ASSIGNMENT COVER SHEET

ANU College of Computer Science and Engineering
Australian National University
Canberra ACT 0200 Australia
www.anu.edu.au

Submission and assessment is anonymous where appropriate and possible. Please do not write your name on this coversheet.

This coversheet must be attached to the front of your assessment when submitted in hard copy. If you have elected to submit in hard copy rather than Turnitin, you must provide copies of all references included in the assessment item.

All assessment items submitted in hard copy are due at 5pm unless otherwise specified in the course outline.

Student ID For group assignments, list each student's ID	U6516811		
Course Code	ENGN6223		
Course Name	Control System		
Assignment number			
Assignment Topic	Design Project		
Lecturer	Guodong Shi		
Tutor			
Tutorial (day and time)			
Word count	3409	Due Date	17/09/2018
Date Submitted	16/09/2018	Extension Granted	

I declare that this work:

- ☐ upholds the principles of academic integrity, as defined in the University Academic Misconduct Rules;
- ☐ is original, except where collaboration (for example group work) has been authorized in writing by the course convener in the course outline and/or Wattle site;
- ☐ is produced for the purposes of this assessment task and has not been submitted for assessment in any other context, except where authorised in writing by the course convener;
- ☐ gives appropriate acknowledgement of the ideas, scholarship and intellectual property of others insofar as these have been used;
- ☐ in no part involves copying, cheating, collusion, fabrication, plagiarism or recycling.

An Introduction of Cruise Control System

Chaohuo Wu

***Abstract*—This paper presents the design process of the cruise controller and briefly analysis the performance of different control approaches including P, PI, and PID. A robust cruise control system is critical not only in automobile fields but also in manufacturing fields and the aerospace industry. Using MATLAB to test the outcome of different control methods can visually and directly validate the feedback control model.**

***Keywords*—PID, Non-linear systems, Cruise controller, Matlab, Feedback**

I. INTRODUCTION

Vehicle cruise control systems are commonly used to automatically control the car speed. It maintains the desired speed of the car over the travel. The output of the system, which is speed controlled by the people, should reach the desired speed in a limited error range, even though there are oscillations at the beginning. In this design, the control method mainly used is PID control, which is a classical feedback control method and widely used in industrial control systems. This design firstly models the cruise control using Newton's second law of motion and fluid dynamics. Practically, many systems are nonlinear, which might have numerous equilibrium points. Therefore, by finding equilibrium points, the equations of this model can be linearized. In the second part of this paper, the how to proceed cruise controller design is presented, which clearly describes the design process and approaches. Besides, validation, shown in the third part of this paper, is required in this design project to guarantee that requirements are fully fulfilled, and the system works in reality. In this part, we examine the dynamics of changing from one target speed to another and add a constant disturbance force caused from uphill slope into this designed controller. Moreover, this paper also discusses and analyzes the result of the designed controller in the next part. The final part is to make conclusions for this design project.

II. MODELING THE CRUISE CONTROL SYSTEM

A. *Model the free body diagram and EOMs*

To model the cruise control system, it is important to find some technical specifications of a car as it is a practical design project. In this paper, we model the BMW 335i and design the cruise controller using the technical information from this car. Shown in Figure 1, the free body diagram of a car, let F be the driving force of a car and f be the air resistance. *v(t)* means the car displacement as time goes. And there is the gravity of the vehicle and the support force from the ground. Here we simply assume the friction between the car and the road is zero. Also, it is noted that this design initially ignores the disturbance force resulting from variations in the uphill slope, which means that the car is on the flat road. However, we will introduce the disturbance force in the final step of design since it will test the robustness of the controller.

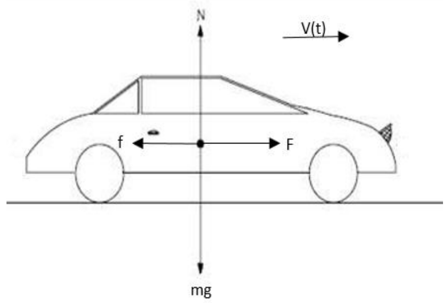

Figure 1: Free body diagram of the car

According to Newton's second law of motion, we can write the equations of motion (EOMs) for the car, starting with a force balance. Let *v* be the speed of the car, *m* the gross vehicle mass. Therefore, the EMOs is simply

$$m\frac{dv}{dt} = F - f \qquad (1)$$

The *F* is the driving force generated by the engine and *f* is the aerodynamic drag, which is proportional to the square of the speed:

$$f = \frac{1}{2}A\rho c_D v^2 \qquad (2)$$

where ρ is the density of air in kg/m^3 , c_D is the dimensionless aerodynamic drag coefficient and A is the cross-sectional area of the car in m^2(looking from the front). Looking through [1], we can obtain parameter ρ= 1.2041, c_D = 0.3, m = 2020kg and A = 2.1m^2

To summarize, we find that the car can be modeled by

$$m\dot{v} + \frac{1}{2}A\rho c_D v^2 = F + d \qquad (3)$$

where d, which represents constant disturbance, is zero initially in this paper and when finishing the controller design, we will investigate how the controller is influenced in terms of unknown disturbance.

B. *Linearize the equations of motion*

Obviously, the above equation is non-linear as it has square and therefore, we need to linearize this equation. This behavior is to investigate the local behavior of a system, where the non-linear effects are expected to small. In this section, we are currently discussing how to locally approximate a system by its linearization. Firstly, we rearrange the equations and make the $m\dot{v}$ equal to zero, which can find the equilibrium points of the equations. Through calculating, we can know that any point can be equilibrium points in this system model. Hence, it is assumed that (V_0, F_0) is an equilibrium point used in this model, where V_0 = 100m/s and F_0 is approximately equal to 3600. So, using the following linearize equation from [2]:

$$f(v,F) \approx f(V_0,F_0) + \frac{\partial f}{\partial v}(V_0,F_0)(V-V_0) + \frac{\partial f}{\partial F}(V_0,F_0)(F-F_0) \qquad (4)$$

we eventually can get a linearized equation of the system model:

$$m\dot{\delta_v} = F - A\rho c_D v_0 \delta_v - F_0 \quad (5)$$

where δ_v is the difference value of the velocity of the car. Now, it is better to change the equation variable to make the equation more understandable. Hence, we replace the δ_v with x. Moreover, it is computable for the coeffects of δ_v with the parameters we described above. Eventually, we can rearrange it and approximately write the linearized equation of this model:

$$2020\dot{x(t)} + 72x(t) = u(t) \quad (6)$$

where *u(t)* is the input of this system, representing the sum of the input F and F_0. Currently we have the linear system model so that we can proceed to the next step, the controller design.

C. *Introduction of PID control*

Before developing a cruise control, it is critical to understand what the PID feedback control is. In this section, we will briefly introduce the PID control mainly using in this project. PID control is a proportional-integral-derivative controller, which is a control loop feedback mechanism widely used in the industrial control system and a variety of other applications requiring continuously modulated control [3]. A PID controller continuously calculates an error value e(t) as the difference between the desired setpoint and a measured process variable. The overall control function can be expressed mathematically as:

$$u(t) = K_p e(t) + K_i \int_0^t e(T)dT + K_d \frac{de(t)}{dt} \quad (7)$$

where K_p , K_i , and K_d , all non-negative, denote the coefficients for the proportional, integral, and derivative terms respectively.

PID control family basically consists of P, PI, PD, and PID control, and these control methods will be selected according to the system. What important is to understand the function of each approach. using integral control is to correct for steady-state error and derivation controller is utilized to control oscillation. P control is mainly control method, which usually eliminates the major part of error [4].

III. Design Process

In this section, this article demonstrates the entire design process beginning from the P control and finalizing the PI control method for this cruise control system. It is an iterated process to choose which control approach is appropriate to make the system robust even though there is a disturbance. Using MATLAB here is to show the performance of designed controller and further verify the whether or not the controller meets demands of the specification of the car, for example, rise time, overshoot. Concrete performance of the vehicle that needs to be achieved is shown in Table 1.

TABLE 1
BMW 335I SPECIFICATIONS

Symbol	Quantity	value
t_r	*Risetime (0-100km/h)*	*>4.04 s*
M_p	*Overshoot*	*< 5%*
t_s	*Settling Time*	*>5.1 s*
k	*Steady-state Error*	*<1%*
T_{speed}	*Top Speed*	*250 km/h*
M	*Mass*	*2020 Kg*
M_{Torque}	*Max Torque*	*400Nm*
E_{speed}	*Engine Speed*	*1200-5000rpm*

A. *Proportional control*

Initially, we start designing from P control, which is the fundamental step for PID control. The P control is also named proportional control majorly used to correct the error between the set value and process value. Obviously, this is the feedback control continuously feeding the output of the system back to the input of the system with the error between the current output value and the reference value. Visually shown in Figure 2.

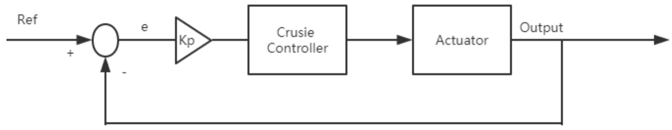

Figure 2: Proportional control dynamic model

Knowing from the figure 2, the output of a proportional controller is the multiplication product of the error signal and the proportional gain Kp. Therefore, combining with the equation 6, the mathematical model of cruise controller using P control is expressed as:

$$2020\dot{x(t)} + (72 + K_p)x(t) = K_p r(t) \quad (8)$$

where *r(t)* is reference speed set by customers and *x(t)* is the present velocity of vehicles. It is apparent that equation 8 is a linear system and we can obtain K_p by specifying reference speed and specifications shown in Table 1. In this section, reference speed is 100km/h. Through calculating, we can obtain the K_p value, which is at least greater than 962 and then using MATLAB to simulate the performance of the system.

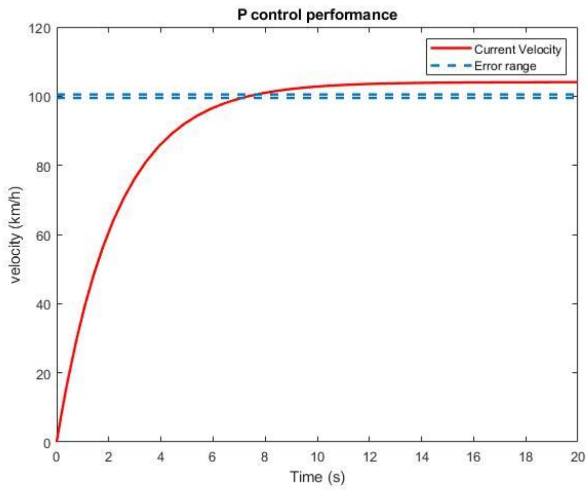

Figure 3: Proportional control performance

It is evident that the final value of a proportional controller is over the permitted error range 1%, which cannot meet practical specifications. It also means that there is a steady-state error using this controlling approach. The reason will be analyzed in the discussion part. In order to eliminate steady-state error and finally reach the desired speed, we are planning to introduce an integral control method in this project, which is effectively correcting the error.

B. *Proportional and integral control*

Basing on the above design, we are currently introducing integral control in this section. Integral control in a PID controller is the sum of the instantaneous error over time and gives the accumulated offset that should have been corrected previously. The integral term accelerates the movement of the movement of the process towards reference value and eliminates the residual steady-state error that occurs with a pure proportional controller. The PI control model is demonstrated in Figure 4.

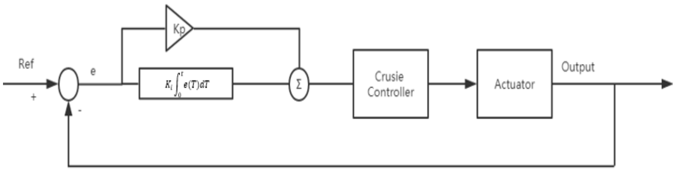

Figure 4: Proportional and integral dynamic control model

From Figure 4, we know that the accumulated error is multiplied by the integral gain Ki and then added with proportional value to input cruise controller. Hence, combining with the equation 6, we can now mathematically obtain the PI control model, which is expressed as:

$$2020\dot{x} = K_p\big(r(t) - x(t)\big) + K_i \int_0^t \big(r(T) - x(T)\big)dT - 72x \quad (9)$$

where *r(t)* is reference speed set by customers and *x(t)* is the velocity of vehicles as time goes. It is obvious that equation 9 is a linear system and after derivation, we can obtain K_p and K_i by specifying the reference speed and specifications shown in Table 1. In this section, reference speed is 100km/h. Through calculating, we can obtain the K_p and K_i value, which are assigned 1800 and 600 here, respectively. and then using MATLAB to simulate the performance of the system, as shown in Figure 5.

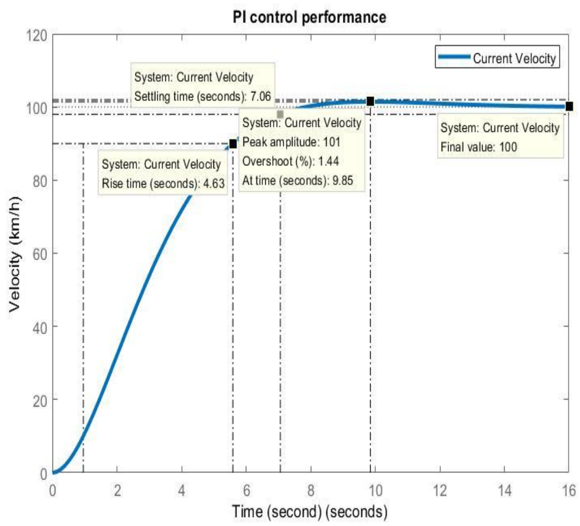

Figure 5: Proportional and integral control performance

It can be seen that this PI control is better than pure P control since it has fantastic performance on eliminating the steady-state error. Moreover, it also fulfills the specifications demand that we specify in Table 1, which not only can final value reach the desired speed 100km/h, but also can the overshoot and rise time meet the requirements defined in Table 1.

IV. Validation

This section is to explain and validate the result of the designed controller. It is quite significant to do validations to confirm that the controller should appropriately work under required conditions. Firstly, by changing the mass of the vehicle, probably equal to the total weight of several passengers plus gross vehicle mass, we examine the effect of uncertainty in mass. Moreover, the designed controller, PI controller, is examined under the dynamics of changing from one target speed to another, for example, 60,80,110, and 30 km/h. Finally, the constant disturbance caused by a sudden transition from flat ground to a very steep uphill slope of 5% grade is introduced into this controller.

A. Performance of the controller under uncertainty in mass

As we known, sometime there might be several passengers in a vehicle over the journey. Therefore, it is necessary to consider the effect of uncertainty in mass. In order to test this performance of controller, we are now assuming that the average weight of each passenger is 70 kg and the maximum number of seats of the car are 5. Based on these assumptions, we put five passengers into the vehicle, which definitely influence the gross vehicle mass (GVM), to assess the reaction of the controller. So, in this way, using MATLAB, we can visually show the outcome of the controller, as shown in Figure 6.

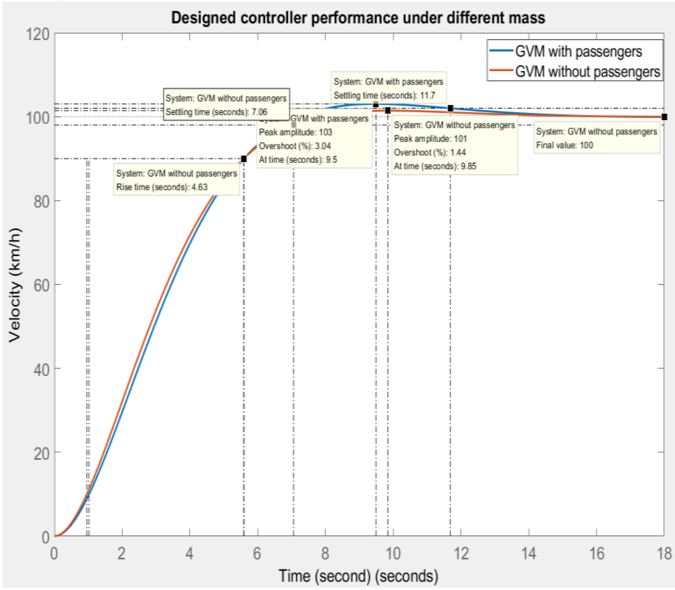

Figure 6: Designed controller performance under different mass conditions

From Figure 6, it is clear that the car finally can reach the desired speed, whether with five passengers, or not. However, there is some difference in settling time and overshoot. The reason why it happens will be stated in the following discussion part.

B. Performance of the controller under changing velocity conditions

To examine the performance under the different velocity, we are going to increase and decrease velocity randomly. For example, the reference velocity initially is set to 100km/h and then decreased to 80, 60, 40 km/h respectively, and finally surged to 120km/h. In this way, we can figure out the output of controller on changing velocity.

Therefore, in this section, utilizing the MATLAB software, by changing the setpoint, we can visually obtain the performance on changing velocity, as shown in Figure7.

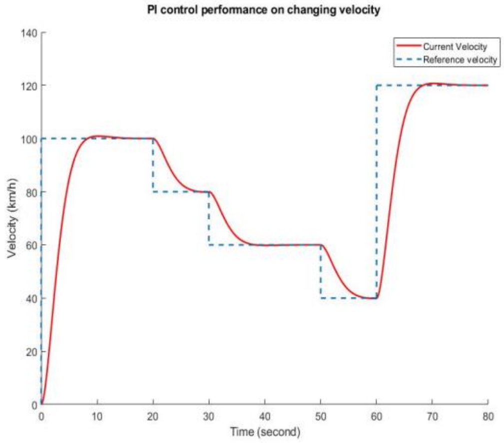

Figure 7: Proportional and integral control performance under changing velocity

Obviously, the car can precisely and smoothly reach the desired velocity whatever the reference speed decreases or increases. Through this examination, it is confirmed that the designed controller can work in any reasonable velocity.

C. Performance of the controller under constant disturbance conditions

In this part, the constant disturbance is introduced into the controller. Firstly, it is necessary to know that the disturbance for vehicle assumed in this paper is generated from a sudden transition from flat ground to a very certain steep uphill slope. For a robust controller, the critical thing is to work accurately even though there is some disturbance.

Hence, we assume that the car is driven in 100km/h on the flat ground but suddenly is driven on a steep uphill slope of 35
% grade, which will create a constant disturbance. According to Newton's second law of motion, we can obtain that the constant disturbance in this assumption is equal to $mgsin\theta$.

Figure 8 displays the free body diagram of the car on an uphill slope.

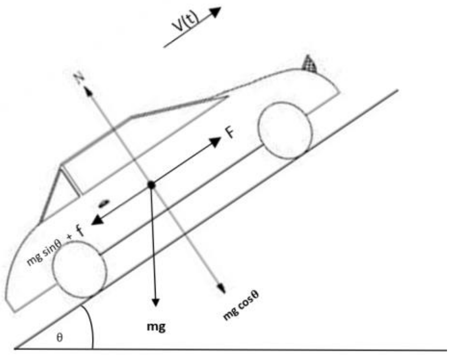

Figure 8: Free body diagram of the car

Because the car is suddenly driven to an uphill slope from flat ground, the previous balance of maintaining desired speed is broken. In order to keep the desired speed, the force generated by the engine of the car needs to be increased. Going backing to the original equation 3, it is time to introduce the disturbance in this system. Obtaining from the above analysis, we can obtain the equation:

$$m\dot{v} + \frac{1}{2}A\rho c_D v^2 = F - mgsin\theta \qquad (10)$$

where g is the gravity of earth, and other remaining variables are explained in the above discussion. Therefore, from a practical perspective, θ is limited and we select 5 degree in this paper.

Through calculating, when maintaining the speed 100km/h, the force 3780N is needed to be generated from the car engine. Nevertheless, the magnitude of disturbance, $mgsin\theta$, is 1725N approximately, which is nearly half of balance force in 100Km/h. Hence, it obvious that this disturbance highly influences the system and velocity, as shown in Figure 9.

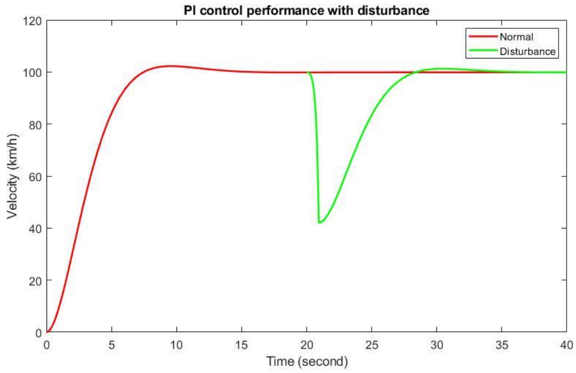

Figure 9: Proportional and integral control performance with disturbance

we can clearly see that the velocity of car sharply decreases because of the disturbance but finally the speed can reach the desired speed even though the driving force that generated by the engine is enormous suddenly and then gradually approaches to 5505N.

V. Discussion

Figure 3 shows a result of P control approach, which finally the speed cannot precisely reach the desired speed. It is also acknowledged that the controller works differently when the gross vehicle mass is changing, shown in Figure 6. Therefore, this section is mainly to explain and discuss the reasons why it happens.

A. Proportional control in terms of steady-state error

It is commonly known that P control is proportional control, which majorly depends on the proportional gain to correct the error between a reference variable and output variable. However, the final value in figure 3 cannot meet the desired value, which means that there is a steady-state error in this control method. The steady-state error is defined as the difference between the input and the output of a system in the limit as time goes to infinity.

From the equation 7, it is acknowledged that it is a first-order linear system with the input *r(t)* and output *x(t).* we are now assuming that the input is a step input as r(t) is a setpoint and constant *Kp* can be calculated. After that, the closed-loop system, therefore, has the step response:

$$x(t) = \left(x_0 - \frac{k_p r}{k_p + a}\right)e^{-(a+k_p)t} + \frac{k_p r}{k_p + a} \qquad (11)$$

Definitely we always select k_p greater than $-a$ so that the closed-loop system will be stable even the original system is unstable. Furthermore, it is evident that when the time goes infinity, which means $x(\infty) = k_p/(k_p + a)$,there is an equation $e(\infty) = ar/(k_p + a)$. Hence it is a so-called steady-state error.

To decrease this error, we can arbitrarily increase the Kp, proportional gain. However, we can not infinitely increase the proportional gain due to the practical limitation. For example, when increasing the Kp, the steady-state error decreases. However, the rise time and overshoot are over the limitation, as shown in Figure 9.

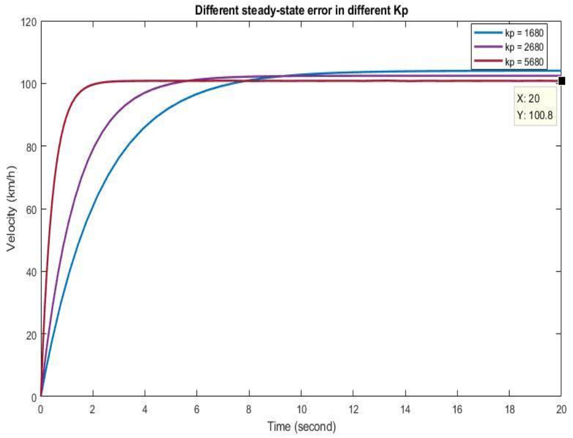

Figure 10: Proportional and integral control performance under changing velocity

Generally, demonstrated in figure 10, the rise time of the controller will increase as the magnitude of Kp increases, which is eventually out of the range of practical specifications of the vehicle although the steady-state error narrows in this case. Therefore, that is the reason why we move to design PI controller to eliminate the steady-state error, rather than arbitrarily increasing the proportional gain.

B. Different performance of the controller on varied gross vehicle mass

Connecting back to the figure 6, we know that when changing the gross vehicle mass by adding several passengers, the performance of controller will be different, especially in settling time and overshoot arguments. From the equation 1, we learn that the gross vehicle mass poses an impact on force generated by engines. Therefore, from a mathematical perspective, once the gross vehicle mass changes, the equation 8 is changed to:

$$(2020+m)\dot{x}(t) = K_p\big(r(t)-x(t)\big) + K_i\int_0^t \big(r(T)-x(T)\big)dT \text{ -}72x \quad (12)$$

where m is the magnitude of increased mass, which depends on how many passengers on the vehicle. According to the formula used to calculate the settling time and overshoot, after derivatizing and rearranging equation 12, the relationship between model parameters and pole locations can be obtained as follows:

$$\sigma = \frac{a}{2}, \qquad w_d = \frac{\sqrt{4b-a^2}}{2} \qquad (13)$$

where a and b are coefficients of $x\dot{(t)}$ and $x(t)$, respectively. Therefore, when the m, total weight of passengers, is increasing, the a is also responded to decrease, which decreases the σ. Similarly, w_d raises due to the decline of a. Moreover, there are further provided formulas demonstrating how the σ and w_d influence the settling time and overshoot, as shown below:

$$t_s = \frac{3.93}{\sigma}, \qquad M_p = e^{-\sigma\,\pi/w_d} \qquad (14)$$

Apparently, t_s rises once the σ declines. and M_p also decrease while σ and w_d decrease and increase, separately.

Therefore, the above analysis is the reason why the controller performs differently with responding to the uncertainty of mass.

VI. Conclusion

This paper mainly discusses the design of a cruise controller, containing linearization for the non-linear model, the introduction of PID control, the designing process of cruise controller. Basically, we start from P control method and analyze the performance. However, there is a steady-state error so that we move to design PI control, which is effective to eliminate the steady-state error. It also mathematically analyzes why we select proportional and integral control method, rather than pure proportional control. Moreover, examinations under several conditions are provided to test the performance of the designed controller, for example, changing target speed to another, suddenly changing from flat ground to steep uphill slope. From these examinations, we understand that how the mass influences the response of the controller and why we cannot select a high value of proportional gain.

In conclusion, this paper provides an example to design a cruise controller.

References

[1] *Bmw.co.za, 2018.* [Online]. Available: https://app.bmw.co.za/dws/VTS/servlet/VtsPdfUI?modelCode=KG72. [Accessed: 15- Sep- 2018].

[2] G. Shi, 2018. [Online]. Available: *https://wattlecourses.anu.edu.au/pluginfile.php/1816532/mod_resource/content/1/Notes_Wk4.pdf*. [Accessed: 16- Sep- 2018].

[3] K. Aström and R. Murray, *An Feedback systems : an introduction for scientists and engineers*. Princeton University Press, 2010.

[4] O. Garpinger *et al*, "Performance and robustness trade-offs in PID control," *Journal of Process Control*, vol. 24, *(5)*, pp. 568-577, 2014.