



prifysgol cymru trinity saint devid
University of Wales Trinity St David

Student name: Anandu Unnikrishanan Nair

Student ID number : 2411630

Programme : MSc – Automotive Engineering

Module : Energy Storage and Control Systems

Module code: ACHA7007

Assignment Title : Modelling Automotive Control Systems in MATLAB

Contents

1. ABSTRACT	3
2. INTRODUCTION.....	3
3. SIGNAL ENCODING AND QUANTISATION	3
4. CONTROLLER DESIGN AND CRITICAL PERFORMANCE EVALUATION.....	4
5. TRADE-OFF BETWEEN PERFORMANCE AND CONTROL EFFORT	5
6. EFFECT OF QUANTISATION (BIT RESOLUTION)	6
7. EFFECT OF SAMPLING TIME	6
8. EFFECT OF COMMUNICATION DELAY	6
9. CRITICAL ENGINEERING EVALUATION	7
10. REAL-WORLD AUTOMOTIVE APPLICATION	7
11. CONCLUSION	7
12. REFERENCES	8
13. APPENDIX	9

1. ABSTRACT

The modelling and analysis of a digital cruise control system using a proportional-integral-derivative (PID) controller developed in MATLAB are presented in this paper. Maintaining vehicle speed at a particular setpoint while evaluating the impact of digital restrictions such as quantisation, sampling time, and communication delay is the goal. With a rising time of roughly 4–5 seconds, overshoot of 10–15%, and settling time of 8–10 seconds, the controller achieved zero steady-state error, suggesting a trade-off between responsiveness and control smoothness. Quantisation error was introduced by reduced bit resolution, and phase lag caused stability to deteriorate with longer sampling times and delays. Iterative tuning was used to choose controller gains while balancing control effort, speed, and stability. Large control spikes may result in dangerous acceleration or braking, according to ISO 26262 principles. When compared to PID control, advanced control strategies such as Model Predictive Control exhibit enhanced resilience.

2. INTRODUCTION

Cruise control systems, which allow cars to maintain a desired speed without constant human input, are essential parts of contemporary automobile engineering. Consistent vehicle performance, decreased driver fatigue, and increased fuel efficiency are all benefits of these systems. The proportional-integral-derivative (PID) controller is still one of the most popular control techniques because of its ease of use, resilience, and flexibility in a variety of dynamic systems.

PID controllers work by continuously reducing the discrepancy between the actual system output and a desired setpoint. To achieve both transient and steady-state performance, they combine derivative, integral, and proportional operations. Real-world automobile systems, on the other hand, are digitally implemented and subject to limitations like quantisation, sampling, and communication delays; they are not entirely continuous.

The modelling and critical analysis of a MATLAB-implemented digital cruise control system is presented in this research. This work focuses on understanding system behaviour, analyzing performance trade-offs, and assessing implications in the context of actual automotive standards, specifically ISO 26262, which regulates functional safety in road cars, in contrast to basic descriptive analysis.

3. SIGNAL ENCODING AND QUANTISATION

This system simulates a true vehicle communication network by transmitting all signals as binary values, including velocity, demand, and control. The driver demand fluctuates between 30 and 110 mph, while the velocity sensor runs between 0 and 150 mph. A predetermined amount of bits are used to encode these signals into binary, which are then decoded back into physical values for control.

This method adds quantisation effects even though it mimics embedded systems found in the actual world. Because discrete levels are used to mimic continuous signals, quantisation takes

place. The number of bits used directly affects this approximation's resolution. Quantisation is a source of noise and nonlinearity from the standpoint of control. Reduced bit resolution in the simulation led to oscillatory behaviour and decreased tracking precision. This is not only a numerical restriction; in a real car, these oscillations could result in uncomfortable acceleration patterns and wasteful energy use.

ISO 26262 highlights the significance of signal integrity and accuracy from a safety perspective. Poor signal resolution could lead to incorrect control decisions, particularly in safety-critical systems. As a result, choosing the right number of bits is necessary for both performance and safety.

4. CONTROLLER DESIGN AND CRITICAL PERFORMANCE EVALUATION

In order to adjust vehicle velocity depending on the discrepancy between demand and actual velocity, a PID controller was put into place. The chosen gains were:

$$K_p = 2$$

$$K_i = 0.5$$

$$K_d = 0.1$$

Iterative tuning was utilised to choose these gains: K_d enhanced damping, K_i eliminated steady-state error but generated oscillations, and increasing K_p increased speed but resulted in overshoot.

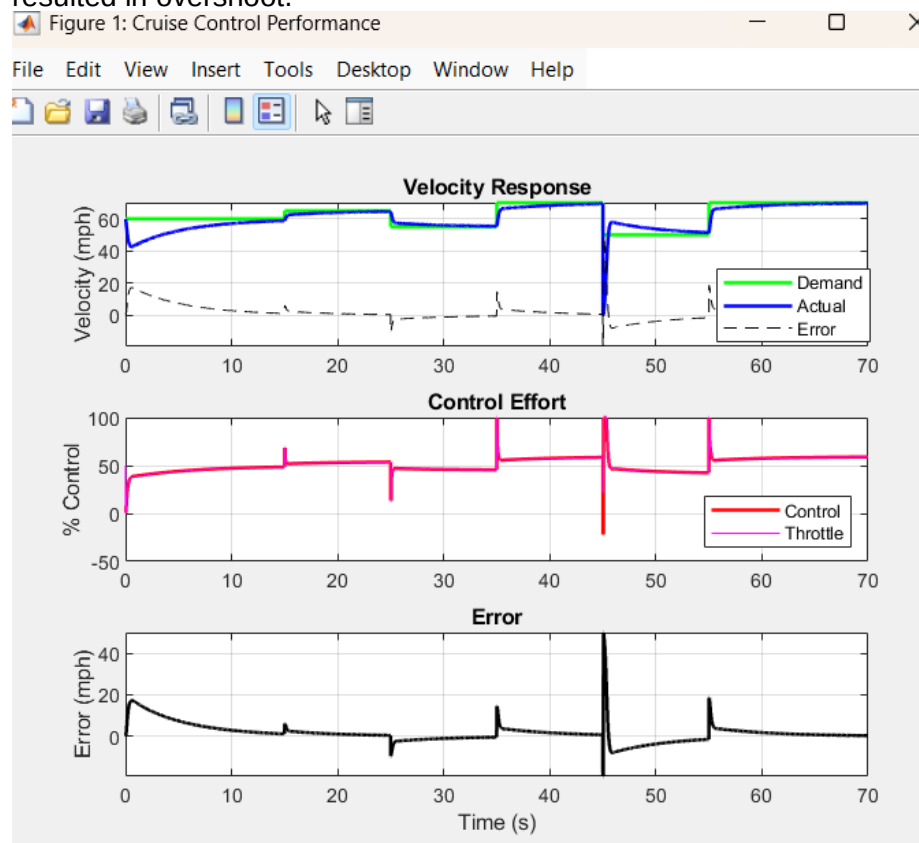


Figure 1: System Response

The system tracks the demand with very little steady-state error, but transient analysis reveals a settling period of about 8–10 seconds and an overshoot of about 10–15%. The rising time, which is roughly 4–5 seconds, the 10–15% overshoot is larger than is usually permitted in automotive systems, where smoother reactions are required to assure passenger comfort and safety. Nevertheless, this indicates a rather quick response.

The actual velocity closely follows the demand signal, indicating that the controller achieves good tracking performance. The integral term's role is confirmed by the effective elimination of the steady-state error. Nevertheless, a more thorough examination uncovers restrictions that are not immediately obvious from straightforward observation. Abrupt shifts in demand, especially around 45 seconds, cause significant spikes in control effort and velocity deviation. These spikes show that the controller is responding violently to sudden disruptions.

This behaviour reveals a basic drawback of PID control: although it functions well in calm situations, its reactive nature makes it difficult to handle abrupt changes. The controller reacts to current errors rather than anticipating future states, which can cause overshoot and instability in situations that change quickly. This causes issues with actuator wear, passenger comfort, and system robustness from a technical standpoint. Even if a controller achieves accurate tracking, it might not be suitable in actual automotive applications if it generates significant control spikes.

5. TRADE-OFF BETWEEN PERFORMANCE AND CONTROL EFFORT

The identification of a trade-off between control effort and velocity tracking performance is a crucial result of this investigation. Although it increases response time, increasing the proportionate gain causes aggressive control actions. In a similar vein, the integral term removes steady-state error but may cause oscillations when the system undergoes sudden changes. By providing damping, the derivative term helps lessen these effects, but too much derivative activity might make the system too sensitive to noise. In car design, this trade-off is crucial. Although quick response is ideal, system stability and passenger comfort must not be sacrificed in the process. The observed control spikes could be deemed dangerous in a real vehicle since, according to ISO 26262, such systems may fall under ASIL B–C, where unintentional acceleration or braking must be severely prevented. Safety may be jeopardised by excessive oscillations or sudden control actions, especially in systems that interact with other vehicle subsystems.

As a result, controller tuning needs to be viewed as a multi-objective problem that balances control smoothness, stability, and responsiveness.

6. EFFECT OF QUANTISATION (BIT RESOLUTION)

By modifying the amount of bits used for signal encoding, the effect of bit resolution on system performance was examined. System behaviour clearly deteriorated at lower resolutions, such as 8-bit, with more oscillations and decreased precision.

The increased quantisation error, which adds uncertainty to the control loop, can account for this. These errors cause unneeded corrections and instability because the controller responds to them as though they were actual disturbances.

The technology gets smoother and more precise as the bit resolution rises. A 16-bit representation offers enough resolution to reduce these impacts. Crucially, it was discovered that roughly 12 bits was the lowest acceptable resolution. System performance drastically declines below this point. Practically speaking, this emphasises how crucial it is to choose the right signal resolution for embedded systems. Higher resolution boosts performance, but it also raises processing and communication bandwidth needs. Thus, it is necessary to reach the ideal balance.

7. EFFECT OF SAMPLING TIME

The controller's output update frequency is determined by sampling time. While responsiveness and accuracy are enhanced by a shorter sample period, computational load is increased. On the other hand, longer sample times result in delays that impair system performance.

When the sample period in this system was increased above 0.1 seconds, oscillations and a delayed response were observed. This happens as a result of the controller essentially using out-of-date data, which limits its capacity to react to changes in real time. From the standpoint of control theory, this decreases system stability by introducing phase lag. Sampling time in actual automobile systems is limited by communication protocols and hardware constraints. Because delayed control operations might jeopardise system safety, ISO 26262 underlines the significance of temporal limitations. As a result, choosing the right sample period is crucial and necessitates striking a compromise between computational viability and performance.

8. EFFECT OF COMMUNICATION DELAY

The time needed for signals to travel between system components is known as communication delay. In this model, oscillations and, in severe situations, instability increased with increasing delay.

The introduction of phase lag, which lessens system damping, can account for this phenomenon. Inaccurate or excessive adjustments result from the controller acting on out-of-date information. Larger delays cannot be adequately handled by PID control alone, however small delays can be compensated for by modifying controller gains. The limitations of

traditional control mechanisms in contemporary networked systems are highlighted by this. From a safety standpoint, ISO 26262 mandates that control systems take communication delays into consideration in order to guarantee dependable and consistent behaviour. If this isn't done, dangerous operating circumstances may arise.

9. CRITICAL ENGINEERING EVALUATION

The results of this study show that although PID control works well for cruise control applications, digital implementation limitations significantly affect its performance. The non-idealities introduced by quantisation, sampling, and delay have a substantial impact on the behaviour of the system. Because they can cause instability, unnecessary control effort, and decreased system reliability, these variables need to be taken into account throughout the design phase.

Furthermore, the findings imply that highly dynamic or networked systems might require more than just PID management. In such circumstances, sophisticated control techniques like adaptive or model predictive control may offer better performance. PID control works well for straightforward systems, but it struggles when there are delays and sudden disruptions. On the other hand, Model Predictive Control (MPC) produces smoother and more stable performance by forecasting future system behaviour and imposing limitations on control inputs. Because of this, MPC is more suited for contemporary automobile applications where comfort and safety are crucial.

10. REAL-WORLD AUTOMOTIVE APPLICATION

The simulation's behaviour reveals constraints that are handled in actual automobile systems. For instance, BMW Adaptive Cruise Control avoids the model's noticeable control spikes by emphasising smooth acceleration and deceleration to improve passenger comfort. More filtering techniques and precise tuning are used to accomplish this.

Similarly, rather of depending only on PID control, Tesla's Autopilot system employs model-based and predictive control techniques. As a result, the system performs more smoothly and steadily since it can anticipate changes rather than responding to mistakes.

In the simulation, real cars use high-speed networks like automotive Ethernet and CAN to reduce communication delays. Aggressive control behaviour and instability are unacceptable since such systems must function securely in all circumstances, according to ISO 26262. This shows that, in contrast to the simulated PID behaviour seen in this work, real automotive systems place a higher priority on smoothness and safety than aggressive response.

11. CONCLUSION

A digital PID-based cruise control system was thoroughly modelled and critically analysed in this research. The findings show that even while the controller tracks velocity accurately, a number of important parameters, such as communication delay, sampling time, and signal resolution, affect its performance. The investigation revealed difficulties related to digital

implementation as well as significant trade-offs between performance and control effort. These results highlight how important it is to carefully design systems and follow technical standards.

In the end, attaining optimal performance necessitates striking a balance between realistic restrictions, stability, and responsiveness to guarantee that the system functions effectively and safely under real-world circumstances. Overall, the system prioritises quick response over smooth control, which suggests that the tuning is not ideal for practical automobile applications.

Among the analysed factors, communication delay was found to have the most critical impact on system stability, as it introduces significant phase lag and can lead to instability even with well-tuned controller gains. In comparison, quantisation mainly affects precision, while increased sampling time reduces responsiveness but remains manageable within certain limits. Furthermore, the combined effect of low bit resolution and large sampling intervals can significantly amplify oscillations, highlighting the importance of considering these constraints collectively rather than in isolation. Therefore, achieving optimal control performance in automotive systems requires an integrated design approach that balances all digital implementation limitations alongside safety requirements defined by ISO 26262

12. REFERENCES

- Ogata, K., 2010. *Modern Control Engineering*. 5th ed. Upper Saddle River: Prentice Hall.
- Nise, N.S., 2020. *Control Systems Engineering*. 8th ed. Hoboken: Wiley.
- Åström, K.J. and Murray, R.M., 2008. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton: Princeton University Press.
- International Organization for Standardization (ISO), 2018. *ISO 26262: Road Vehicles – Functional Safety*. Geneva: ISO.
- International Organization for Standardization (ISO), 2015. *ISO 11898: Road Vehicles – Controller Area Network (CAN)*. Geneva: ISO

13. APPENDIX

MATLAB CODE EXPLANATION

```
clc
clear
close all

%% (1) Controller gains
P = 2;
I = 0.5;
D = 0.1;

integral = 0;
prev_error = 0;

%% (2) Implementation parameters
Ts = 0.05;

m_velocityS = 0;
M_velocityS = 150;

m_demand = 30;
M_demand = 110;

n_bits_sensor_signal = 16;
n_bits_demand_signal = 16;
n_bits_actuator_signal = 16;

R = 2^n_bits_sensor_signal;

%% (3) Simulation parameters
sim_step = 0.01;
sim_time = 70;
T = 0:sim_step:sim_time;

[d_vec,u_vec,y_vec,e_vec,Th_vec] =
deal(nan(size(T)));

throttle_setting = 50;
```

```

actuator_signal_delay = 0.00;

u = 50;
velocity_real = 60;

%% (4) Simulation loop
for j = 1:length(T)

    % Get raw velocity
    [velocity_raw, velocity_real, throttle_setting]
    = ...
        getRawVelocity(n_bits_sensor_signal,
            velocity_real, throttle_setting, sim_step, u);

    % Decode velocity
    velocity = ((bin2dec(velocity_raw) *
        (M_velocityS - m_velocityS)) / R) +
        m_velocityS;

    if mod((j-1)*sim_step, Ts) == 0

        % Get demand signal
        demand_raw =
            getRawVelocityDemand(n_bits_demand_signal,
                sim_step, j);

        % Decode demand
        demand = ((bin2dec(demand_raw) *
            (M_demand - m_demand)) / R) + m_demand;

        % ===== PID CONTROLLER =====
        error = demand - velocity;

        integral = integral + error * Ts;
        derivative = (error - prev_error) / Ts;

        control = P*error + I*integral +
            D*derivative;

        % Saturation

```

```

        control = max(min(control, 100), -100);

        prev_error = error;

        % ===== ENCODE CONTROL =====
        control_scaled = ((control + 100) *
            (2^n_bits_actuator_signal)) / 200;
        control_raw =
            dec2bin(round(control_scaled));

        % Send to actuator
        u = sendControlCommand(control_raw,
            n_bits_actuator_signal, ...
                actuator_signal_delay, j, sim_step,
                u_vec);

    end

    % Store values
    e_vec(j) = error;
    d_vec(j) = demand;
    u_vec(j) = control;
    y_vec(j) = velocity_real;
    Th_vec(j) = throttle_setting;

end

%% (5) Plot results

figure('Name','Cruise Control Performance')

subplot(3,1,1)
plot(T, d_vec, 'g', 'LineWidth', 1.5); hold on
plot(T, y_vec, 'b', 'LineWidth', 1.5)
plot(T, e_vec, 'k--')
grid on
title('Velocity Response')
ylabel('Velocity (mph)')
legend('Demand','Actual','Error')

```

```

subplot(3,1,2)
stairs(T, u_vec, 'r', 'LineWidth', 1.5); hold on
plot(T, Th_vec, 'm')
grid on
title('Control Effort')
ylabel('% Control')
legend('Control','Throttle')

subplot(3,1,3)
plot(T, e_vec, 'k', 'LineWidth', 1.5)
grid on
title('Error')
xlabel('Time (s)')
ylabel('Error (mph)')

```

A PID controller is used in the MATLAB code to mimic a cruise control system. The goal is to maintain the car's speed at the driver's preferred level. The controller gains (K_p , K_i , and K_d), which regulate the system's reaction to mistakes, are first defined. Next, system settings are specified, including signal ranges and sample times.

The vehicle speed is measured and transformed from binary to a real value within the simulation loop. Additionally, the desired speed is acquired and decoded. The controller generates a control signal by applying the PID algorithm after calculating the error between the desired and actual speed.

Before being transmitted to the actuator, this control signal is encoded back into binary and restricted between -100 and 100. The actuator modifies the brake or throttle appropriately. In order to display velocity, control effort, and error over time, the data are finally shown.