Pune Institute of Computer Technology
Dhankawadi, Pune

Greg Viot's Fuzzy Cruise Controller

SUBMITTED BY

Abhishek Bhargav (41201)
Ayush Raina (41208)
Abhijeet Chavan (41216)
Class: BE-2

DEPARTMENT OF COMPUTER ENGINEERING
Academic Year 2021-22

Title: Greg Viot's Fuzzy Cruise Controller

Introduction:
Cruise control system has become a common feature in automobiles nowadays. Instead of having the driver frequently checking the speedometer and adjusting pressure on the gas pedal or the brake, cruise control system control the speed of the car by maintaining the constant speed set by the driver. Therefore, cruise control system can help reduce driver's fatigue in driving a long road trip. This paper presents the Greg Viot's Fuzzy control system behind a cruise control.

Motivation: Motivation behind Cruise control system is to maintain the speed of the car constant set by the driver using the fuzzy controller.

Objectives:

- Understand and design Greg Viot's fuzzy cruise controller.
- Implement a cruise control system to determine degree of throttle.

Software and Hardware Packages:

- Jypyter Notebook
- Python 3.8
- Matplotlib
- OS: Ubuntu 20.04 (64 bits)

Block Diagram

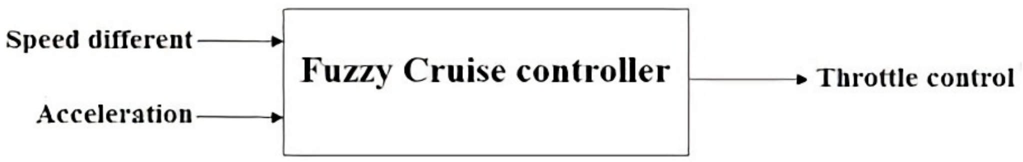


Figure 1: Greg Viot's fuzzy cruise controller

Theory:

G-V controller is used to maintain a vehicle at the desired speed this system consists of two fuzzy inputs namely speed ditference and acceleration and one fuzzy output, namcly throtle control. as shown in *figure 1.*

Degree of Membership for fuzzy set of input and output is calculated by membership Function given below.

$$degree\ of\ membership = \begin{cases} 0 & \text{if } delta < 0\ or\ delta \leq 0 \\ min(delta1 * slope1,\ delta2 * slope2) & \text{otherwise} \end{cases}$$

$$where,\ delta1\ =\ x - point1\, delta2\ = point2 - x$$

And, graphically represented as

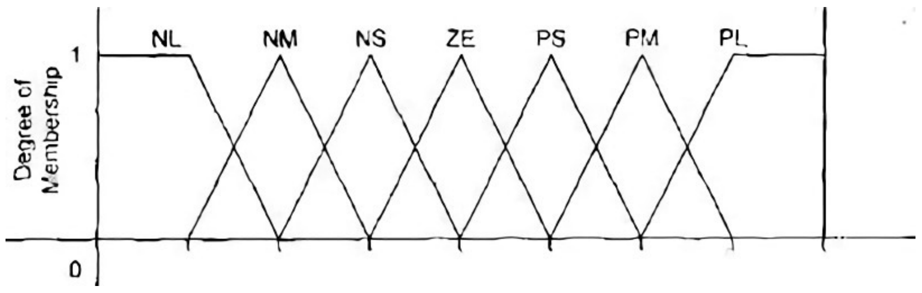


Figure 2: Memership function

Here, ZE = Zero, PS, PM, PL are positively small, medium and large respectively also NS, NM and NL represent negatively small, medium and large respectively.

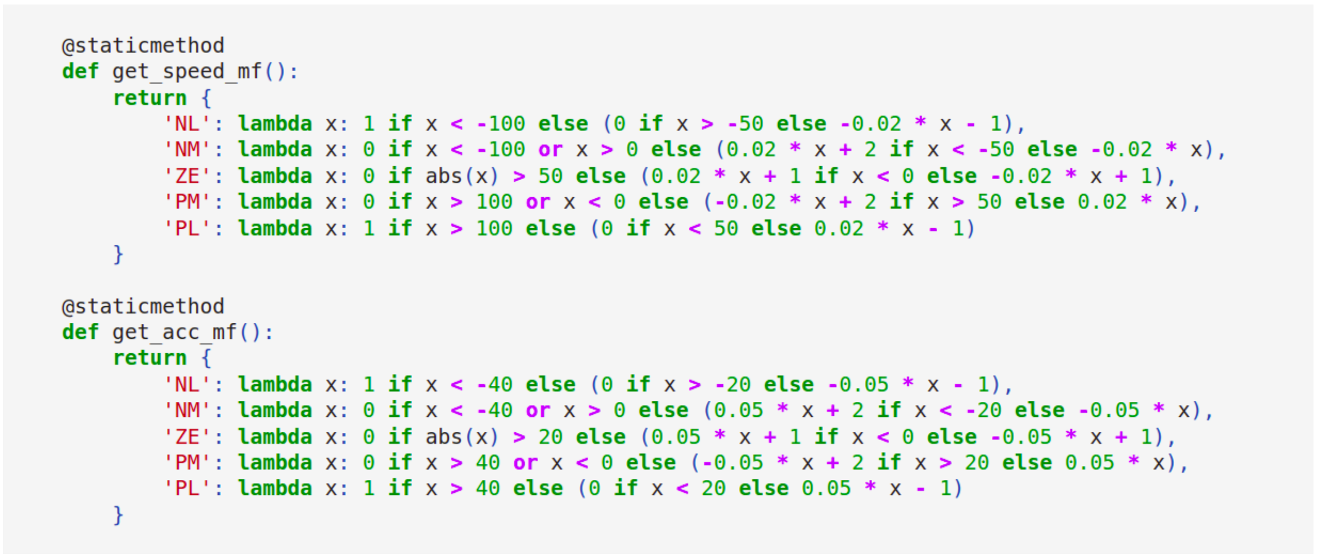
Rule base consider in this controller are as follows:

Rule 1	If (speed difference is NL) and (acceleration is ZE) then (throttle control is PL)
Rule 2	If (speed difference is ZE) and (acceleration is NL) then (throttle control is PL)
Rule 3	If (speed difference is NM) and (acceleration is ZE) then (throttle control is PM)
Rule 4	If (speed difference is NS) and (acceleration is PS) then (throttle control is PS)
Rule 5	If (speed difference is PS) and (acceleration is NS) then (throttle control is NS)
Rule 6	If (speed difference is PL) and (acceleration is ZE) then (throttle control is NL)
Rule 7	If (speed difference is ZE) and (acceleration is NS) then (throttle control is PS)
Rule 8	If (speed difference is ZE) and (acceleration is NM) then (throttle control is PM)

Implementation and Results:

1. Input(s):
   - $speed\ difference \in [-100,\ 100]$
   - $acceleration \in [-40,\ 40]$
2. Output(s):
   - $throttle \in [-20,\ 20]$

3. Implementation of speed difference & acceleration membership functions:

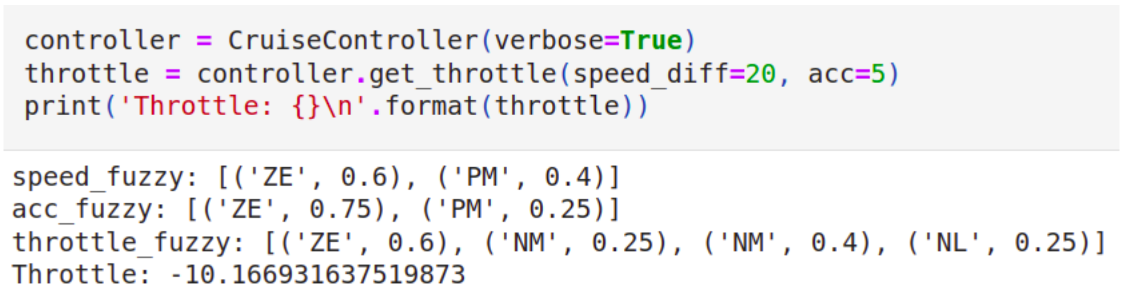


```python
@staticmethod
def get_speed_mf():
    return {
        'NL': lambda x: 1 if x < -100 else (0 if x > -50 else -0.02 * x - 1),
        'NM': lambda x: 0 if x < -100 or x > 0 else (0.02 * x + 2 if x < -50 else -0.02 * x),
        'ZE': lambda x: 0 if abs(x) > 50 else (0.02 * x + 1 if x < 0 else -0.02 * x + 1),
        'PM': lambda x: 0 if x > 100 or x < 0 else (-0.02 * x + 2 if x > 50 else 0.02 * x),
        'PL': lambda x: 1 if x > 100 else (0 if x < 50 else 0.02 * x - 1)
    }

@staticmethod
def get_acc_mf():
    return {
        'NL': lambda x: 1 if x < -40 else (0 if x > -20 else -0.05 * x - 1),
        'NM': lambda x: 0 if x < -40 or x > 0 else (0.05 * x + 2 if x < -20 else -0.05 * x),
        'ZE': lambda x: 0 if abs(x) > 20 else (0.05 * x + 1 if x < 0 else -0.05 * x + 1),
        'PM': lambda x: 0 if x > 40 or x < 0 else (-0.05 * x + 2 if x > 20 else 0.05 * x),
        'PL': lambda x: 1 if x > 40 else (0 if x < 20 else 0.05 * x - 1)
    }
```

Figure 3: Membership function methods

4. Results



```python
controller = CruiseController(verbose=True)
throttle = controller.get_throttle(speed_diff=20, acc=5)
print('Throttle: {}\n'.format(throttle))
```

```
speed_fuzzy: [('ZE', 0.6), ('PM', 0.4)]
acc_fuzzy: [('ZE', 0.75), ('PM', 0.25)]
throttle_fuzzy: [('ZE', 0.6), ('NM', 0.25), ('NM', 0.4), ('NL', 0.25)]
Throttle: -10.166931637519873
```

Figure 4: Throttle position adjustment

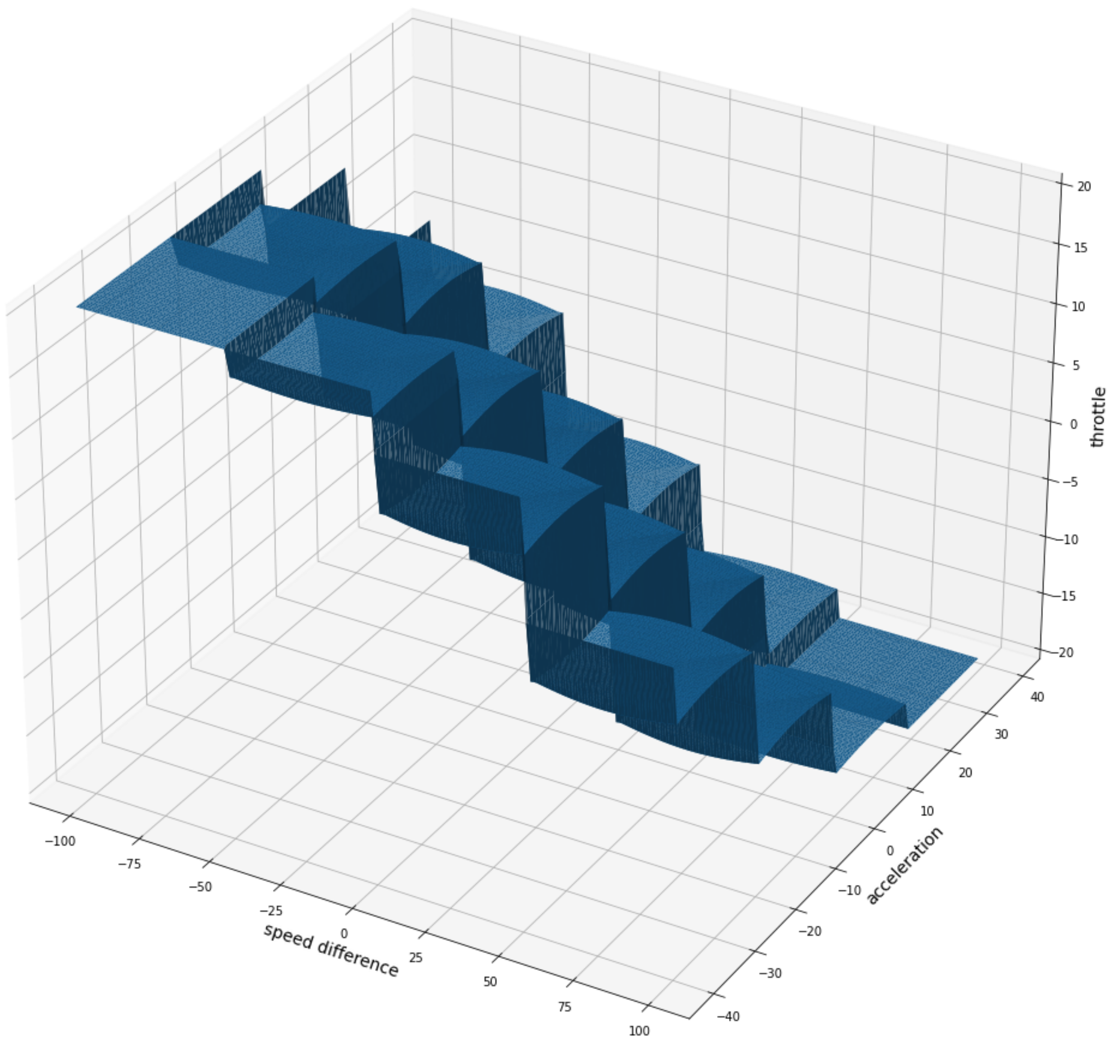


Figure 5: Throttle position based of speed difference, accelration

Conclusion:
With this study, we have demonstrated the application of fuzzy logic for designing Greg Viot's fuzzy cruise controller and observed the prediction of throttle adjustment for varying speed difference and instantaneous acceleration.