Cruise control system design using fuzzy logic

Tsegazeab Shishaye [2012420012]

Northwestern polytechnical university, Xi'an, China

Abstract

In this paper a vehicle speed control using PI and PD fuzzy logic is modeled, simulated and checked for accuracy. A clear and precise step by step design was used for better understandability. First, the dynamics of the system were modeled as a subsystem in simulink, then the overall control system for both the PI and PD controllers are modeled using Matlab simulink blocks and appropriate values. Finally, the fuzzy controller is designed carefully. All the simulation outputs are put and checked according to the design specifications.

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 control system behind the cruise control.

Tasks

a) Designing controller using fuzzy logic and

b) Making the automobile's speed keep constant

Model description of the automobile:

The dynamics of the automobile are given as follows:

$$\dot{v} = \frac{1}{m}\left(-A_p v^2(t) - d + f(t)\right)$$

$$\dot{f} = \frac{1}{\tau}\left(-f(t) + u(t)\right)$$

Where

u - control input ($u > 0$ represents throttle input and $u < 0$ represents brake input)

m - 1300kg is mass of the vehicle

A_p - 0.3 Ns^2/m^2 is its aerodynamic drag

d - 100 N is a constant frictional force

f - is the driving/braking force

$\tau = 0.2sec$ is saturated at $\pm 1000N$

Fuzzy control method can be used to design a cruise control system. Obviously, the fuzzy cruise control design objective is to develop a fuzzy controller that regulates a vehicle's speed $v(t)$ to a driver-specified value $v_d(t)$.

1. Designing of PI fuzzy controller

A "PI fuzzy controller" can be used as shown in Figure to track a step or ramp change in the driver specified speed value $v_d(t)$ very accurately.

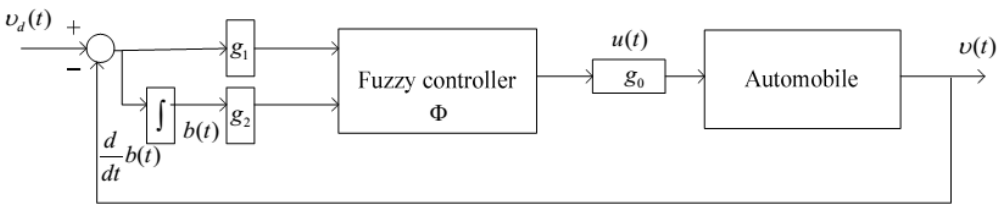

Fig. 1 Speed control system using a PI fuzzy controller

g_0, g_1, g_2 - are scaling gains

$b(t)$ - is the input of the integrator

Procedure of design:

- Finding the differential equation that describes the closed-loop system
- Letting the state be $x = [x_1, x_2, x_3]^T = [v, f, b]^T$ and
- finding a system of three first-order ordinary differential equations that can be used by the Runge-Kutta method in the simulation of the closed-loop system

For the reference input, three different test signals can be used as follows:

a: Test input 1 makes $\upsilon_d(t)$=18m/sec (40.3 mph) for $0 \le t \le 10$ and $\upsilon_d(t) = 22$ m/sec (49.2 mph) for $10 \le t \le 30$.

b: Test input 2 makes $\upsilon_d(t)$=18m/sec (40.3 mph) for $0 \le t \le 10$ and $\upsilon_d(t)$ increases linearly (a ramp) from 18 to 22 $m/\sec$ by $t = 25\sec$, and then $\upsilon_d(t) = 22$ for $25 \le t \le 30$.

c: Test input 3 makes $\upsilon_d(t)$=22 for $0 \le t$ and we use $x(0)$ as the initial condition (this represents starting the vehicle at rest and suddenly commanding a large increase speed).

Use $x(0) = [18, 197.2, 20]^T$ for test input 1 and 2.

Design specifications:

For the jump from 18 to 22 m/sec in "test input 1"

- Designing fuzzy controller to get less than 2% overshoot
- A rise time between 5 and 7 sec
- A settling time less than 8 sec

Also for the ramp input ("test input 2")

- it must have less than 1mph (0.447 m/sec) steady state error
- fully specify the controller (i.e. the membership functions, rule base defuzzification, etc)
- simulate the closed loop system to demonstrate its performance
- provide plots of $v(t) and\ v_d(t)$ on the same axis and $u(t)$ on a different plot

For test input 3

- Find the rise time, overshoot, 2% settling time, and steady state error for the closed loop system for the controller designed to meet the specifications of test input 1 and 2.

Step by step design

Taking the dynamics equations:

$$\dot{v} = \frac{1}{m}\left(-A_p v^2(t) - d + f(t)\right)$$

$$\dot{f} = \frac{1}{\tau}\left(-f(t) + u(t)\right)$$

$$b(t) = v_d(t) - v(t)$$

The dynamics equations can be put in simulink using Matlab function blocks and can be linked to an M-file written fuction which represent the non-linear dynamics equations using the M-file's name. they will be ready for the ode4(Runge-Kutta) solver.

The first two equations which represent the dynamics of the vehicle can be modeled in **Matlab script** as follow:

function for the driving/braking force, (saved as **force.m**)

```
function [fdot]=force(in)

% u = array of two inputs: the force and the controlling signal
% fdot=df/dt=5*(u(t)-f(t))

fdot=5*(in(1)-in(2));

end
```

Function for the velocity (saved as **velocity.m**)

```
function [vdot]=velocity(in)

% the array of inputs consists of the velocity and force
% vdot=dv/dt=(f(t)-0.3*v^2(t)-100)/1300

vdot=(in(1)-0.3*in(2)*in(2)-100)/1300;

end
```

So, the following simulink block is a complete model of for the dynamics of the vehicle and will be injected in to the total control system as a sub-system named "**Automobile**".
Here, the initial conditions of the system $v0 = 18$, $f0 = 197.2$ and $b0 = 20$ are used as given previously.

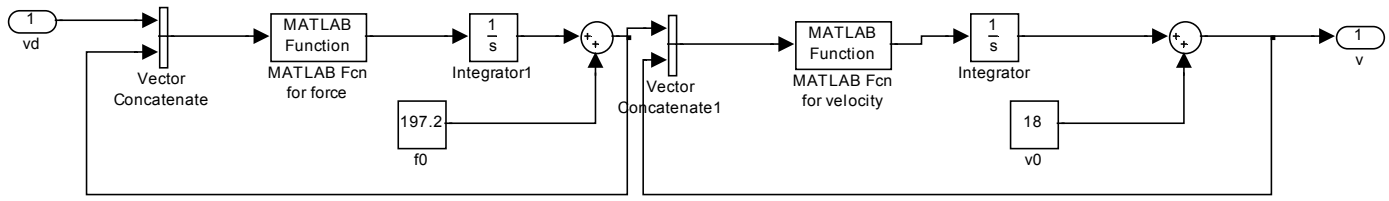

Fig.2. Simulink representation of the non-linear dynamics of the system

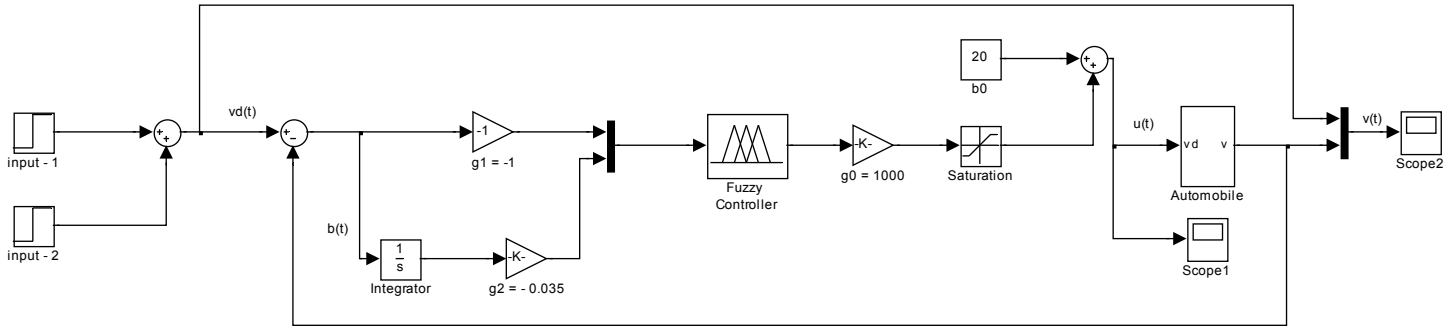

Fig.3. Simulink model for Speed control system using a **PI** controller

Designing fuzzy controller

To design the fuzzy controller, two inputs are considered – the error $\boldsymbol{b = v_d - v}$ and its derivative $\boldsymbol{\dot{b}}$. Appropriate ranges of both values are considered and the following membership functions and rules are developed.

E ED	LN	MN	ZERO	MP	LP
LN	AVHP	AVHP	AHP	AHP	AMP
MN	AVHP	AHP	AMP	AMP	ASP
ZERO	ASP	ASP	ANO	ASN	ASN
MP	ASN	AMN	AMN	AHN	AHN
LP	AMN	AHN	AHN	AVHN	AVHN

Fig.4. Tabular representation of Fuzzy rules

Description

LN=Large Negative

MN=Medium Negative

Zero=Zero

MP=Medium Positive

LP=Large Positive

AVHP = Add Very High Positive

AHP = Add High Positive

AMP = Add Medium Positive

ASP = Add Small Positive

ANO = Add No

ASN = Add Small Negative

AMN = Add Medium Negative

AHN = Add High Negative

AVHN = Add Very High Negative

The membership functions for each input, the rules, and the corresponding overall fuzzy surface are presented below.

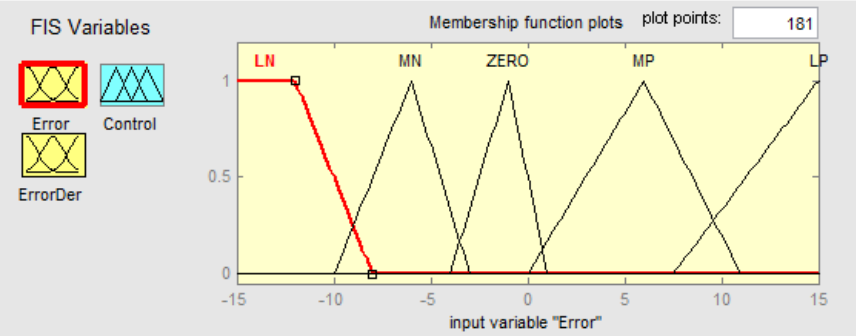

Fig.5. Membership functions of error (input-1)

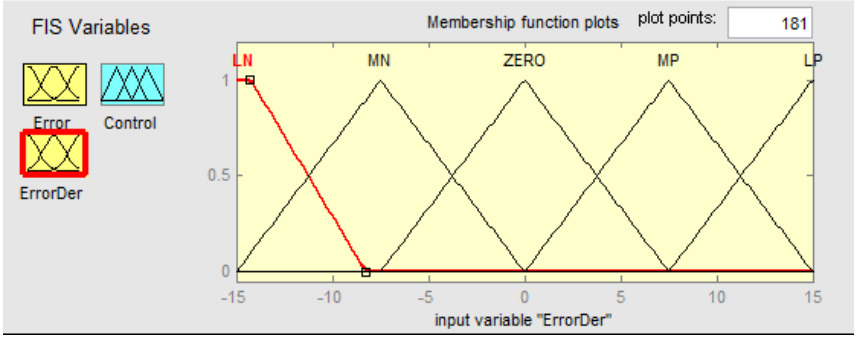

Fig.6.Mmembership functions of error derivative (input-2)

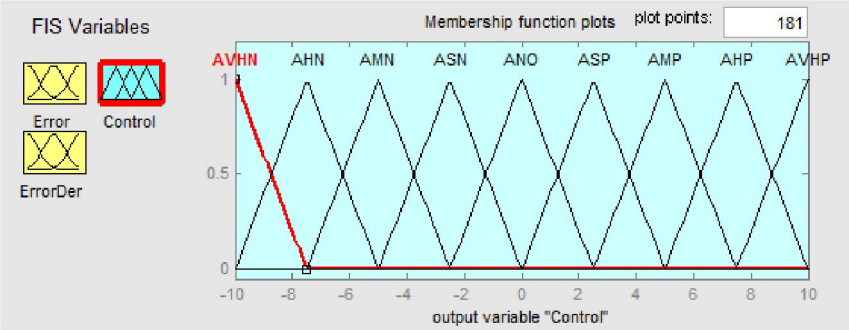

Fig.7. Membership functions of the **control** output

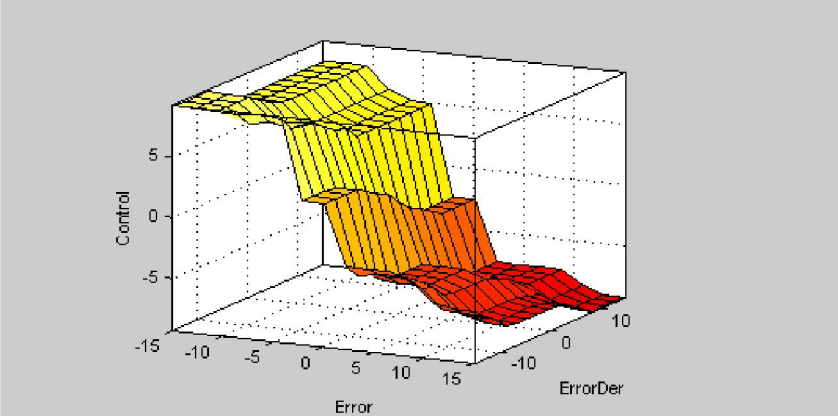

Fig.8. surface look of the fuzzy logic

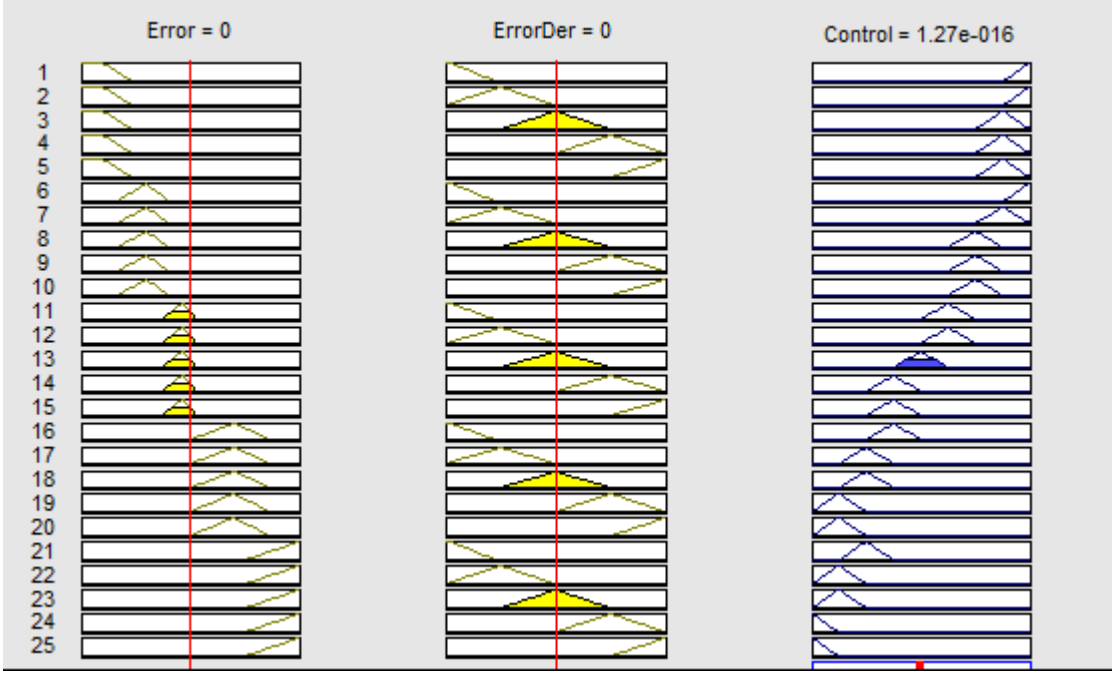

Fig.9. Plot of the fuzzy logic rules

Simulated outputs of the system

a. For test input – 1

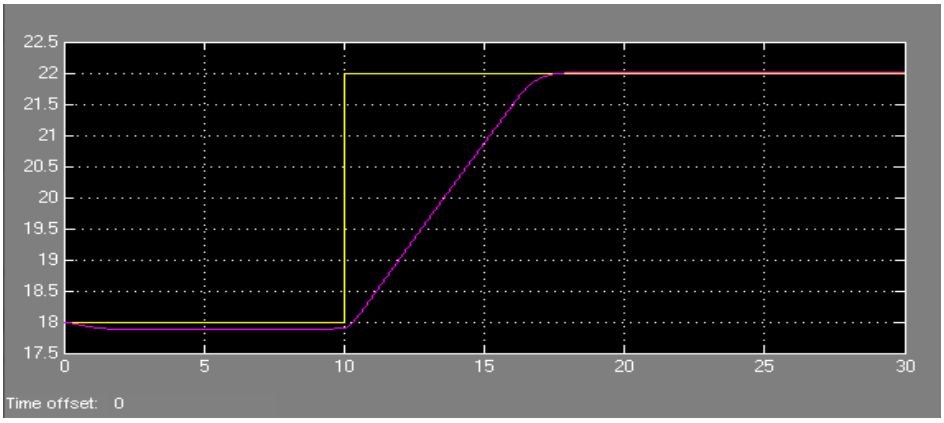

Fig.10. desired speed (yellow) and vehicle speed (purple)

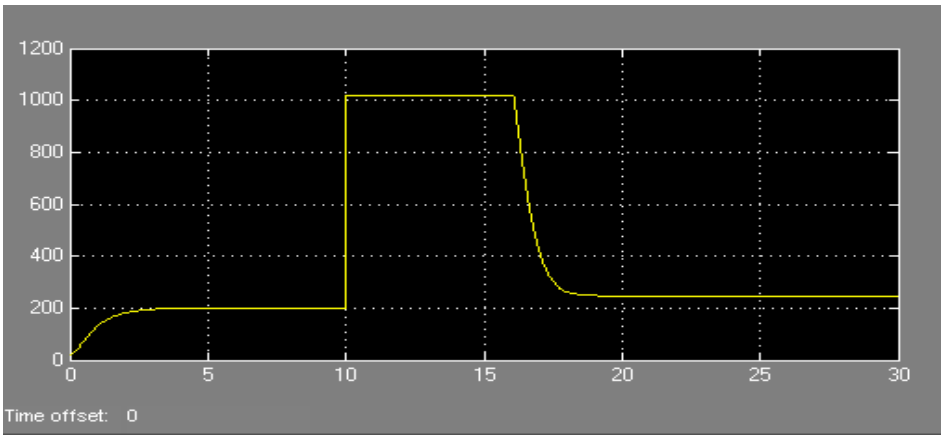

Fig.11. output of fuzzy controller (input to the vehicle)

b. For test input – 2

This can be done by applying two ramp input, one with positive and the other with negative slopes, instead of input-2 shown in figure 3.

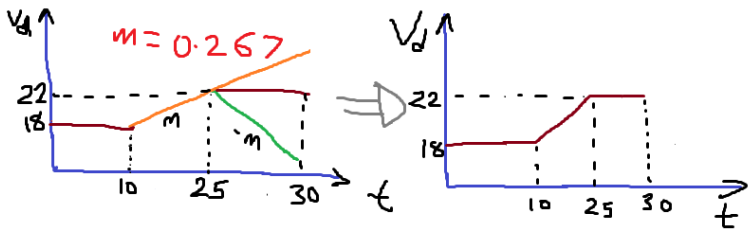

Fig.12. Explanation of the applied inputs (**1** step and **2** ramp inputs)

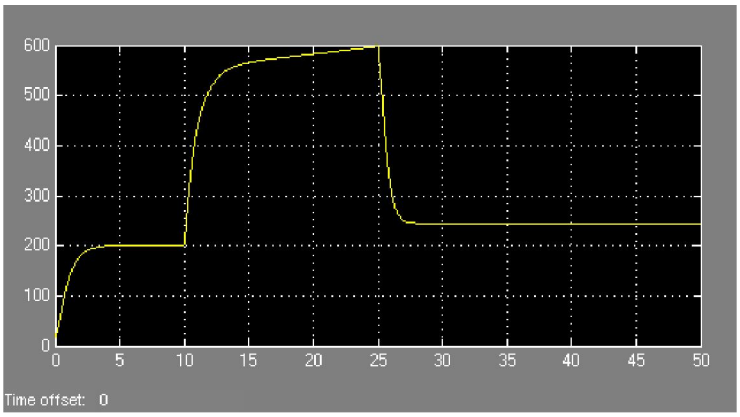

Fig.13. Output of fuzzy controller (input to the vehicle)

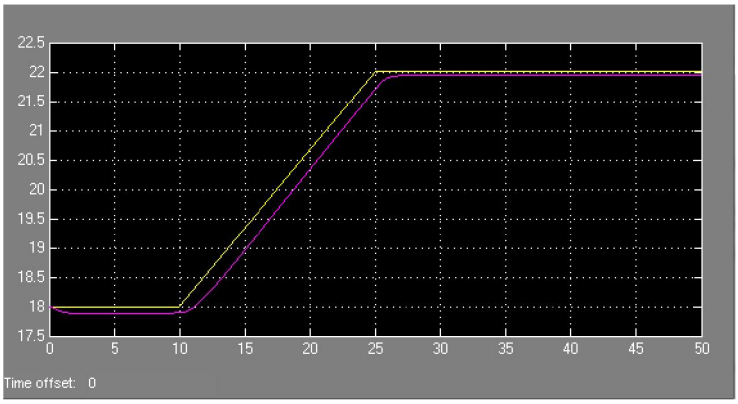

Fig.14. vehicle speed (purple) and desired speed (yellow)

c. For test input – 3

Applying only one step input to the system as specified above, the following outputs are found.

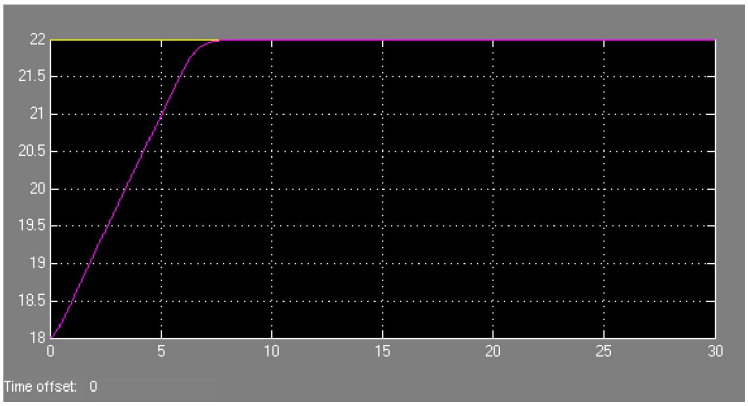

Fig.15. desired speed (yellow) and vehicle speed (purple)

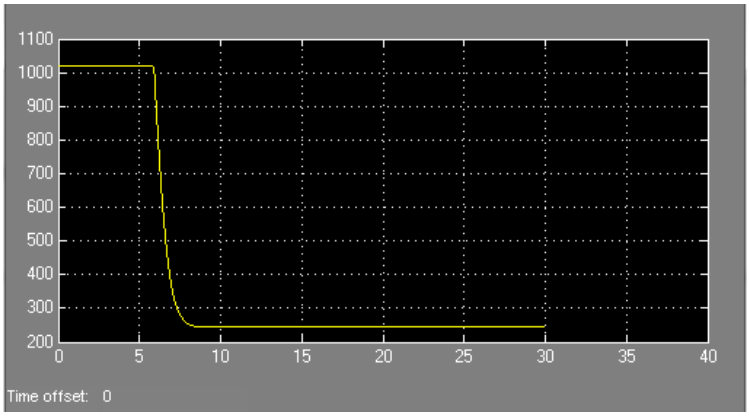

Fig.16. output of fuzzy controller (input to the vehicle)

N.B. **All the above results from the three test inputs fulfill all the design specifications, means, overshoot, settling time, rise time etc!**

1. Designing PD fuzzy controller

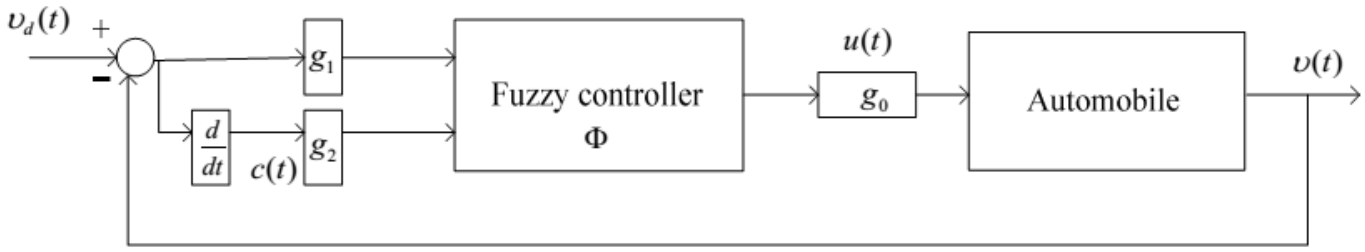

Fig.17. Speed control system using a PD fuzzy controller

When we are concerned with tracking a step change in $v_d(t)$ accurately, we use the fuzzy controller shown in Fig.2. to represent the derivative we can simply use a backward difference.

$$c(t) = \frac{e(t) - e(t-h)}{h}$$

Where h is the integration step size in your simulation (or it could be your sampling period in an implementation).

Design specifications:

- < 2% overshoot, a rise-time between 7 and 10 sec, and a settling time of less than 10 sec for test input 1
- Also, for the ramp input (test input 2 in 1)) it must have less than 1 mph steady-state error to the ramp (i.e., at the end of the ramp part of the input, have less than 1 mph error).

- Fully specifying the controller and simulate the closed-loop system to demonstrate that it performs properly
- provide plots of $v(t) and\ v_d(t)$ on the same axis and $u(t)$ on a different plot

In the simulations, the Runge-Kutta method is used and an integration step size of 0.01.

Assume that $x(0) = [18, 197.2]^T$ for test inputs 1 and 2 (hence we ignore the derivative input in coming up with the state equations for the closed-loop system and simply use the approximation for c(t) that is shown above so that we have a two-state system). As a final test let $x(0) = 0$ and use test input 3 defined in 1).

Only the integral error part is changed in to a derivative form. Below is the modified simulink model for the PD controller.

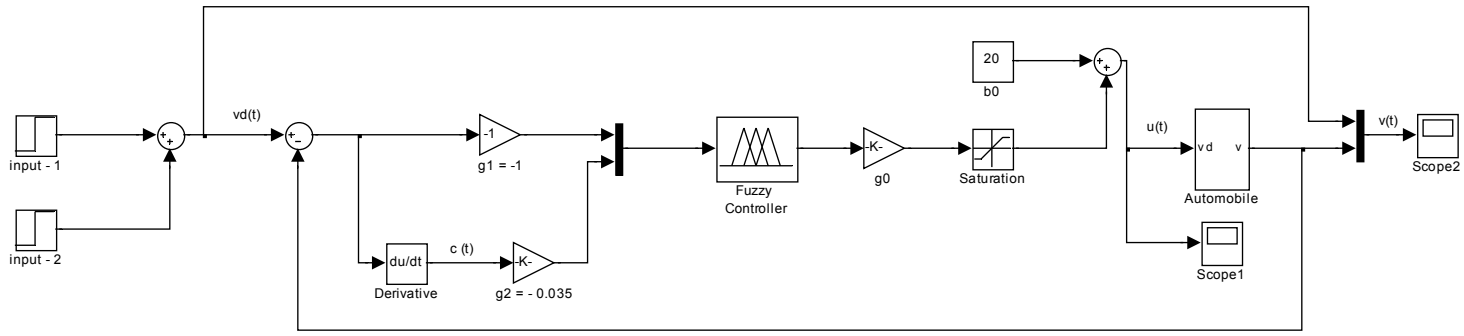

Fig.18. speed control system using **PD** fuzzy controller

a. output for test input – 1

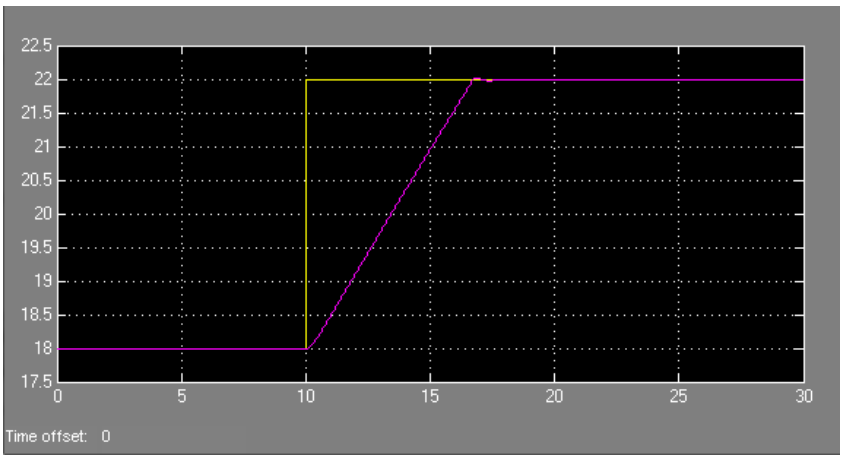

Fig.19. vehicle speed (purple) and desired speed (yellow)

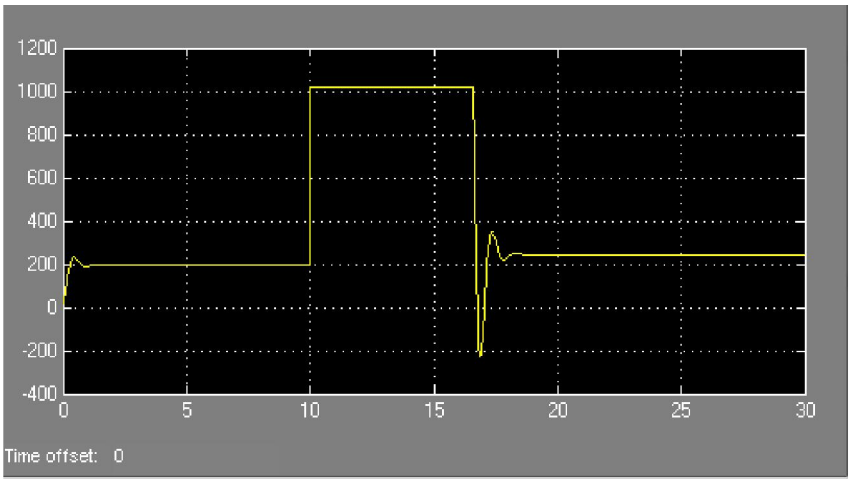

Fig.20. Output of fuzzy controller (input to vehicle)

b. for test input – 2

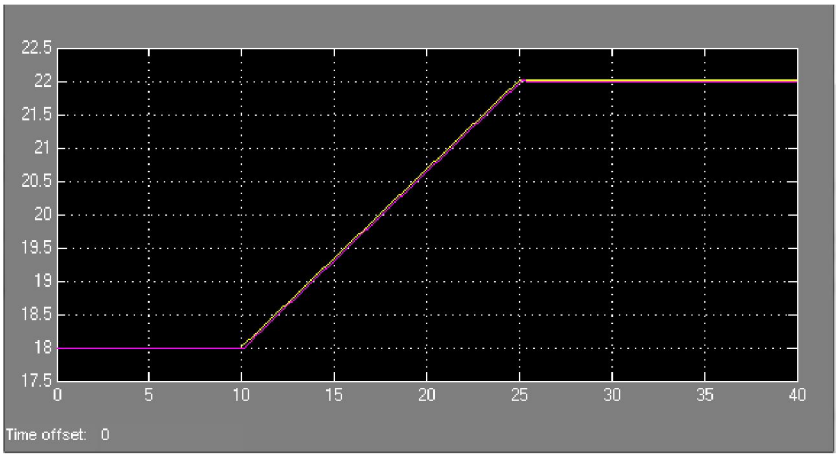

Fig.21. vehicle speed (purple) and desired speed (yellow) [they are perfectly coincided]

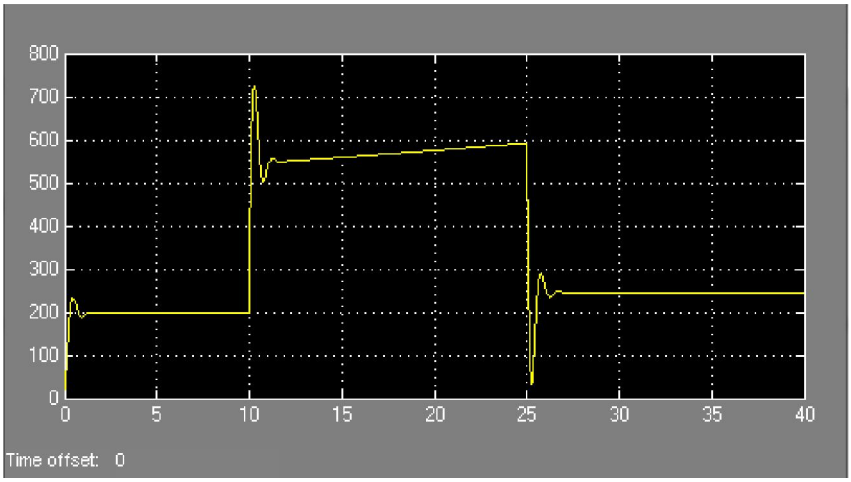

Fig.22. fuzzy controller output (vehicle input)

c. for test input – 3

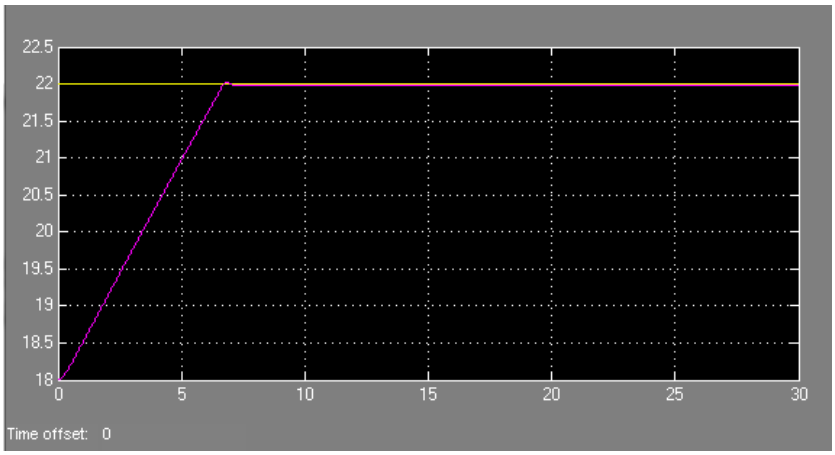

Fig.23. Vehicle speed (purple) and desired speed (yellow)

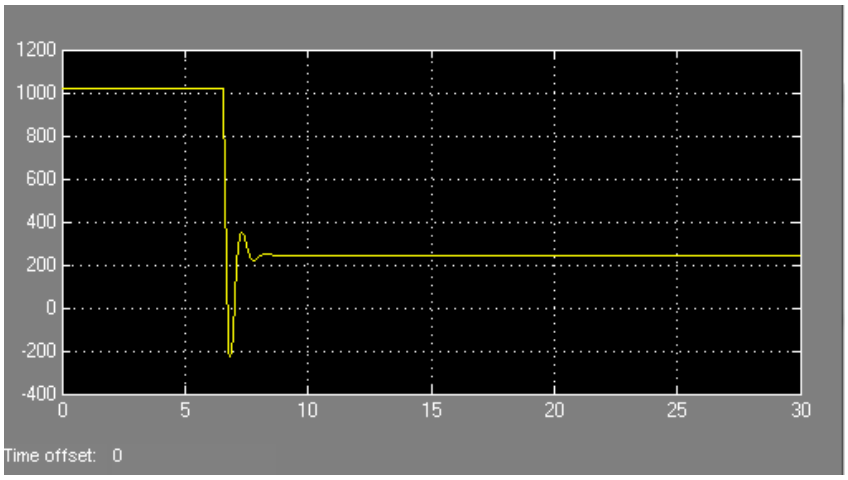

Fig.24. fuzzy controller output (vehicle input)

N.B All the outputs of the simulink model satisfy all the design specifications!

Conclusion

This moderate project tried to show that how it is easy and efficient to use the application of fuzzy logic for cruise control. It can be observed that how the inputs affect the system and how the gains can be adjusted to track the desired value. Considering the error and its derivative (or its integral) and designing a fuzzy controller with appropriate ranges can save too much energy and can produce a satisfactory results as it has been tried to be shown in the whole document.

Reference

[1] L. A. ZADEH, “Fuzzy Algorithms”, INFORMATION AND CONTROL 12, 94-102 (1968)

[2] Kevin M. Passino, Stephen Yurkovich, “Fuzzy Control”, 1998 Addison Wesley Longman, Inc.

[3] L. A. ZADEH, “Fuzzy sets*”, Information and control 8,338-353(1965)

[4] Robert E. King, “Computational Intelligence in control Engineering”,

[5] Karen Lie, “Cruise control system in Vehicle”, Calvin College

[6] Li-xing Wang, “A course in Fuzzy systems and Control”, Prentice-Hall International, Inc.

[7] Matlab Help and Simulink demos