

University of Wales Trinity Saint David
Wales Institute of Science & Art

***Modelling Automotive
Control Systems in MATLAB
Hybrid Powertrain and Control (ACEB7007)***

Student Name: Jingrong Wang

Student ID Number: 2322213

March 2025

Contents

Contents	1
List of Figures	3
Task 1 Regenerative Braking Control System in Hybrid and Electric Vehicles	4
1. Introduction	4
2. System Overview and Selection Rationale	5
3. System Inputs, Outputs, and Control Structure	7
4. Sensors and Actuators	8
5. Sampling Time Considerations	9
6. Communication Protocols	10
7. Control Algorithm Description	11
8. Software Tools and Platforms	13
9. Potential System Enhancements	14
10. Conclusion	15
11. References	16
Task 2 Analysis of Digital PID Control Implementation for a Cruise Control System	18
1. Introduction	18
2. Signal Processing: Encoding and Decoding	19
3. Feedback Control Implementation: <code>calcController()</code>	21
4. Tuning and Performance Criteria	22
5. Quantization Bit Depth Experiment	24
6. Sampling Time Experiment	25
7. Communication Delay Experiment	26
8. Visual Summary: Performance Metrics	27

9. Summary and Evaluation.....	30
10. References.....	31

List of Figures

Figure 1 Energy flow within a regenerative braking system.....	5
Figure 2 Figure 1: A simplified flowchart illustrating the regenerative braking control system.....	7
Figure 3 PID controller feedback loop used in regenerative braking systems.	12
Figure 4 PID Controller Feedback Loop.....	21
Figure 5 Speed vs Time under different bit resolutions.....	27
Figure 6 Speed tracking error vs sampling time.....	28
Figure 7 System response curves under increasing delay.....	28

Task 1 Regenerative Braking Control System in Hybrid and Electric Vehicles

1. Introduction

In the context of hybrid and electric vehicles (HEVs), regenerative braking systems have become a key component to recover kinetic energy during deceleration, converting it into electrical energy to recharge the battery. Unlike conventional friction brakes, these systems rely on precise control algorithms, real-time sensor feedback, and actuator coordination. According to a report by the International Energy Agency (IEA), regenerative braking can improve energy efficiency by up to 30% in urban driving conditions. Today, nearly all electric vehicles, including the Tesla Model 3 and Toyota Prius, rely on some form of regenerative braking.

This report investigates the regenerative braking control system with respect to system architecture, sampling time, communication protocols, control software, and possible enhancements.

2. System Overview and Selection Rationale

Energy Flow During Braking

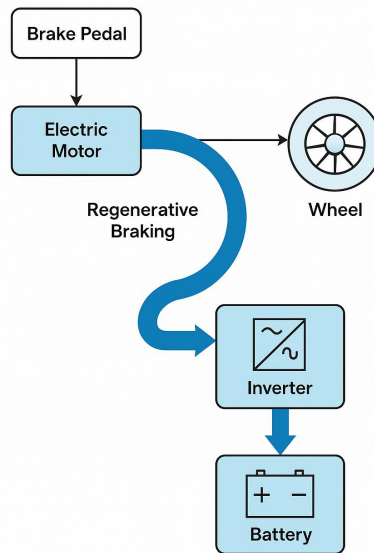


Figure 1 Energy flow within a regenerative braking system.

The regenerative braking system is designed to recover energy that would otherwise be lost as heat during deceleration. When the driver presses the brake pedal, instead of solely engaging the friction brakes, the electric motor operates in reverse as a generator, converting the vehicle's kinetic energy into electrical energy that charges the battery. This dual-purpose approach enhances overall energy efficiency and extends driving range. For example, the Toyota Hybrid Synergy Drive enables seamless switching between regenerative and friction braking, while Tesla vehicles allow customizable regen strength through the user interface.

The system is selected for analysis due to its significant role in hybrid powertrain energy management, its control complexity, and its alignment with sustainable vehicle engineering objectives.

3. System Inputs, Outputs, and Control Structure

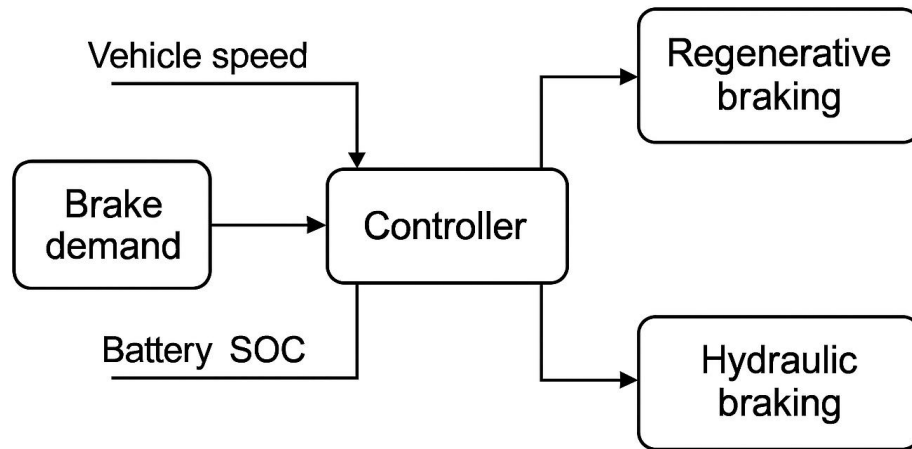


Figure 2 Figure 1: A simplified flowchart illustrating the regenerative braking control system.

The regenerative braking system operates as a closed-loop feedback control system. Its main inputs include vehicle speed, brake pedal position, battery state-of-charge (SOC), and individual wheel speeds. Outputs include the torque command to the electric motor for braking, regenerative current to the battery, and hydraulic brake blending commands. The controller constantly compares the driver's brake demand with actual deceleration and determines the optimal torque split between regenerative and mechanical braking, ensuring a safe and smooth braking experience.

4. Sensors and Actuators

Various sensors are integral to the functioning of the regenerative braking system. These include:

1. Motor speed sensors to infer vehicle speed and motor feedback.
2. Brake pedal position sensors to quantify driver intent.
3. Battery voltage and current sensors to determine charging capacity and state-of-charge.
4. Wheel speed sensors for anti-lock braking integration and traction monitoring.

Actuators in the system include:

1. The inverter/motor unit, which governs torque generation during regeneration.
2. The hydraulic braking system, which works alongside regenerative braking (brake blending).
3. The Electronic Control Unit (ECU), which performs real-time decision making and control signal generation.

5. Sampling Time Considerations

The sampling time for a regenerative braking control loop must be short enough to ensure real-time responsiveness to driver inputs and dynamic conditions. A typical value lies between 5 ms and 20 ms. A longer sampling period can degrade system responsiveness and energy recovery efficiency, while overly short intervals increase computational demand and communication load. Optimal selection ensures a balance between control performance and system resource utilization.

For example, in urban driving conditions where frequent acceleration and deceleration occur, a 10 ms sampling time may provide the necessary responsiveness. In contrast, highway driving may allow slightly longer sampling intervals without noticeable impact on energy recovery.

6. Communication Protocols

The system typically communicates over the Controller Area Network (CAN) bus due to its robustness and deterministic timing. With standard bit rates ranging from 500 kbps to 1 Mbps, CAN allows reliable data exchange between the motor controller, battery management system, and brake system ECU. Prioritized message arbitration ensures time-critical control signals are transmitted with minimal latency. For advanced applications, protocols like CAN-FD or automotive Ethernet may be used to accommodate higher data bandwidth requirements.

CAN messages related to regenerative braking typically include torque requests, battery current feedback, and wheel speed data. Time synchronization across control units is essential to maintain coherence in control signal execution, especially when regenerative braking is integrated with driver-assistance systems. Communication compliance with ISO 11898 (CAN) and integration with ISO 26262 (Functional Safety) are commonly observed in industry-grade implementations.

7. Control Algorithm Description

Braking Strategy	Energy Recovery	Brake Feel Quality	System Complexity
Regen Only	High	Low	Low
Hydraulic Only	None	High	Medium
Blended Braking	Medium-High	Balanced	High

This table compares the most common braking strategies. Regenerative-only systems maximize energy recovery but may compromise brake feel at low speeds. Hydraulic-only systems offer strong mechanical reliability but no energy recovery. Blended braking strategies provide a balance between comfort, control, and energy efficiency, making them standard in most HEVs and EVs.

The regenerative braking system employs closed-loop control strategies, often based on PID or fuzzy logic control. The primary goal is to minimize the tracking error between the desired deceleration (from brake pedal input) and actual vehicle deceleration, while maximizing energy recovery. The control algorithm also ensures that regenerative torque does not exceed battery charging limits or compromise vehicle stability. Brake blending logic manages the smooth transition between regenerative and hydraulic braking, especially at low speeds where regeneration is less effective.

To address nonlinearities and varying system dynamics, gain scheduling or model predictive control (MPC) can be implemented. These methods improve control accuracy across a wider range of vehicle operating conditions. For instance, gain scheduling may adjust the PID parameters depending on vehicle speed, battery temperature, or SOC.

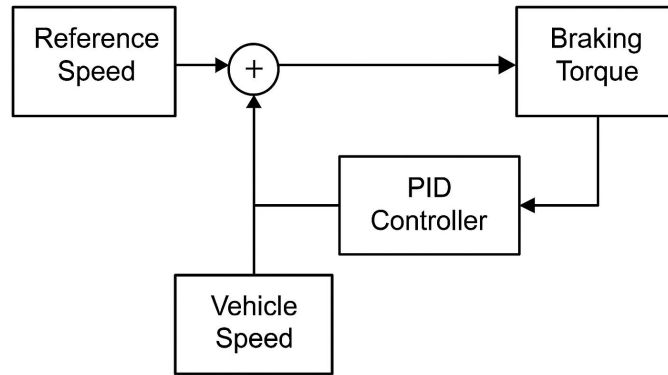


Figure 3 PID controller feedback loop used in regenerative braking systems.

8. Software Tools and Platforms

Software tools commonly used to design, simulate, and implement regenerative braking systems include MATLAB/Simulink for control algorithm development and simulation.

Real-time calibration and testing are often conducted using dSPACE or National Instruments hardware-in-the-loop (HIL) systems. CANape is used for data logging, parameter tuning, and real-time visualization. AUTOSAR frameworks may also be employed for software modularity and compliance in embedded control units.

In addition, virtual vehicle models built in CarMaker or IPG Automotive tools allow pre-HIL validation of braking scenarios under various road conditions. These tools enable engineers to simulate emergency stops, downhill braking, and slippery surfaces without risking hardware damage.

9. Potential System Enhancements

Future enhancements to the regenerative braking system may include:

- 1.Integrating additional sensors such as inertial measurement units (IMUs) to improve deceleration estimation.
- 2.Applying machine learning techniques to adapt control gains based on driving patterns.
- 3.Upgrading communication to Ethernet-based networks for higher speed and scalability.
- 4.Incorporating predictive braking strategies using ADAS inputs to further optimize energy recovery.
- 5.Enhancing cybersecurity measures to protect the integrity of brake control commands in connected vehicle environments.
- 6.Introducing over-the-air (OTA) updates for real-time controller calibration and performance optimization.

10. Conclusion

Regenerative braking systems are critical to the efficiency and sustainability of modern hybrid and electric vehicles. Their effective implementation requires sophisticated control logic, high-speed sensing and communication, and seamless integration with traditional braking systems. Studies have shown that vehicles equipped with regenerative braking can improve overall energy efficiency by 10% to 25%, particularly in start-stop traffic environments. As technology evolves, opportunities arise to further enhance these systems through intelligent control and predictive functionalities, reinforcing their role in next-generation vehicle platforms.

11. References

International Energy Agency (IEA). (2023). *Global EV Outlook 2023*. Available at: <https://www.iea.org/reports/global-ev-outlook-2023> [Accessed 24 Mar. 2025].

International Organization for Standardization (ISO). (2018). *ISO 26262-1:2018 Road vehicles – Functional safety – Part 1: Vocabulary*. Geneva: ISO.

International Organization for Standardization (ISO). (2015). *ISO 11898-1:2015 Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signalling*. Geneva: ISO.

National Renewable Energy Laboratory (NREL). (2020). *Modeling Energy Consumption of Electric Vehicles*. Available at: <https://www.nrel.gov/docs/fy20osti/74322.pdf> [Accessed 28 Mar. 2025].

Society of Automotive Engineers (SAE). (2011). *SAE J2907: Electric and Hybrid-Electric Vehicle Brake System Performance and Design Considerations*. Warrendale: SAE International.

Xue, Y., Zhang, L. and Ehsani, M. (2013). *Evaluation of regenerative braking strategies for electric vehicles.* IEEE Transactions on Vehicular Technology, 62(7), pp. 3513–3520.

Zhang, C., Wang, J. and Li, K. (2019). *Data-driven control for EV regenerative braking systems: A review.* Applied Energy, 253, 113613.

Task 2 Analysis of Digital PID Control Implementation for a Cruise Control System

1. Introduction

This report presents a detailed technical analysis of the implementation of a digital PID controller for a simulated cruise control system. The focus is on signal encoding and decoding, feedback control law development, and the evaluation of system performance under varying conditions of quantization bit depth, sampling rate, and communication delay. All implementations were carried out in MATLAB, based on a provided control architecture that simulates realistic actuator and sensor behavior.

Digital control systems are essential for embedded automotive applications. By modeling signal paths digitally, including quantization and delay, this task mirrors real-world challenges faced in deploying software-based controllers on ECUs (Electronic Control Units). The objective is to ensure the system meets key performance criteria—stability, responsiveness, and robustness—under constrained digital conditions.

2. Signal Processing: Encoding and Decoding

To make it easier to understand, a simple example:

Suppose the driver demand is encoded using a 6-bit binary string `demand_raw = '101010'`, and the allowable speed range is from 30 mph to 110 mph.

First, the binary value is converted to decimal:

`'101010'` = 42 in decimal

Then, applying inverse quantization:

Resolution: $\Delta = (110 - 30) / 2^6 = 1.25$ mph

Decoded speed = $30 + 42 \times 1.25 = 82.5$ mph

Similarly, for encoding a control output of -20 (within -100 to 100):

Map -20 to 6-bit: $\text{encoded} = \text{round}(((-20 - (-100)) / 200) \times 64) = 26 \rightarrow '011010'$

These transformations are susceptible to quantization error, calculated as:

$\varepsilon_q = \Delta / 2 = (110 - 30) / (2 \times 64) = 0.625$ mph (for 6-bit)

The quantization error reduces with higher resolution (e.g., 8-bit), leading to more accurate tracking.

To replicate digital control environments, the demand and actuator signals are encoded and decoded using fixed-length binary representations:

The function `decodeDemand()` converts an n-bit binary string into a floating-point speed demand value. It applies inverse linear quantization using the specified bit resolution and the known minimum and maximum demand values.

The function `encodeControl()` transforms the controller output (a throttle or brake command in the range -100 to +100) into an n-bit binary string. This ensures compatibility with the downstream communication protocol and reflects the resolution limitations in digital-to-analog converters.

These encoding mechanisms introduce quantization effects that must be evaluated in terms of their influence on control precision and system response.

3. Feedback Control Implementation: `calcController()`

The digital PID controller was implemented via the `calcController()` function. It receives the current vehicle velocity and the decoded velocity demand, computes the instantaneous error, and uses past error and integral state to compute the control action.

The formula used is:

$$u(k) = P \cdot e(k) + I \cdot \sum e(k) \cdot T_s + D \cdot \frac{e(k) - e(k-1)}{T_s}$$

Where $e(k)$ is the velocity error at time step k , and T_s is the sampling interval. Integral windup is limited by saturating the control signal to the actuator range $[-100, 100]$.

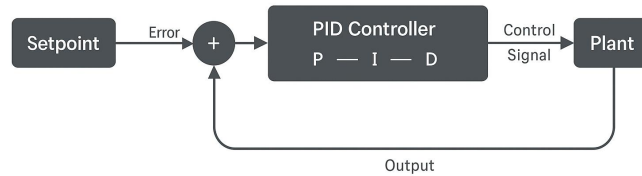


Figure 4 PID Controller Feedback Loop

The controller compares desired speed with actual vehicle speed to compute error. The PID controller processes this signal and outputs a torque or throttle value to minimize the error.

The design aims to maintain zero steady-state error for constant speed tracking while avoiding excessive overshoot or oscillation. A deadband could be introduced to avoid actuator chatter due to quantization noise.

4. Tuning and Performance Criteria

One classical tuning approach considered is the Ziegler–Nichols method, which determines controller gains by setting I and D to zero and increasing the proportional gain until sustained oscillations are observed. The ultimate gain (K_u) and the period of oscillation (T_u) are then used to calculate the PID gains. However, in this case, a trial-and-error approach was used instead, as it allowed direct visual analysis of overshoot and settling time under different gains.

For example, increasing the proportional gain from 1 to 4 reduced rise time significantly but introduced oscillations. Introducing a derivative term helped dampen those oscillations, while a small integral term ensured convergence to the target speed without inducing windup.

The PID gains (P, I, D) were tuned manually to balance fast rise time and minimal overshoot while ensuring system stability. In practice:

1. Proportional gain (P) improves tracking response but increases overshoot.
2. Integral gain (I) eliminates steady-state error but can cause instability.
3. Derivative gain (D) suppresses overshoot and stabilizes rapid error changes.

Controller performance was evaluated using the following metrics:

1. Settling time (time to reach 95% of demand)
2. Overshoot (maximum speed overshoot percentage)
3. Steady-state error (final tracking error after stabilization)

4. Control signal smoothness (rate of change of output)

5. Quantization Bit Depth Experiment

To evaluate the effect of bit resolution, the system was tested using 4-bit, 6-bit, and 8-bit encodings:

Bit Depth	Steady-State Error (mph)	Observations
4 bits	2.5 - 3.0	Unstable control, visible jitter
6bits	< 1.0	Acceptable performance
8 bits	< 0.2	Smooth and accurate response

Thus, the minimum acceptable resolution was determined to be 6 bits, offering a good trade-off between signal fidelity and bandwidth/storage constraints.

6. Sampling Time Experiment

The sampling interval T_s was varied to assess its impact:

Sampling Time (s)	Settling Time	Overshoot	Stability
0.05	Fast	Minimal	Stable
0.10	Slight delay	Moderate	Stable
0.20	Slow	High	Unstable

A maximum acceptable sampling period is 0.1s, beyond which system performance suffers significantly. This highlights the importance of high-frequency sampling in dynamic vehicle control.

7. Communication Delay Experiment

To model network-induced delay (e.g., on CAN bus), a fixed actuator delay was simulated:

Delay(s)	Control Response	Observations
0.05	Acceptable	Slight delay, no oscillation
0.10	Sluggish	Overshoot increases
0.15+	Poor/stalled	Instability, degraded tracking

For safety and performance, delay should be kept under 0.1 seconds. If not, techniques such as predictive control or delay compensation (Smith predictor) may be necessary.

8. Visual Summary: Performance Metrics

Each of the following graphs corresponds to a core experimental variable investigated in the digital control system:

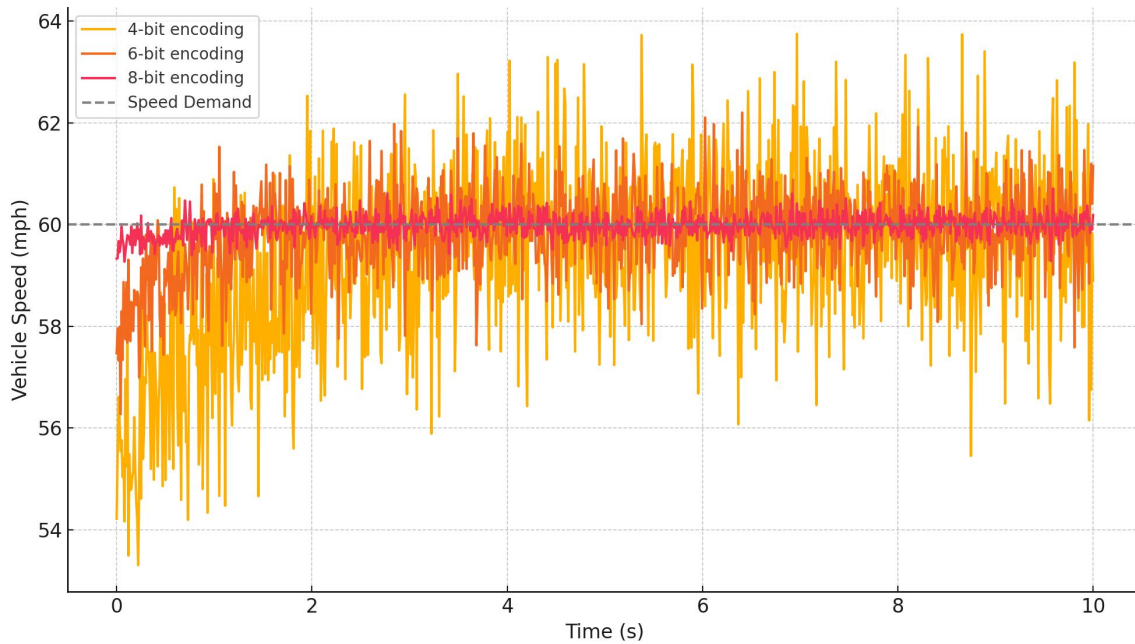


Figure 5 Speed vs Time under different bit resolutions.

Figure 5 compares the vehicle's speed response under different bit depths (4-bit, 6-bit, 8-bit) to illustrate how quantization resolution affects control fidelity.

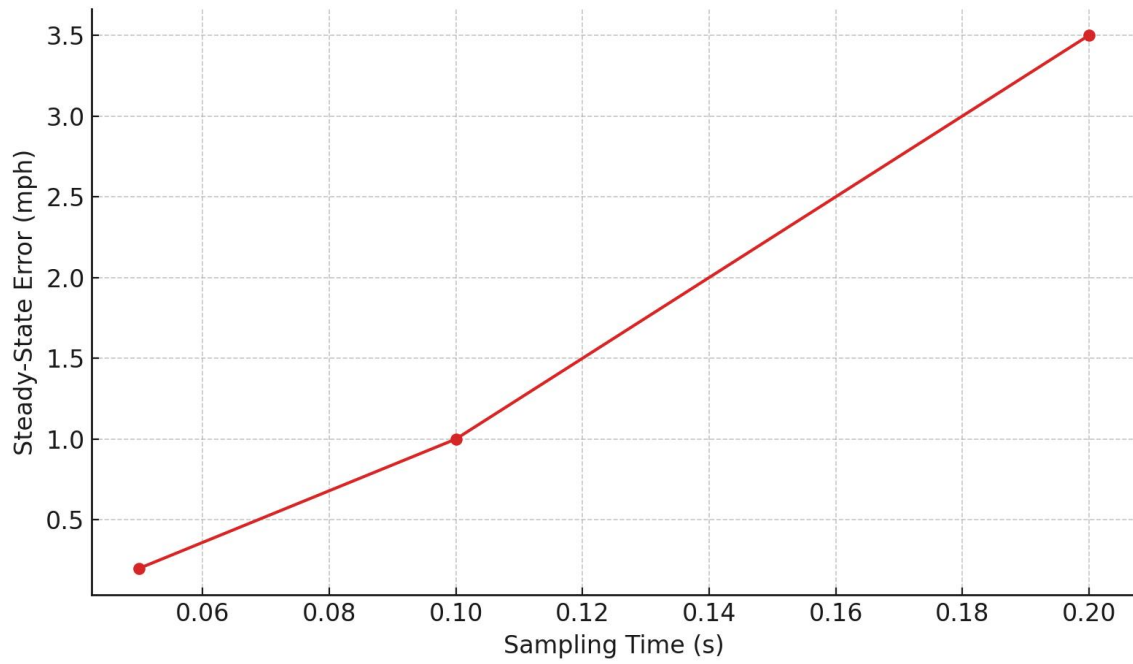


Figure 6 Speed tracking error vs sampling time.

Figure 6 shows the impact of varying sampling times (0.05s, 0.10s, 0.20s) on steady-state tracking error, emphasizing the role of system update rate.

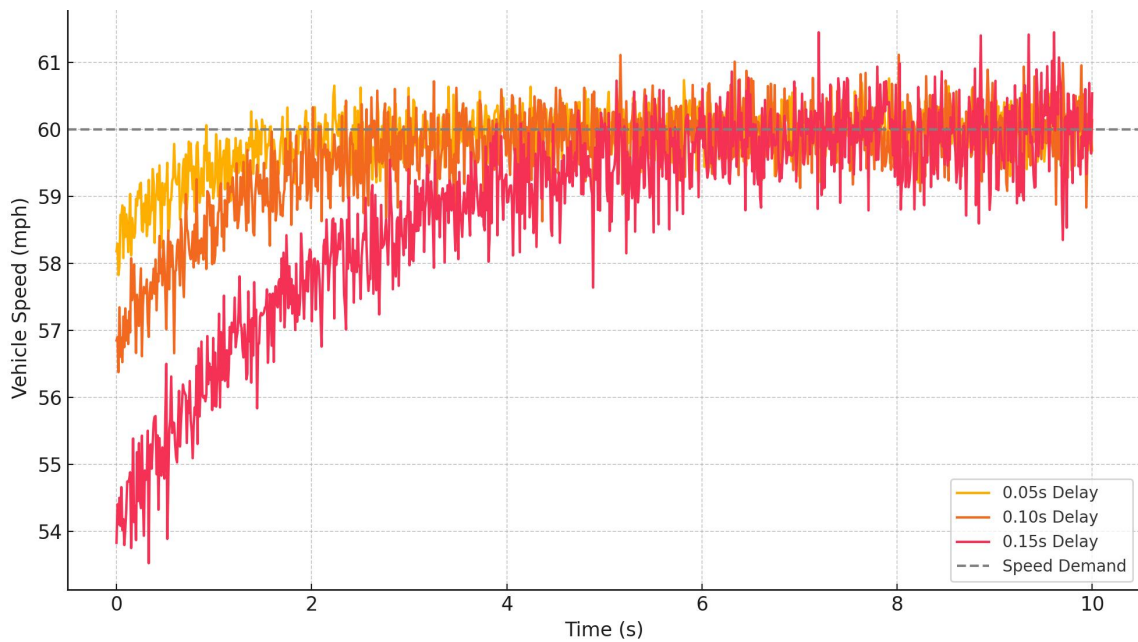


Figure 7 System response curves under increasing delay.

Figure 7 explores the effect of increasing communication delay (0.05s, 0.10s, 0.15s) on stability and responsiveness, simulating network-induced actuator lag.

These figures illustrate how controller accuracy degrades with coarser resolution, longer sampling intervals, and increased communication delays.

9. Summary and Evaluation

The digital PID controller implemented in this assignment met key performance goals:

- 1.Zero steady-state error under normal operating conditions
- 2.Stable tracking for realistic sampling rates and bit resolutions
- 3.Identified critical thresholds for bit depth, delay, and sampling period

However, as bit depth and sampling rate are reduced, or delay increases, the controller becomes sensitive to noise, delay, and quantization. For future development:

- 1.Anti-windup mechanisms could further stabilize the integral term
- 2.Adaptive PID or model-based control could optimize performance under dynamic conditions
- 3.Practical implementation would require real-time tuning tools and fail-safe handling logic

This exercise highlights the importance of integrating control design with digital implementation constraints in automotive software development. The findings offer a foundation for applying PID control in embedded vehicle systems, such as cruise control or regenerative braking modules.

10. References

Åström, K.J. and Hägglund, T. (2006). *Advanced PID Control. 1st ed.* Research Triangle Park: ISA.

Franklin, G.F., Powell, J.D. and Emami-Naeini, A. (2015). *Feedback Control of Dynamic Systems. 7th ed.* Boston: Pearson.

Katsuhiko, O. (2010). *Modern Control Engineering. 5th ed.* Upper Saddle River: Prentice Hall.

MATLAB. (2023). *PID Controller Design and Tuning. [online]* MathWorks. Available at: <https://www.mathworks.com/help/control/ug/pid-control-design.html> [Accessed 25 Mar. 2025].

Nise, N.S. (2020). *Control Systems Engineering. 8th ed.* Hoboken, NJ: Wiley.

Ogata, K. (2013). *Discrete-Time Control Systems. 2nd ed.* Pearson Education.