Tobias Hallen
16322468
29/11/19

CSU44001 Fuzzy Logic Report -

Acceleration of a Car Relative to its Position to a Traffic Light

https://github.com/TobiasHallen/CS4001-Car-Acceleration

Introduction:

The topic of this project is the problem of manipulating a car's acceleration relative to its position to a traffic light, as well as that traffic light's current state. With autonomous/assisted driving being a hot topic of discussion currently, I thought it would be interesting to explore one of the simpler ways in which a self-driving vehicle may respond to different traffic conditions. In this case, a stop light.

While autonomous driving is still far from being commercially and logistically adoptable in the greater market, great strides have been made in the field.
Fuzzy logic plays a large role in the field, as many of the computations involved in certain aspects of a self-driving vehicle involve some level of fuzziness. Fuzzy logic can be described as a different way of representing information. Where otherwise information is often represented as a binary set of observations, in 'True' and 'False' fashion, real world problems are not always so simple.
For example, fuzzy logic is employed in fields such as facial recognition, as there are a large number of different aspects which combine to create a human likeness. With fuzzy logic, these different factors are weighed up to determine whether or not a match is made, based on a set number of thresholds to be exceeded. In similar fashion, it has been employed in this project, though much more simply.
Fuzzy logic can be seen as representing more complicated systems, such as this, based on 'degrees of truth' rather than boolean logic.

Assumptions:

1. This is a one-car scenario. The model could be expanded to include any number of vehicles and obstacles, however not within the scope of this project.
2. If a car encounters a red light, it must stop, and only pass when the light turns green.
3. In a traffic light, Green follows Red follows Yellow.
4. The car is capable of detecting the color and position of the traffic light, relative to its own position.

Description:

The service which is modelled in this project is the acceleration and deceleration of a car at a traffic light, based on both the position of the car in regards to the light, as well as the light's current state. The system considers a simple one-way road, with a stoplight at one end. Only this simple scenario was considered in this project, as the logic applied can be carried over to more complicated road and traffic situations.
As a car approaches a stoplight, it would be preferable if the car would adhere to the following basic behaviour pattern:

1. If the car is close to the light or the light is red, decelerate.

2. If the light is at a medium distance to the light or the light is green, maintain a middling speed.
3. If the car is far from the light or the light is orange, accelerate.

In this way, the car's behaviour patterns should always follow the rules of the road, while still allowing for some leeway, not necessarily requiring the car to ever come to a full stop, which should improve both fuel and time efficiency as the car need not ever accelerate from a full stop.

Fuzzification Outline:

The goal of a fuzzy logic system is to produce a conclusion or set of conclusions in order to provide actionable results from uncertain positions. This involves certain procedures:

1. **Define Linguistic Variables:**
   Determine linguistic input and output variables involved in the system in common terms.
2. **Membership Functions:**
   For each of these variables, construct a membership function to describe them.
3. **Construct Knowledge Base Ruleset:**
   Create a matrix of input variables vs. the expected output variable. Build a set of IF-ELSE rules to describe this behaviour.
4. **Summation of Fuzzy Exit Variables:**
   Perform the evaluation of this ruleset using fuzzy logic. The operations used in fuzzy logic are Max and Min for Or and AND respectively.
5. **Defuzzification:**
   5 methods to determine the defuzzed result, computing the:
   a. Centroid (center weight)
   b. Bisector (area)
   c. Middle of maximum (average of maximum value of output set)
   d. Largest of Maximum
   e. Smallest of Maximum

   Defuzzification is the process of producing a quantifiable Crisp logic result from a fuzzy set and membership degrees. It is needed to produce the final resulting output behaviour of the system.

In this model we have the following linguistic variables:

Distance from Light	**Color of Light**	**Car Acceleration**
Near	Yellow	Slow
Middling	Red	Maintain
Far	Green	Increase

We take the crisp distance provided by the car's sensors, and categorize it into one of the three distances, which we shall call 'DfL'. The same is done for the color of the light (CoL). Now that both input variables have been fuzzied, the output must be fuzzied also. We categorize the car behaviour

into 3 pools, slowing down, maintaining acceleration, and increasing acceleration. This fuzzification can be seen in the chart below:

A similar graph exists for the color of the light.

We can now use these fuzzied variables to form a behaviour matrix:

	Yellow	**Red**	**Green**
Near	Slow / Increase	Slow / Slow	Slow / Maintain
Middling	Maintain / Increase	Maintain / Slow	Maintain / Maintain
Far	Increase / Increase	Increase / Slow	Increase / Maintain

As one can see, the solution is not so simple, as in most cases, the directives are directly opposing each other. For example:

When the light is Red and the car is Far, increase acceleration and decrease acceleration.

The rules followed by the system are:

1. If car is near OR traffic light is Red, decelerate.
2. If car is at middling distance OR traffic light is green, maintain speed.
3. If car is far from traffic light or traffic light is orange, increase acceleration.

Therefore, we must determine how much each rule affects the outcome in each possible state. To do this, fuzzy operations are applied.
For each possible output case, it must be calculated how much each input variable affects it. Each output case determines on a scale from 0-100% how much it should affect the model at any given time.

1st Case:

$$max(gaussianMF(DfL,\ tlPosition/5.5,\ 0),\ triangularMF(CoL,\ 0,\ redCol,\ redCol))$$

2nd Case:

$$max(gaussianMF(DfL,\ tlPosition/5.5,\ tlPosition/2),\ triangularMF(CoL,\ redCol,\ redCol + YelCol,\ redCol + YelCol\ + greenCol))$$

3rd Case:

$$max(gaussianMF(DfL,\ tlPosition/5.5,\ tlPosition),\ triangularMF(CoL,\ redCol + YelCol,\ redCol + greenCol,\ redCol + YelCol + greenCol))$$

These results are then conglomerated and used to determine the resulting action, i.e.

$$car\ speed\ =\ case1 * 0\ +\ case2 * 0.5\ +\ case3$$

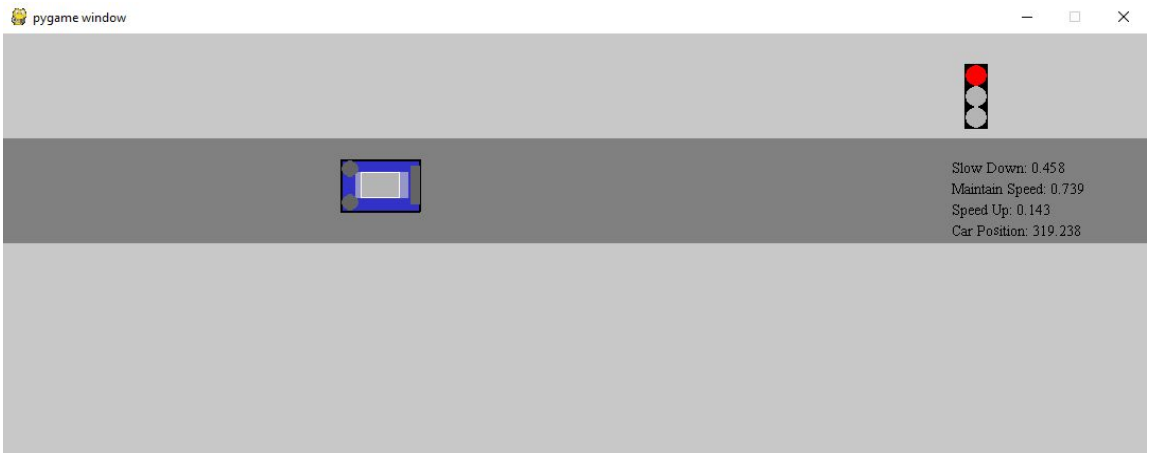
This speed is then defuzzified, dividing the figure by 3 and utilizing the centroid method mentioned above.

Simulation:

The simulation was created in Python using, in part, the skfuzzy library to perform the gaussMF function. The triangularMF function was self-implemented. The code is sectioned into 4 parts. Main.py contains all the fuzzy code and controls all aspects of the simulation, whereas the other 3 files, road.py, car.py and TrafficLight.py each contain the code structure for each of their respective objects.

The simulation is drawn using the pygame library. While the models are all exceedingly basic, it serves the purpose of displaying the effect of the fuzzy calculations on the speed and acceleration of the car. I have attached a link to the github hosting this code, which will contain some video of the simulation in action.

The car is drawn driving down a road approaching a traffic light. The traffic light is constantly changing states. When the car reaches the end of the road, its position is reset to the start. All 3 of the possible outcome states have their current weights displayed on the side of the road for convenient analyzing. The car also features 2 brake lights, which will light when the car is decelerating and turn off when the car is accelerating, and serve as another useful indicator of how the model is functioning. Below is an image, though a video does it more justice.



Conclusion:

From viewing the above model at work, one can infer that the fuzzy logic applied to the car's acceleration in relation to both the distance of the car from the traffic light, as well as the light's state, has the functional effect of allowing the car to observe the rules of the road, without introducing any overcomplicated systems. This simple fuzzy logic based system has been shown to solve this problem, and therefore has merit in this application.