from Fuzzy Logic model
     fuzzy->setInput(1, ptr_LF_sensor->result);
     fuzzy->fuzzify();
     correction = (int16_t) fuzzy->defuzzify(1);

     // set motors         L                          R
     set_motors((fwd_mtr_spd + correction), (fwd_mtr_spd - correction));
}
```

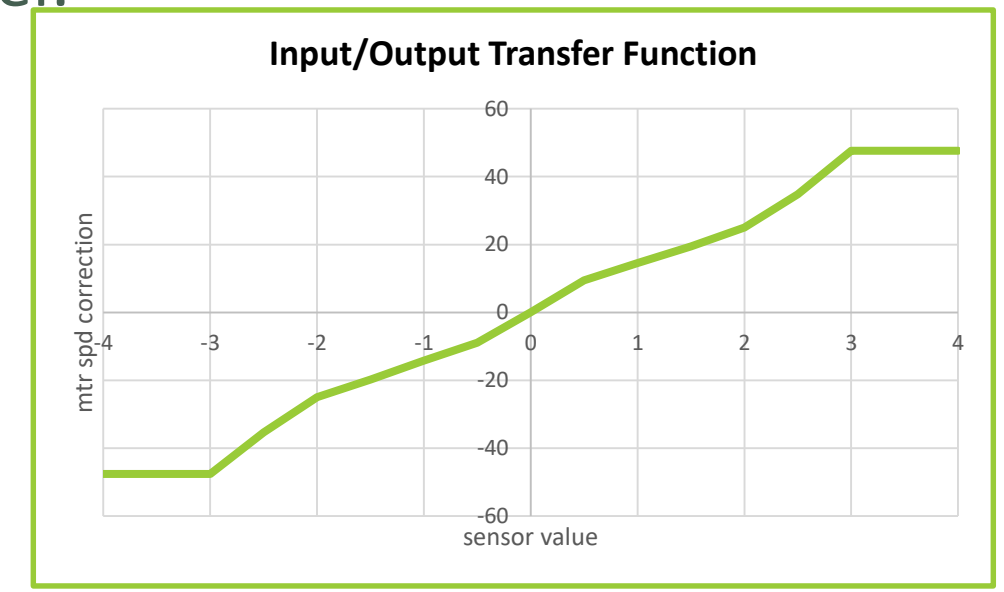
Defuzzification occurs in the loop() function.

 Copyright © Doug Paradis • www.dprg.org

Demo: A Fuzzy Logic line following controller.

Very easy to create data for a transfer function.

```
// simulation (code really rough)
float j = -4.0;

while (j < 4.2) {
if (j > 4.0) j = 4.0;

fuzzy->setInput(1, j);

fuzzy->fuzzify();

//Serial.print(off_R->getPertinence());
Serial.print(j);
Serial.print(", ");
Serial.print(fuzzy->isFiredRule(1));
Serial.print(",");
Serial.print(fuzzy->isFiredRule(2));
Serial.print(",");
Serial.print(fuzzy->isFiredRule(3));
Serial.print(",");
Serial.print(fuzzy->isFiredRule(4));
Serial.print(",");
Serial.println(fuzzy->isFiredRule(5));


float output = fuzzy->defuzzify(1);

Serial.print(j);
Serial.print(",");
Serial.println(output);


j +=0.5;

}
for(;;){};
```



Sensor Value	Rule 1	Rule 2	Rule3	Rule4	Rule5
-4	1	0	0	0	0
-3.5	1	0	0	0	0
-3	1	0	0	0	0
-2.5	1	1	0	0	0
-2	0	1	0	0	0
-1.5	0	1	1	0	0
-1	0	1	1	0	0
-0.5	0	1	1	0	0
0	0	0	1	0	0
0.5	0	0	1	1	0
1	0	0	1	1	0
1.5	0	0	1	1	0
2	0	0	0	1	0
2.5	0	0	0	1	1
3	0	0	0	0	1
3.5	0	0	0	0	1
4	0	0	0	0	1

 Copyright © Doug Paradis • www.dprg.org

Demo: A Fuzzy Logic line following controller.

Run the robot!

 Copyright © Doug Paradis • www.dprg.org