



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

MSc - Motorsport Engineering

Modelling Automotive Control
Systems in MATLAB

Module Name	Hybrid Powertrain and Control
Module Number	ACEB 7007
Student Name	Sean Doyle
Student Number	2305948
Date of Submission	30/03/2025

University of Wales Trinity Saint David
School of Engineering
2024-2025

Contents

Table of Figures	2
Table of Graphs.....	3
Table of Tables	3
Introduction.....	4
1 Task 1 - Yaw Stability Control	5
1.1 Background Review.....	5
1.2 Hardware and Communication	7
1.2.1 Sensor Inputs and Sample Rates	8
1.2.2 Actuators and Outputs.....	8
1.3 Software.....	9
1.4 Future Enhancements.....	10
2 Task 2 – MATLAB Practical.....	11
2.1 Encoding and Decoding.....	11
2.2 Controller Coding	13
2.2.1 Regulating Acceleration and Deacceleration	16
2.3 Coding Bits	19
2.4 Sample Times	21
2.5 Communication Network Delays.....	22
2.6 Concluding Comments.....	24
Reference List	25

Table of Figures

Figure 1: Electronic Limited Slip Differential Design Schematic for Implementation in Yaw Stability Control System (Left) Components (Piyabongkarn, et al., 2006), (Right) Cutaway eLSD Design (Jeep, 2016).	5
Figure 2: Electronic Limited Slip Differential and Yaw Stability Control Flowchart for Input Processing, Logic Control and Processing, Output and Feedback System. Image Retrieved and Adapted from (Annamalai, et al., 2013).	6
Figure 3: YSC Closed Loop System Diagram.	6
Figure 4: Vehicle Communication Diagram for eLSD YSC (Top) CAN Bus Usage (Eaton, n.d.), (Bottom) Modern FlexRay System (Guidara, 2016).....	7
Figure 5: Principle Workings of eLSD with YSC, (A) Overview of System as Open Differential, (B) Wheel Speed Differences Cause Gerotor Pump to Rotate Building Hydraulic Pressure, (C) YSC System Controls Pressure on Clutch Pack using Pressure Regulating Valve, (D) Change in Differential Locking and Distributes Torque to Wheels Accordingly (Eaton, n.d.).	9
Figure 6: Co-simulation for ABS (Left) Simulink System, (Right) VI-CarRealTime Graphical Results (VI-GRADE, 2020).....	9
Figure 7: Sample Rate and Bit Depth Effects on Error (Macatee, 2010).....	11
Figure 8: MATLAB Conversion for Demand Raw Signal to Physical Signal.	12
Figure 9: MATLAB Conversion for Control Physical Signal to Raw Signal for Actuator.	12
Figure 10: Baseline MATLAB PID Cruise Controller code.....	13
Figure 11: For Loop Range Values for Gain Tests.....	14
Figure 12: For Loop Iteration Values for Gain Parameter Adjustments.....	15
Figure 13: MATLAB Code for Acceleration/Deacceleration Limiting of Cruise Controller Outputs.	17
Figure 14: For Loop Value Ranges with Acceleration and Deacceleration Limits.	17
Figure 15: For Loop Iteration Values for Gain Parameter Adjustments with Acceleration and Deacceleration Limits.	17

Table of Graphs

Graph 1: Comparing Error Values for Triple Nested for Loop Gain Parameters.	14
Graph 2: A Graph Plotting Error at Each Simulation Step for a Triple Nested For Loop Comparing the Effects of Different Gain Parameter Values for P, I and D.	15
Graph 3: Refined Controller Results for Minimal Overshoot.....	16
Graph 4: Comparing Effects on Velocity Error for Different Gain Values with Acceleration and Deacceleration Limits.	18
Graph 5: Comparing Error Values for Triple Nested for Loop Gain Parameters with Acceleration/Deacceleration Limits.....	18
Graph 6: Refined Controller Results for Reduced Overshoot and Acceleration Control.	19
Graph 7: Accelerator/Decelerator Limited PID Controller with 14-Bit Rates.....	19
Graph 8: Basic Cruise Control PID Controller with 6-Bit Rate.....	20
Graph 9: Basic Cruise Control PID Controller with 4-Bit Rate.....	20
Graph 10: Acceleration/Deacceleration Limited Cruise Controller PID Outputs with Time Sample of 0.14 Seconds.	21
Graph 11: Acceleration/Deacceleration Limited Cruise Controller PID Outputs with Time Sample of 0.5 Seconds	21
Graph 12: Acceleration/Deacceleration Limited Cruise Controller PID Outputs with Communication Delays of 0.1 Seconds.....	22
Graph 13: Acceleration/Deacceleration Limited Cruise Controller PID Outputs with Communication Delays of 0.5 Seconds.....	23
Graph 14: Comparing Effects on Velocity Error for Different Gain Values with Acceleration and Deacceleration Limit when Communication Network Delays Reach 0.5 Seconds.	23

Table of Tables

Table 1: Effects on Control Signal Due to PID Gain Adjustments, Where T_i is Rise Time, T_s is Settling Time, OS is Overshoot and SSE is Steady State Error (Hussien, et al., 2015).	15
Table 2: PID Parameters	16

Introduction

Task 1 focuses on a yaw stability control system utilising active electronic limited slip differential technology. The review focuses on: operations, control aspects/systems, hardware, software and future enhancements of this controller. Special attention is paid to the operational differences and benefits between this system and alternative yaw stability control systems.

Task 2 consists of integrating a proportional-integral-derivative controller into a cruise control system. Research is conducted to specify suitable acceleration and deacceleration rates to optimise passenger comfort. The controller performance is also reviewed after changes to parameters such as bit rate and sample rate.

1 Task 1 - Yaw Stability Control

1.1 Background Review

Yaw stability control (YSC) is often implemented into vehicles for handling and safety purposes, where modern day vehicles have YSC integrated into a more complex system such as electronic stability control (ESC) (Annamalai, et al., 2013). Depending on the system complexity, type and vehicle drivetrain, the quantity of outputs varies. These output quantities vary due to vehicle components and the type and number of actuators used, whether they are controlled by some or all the following: active differentials, steering mechanism (steer by wire or mechanical) and braking system (Aripin, et al., 2014). The focus of this controller review will be on yaw stability controls with active electronic limited slip differential (eLSD) (figure 1) technology. The purpose of such a system is to help maintain vehicle longitudinal velocity/motion whilst improving lateral dynamic stability (Piyabongkarn, et al., 2006). Other developed systems utilise individual wheel braking in a YSC system to optimise vehicle lateral stability. As the objective of this system is to maintain longitudinal motion, braking applications are not required to keep the vehicle stable (Rajamani, 2006). Therefore, having minimal impact on longitudinal motion providing benefits for motorsport applications and automotive fuel economy. An overview of the eLSD YSC system is exemplified in the flow chart in figure 2, detailing inputs, logic processing and control, and outputs.

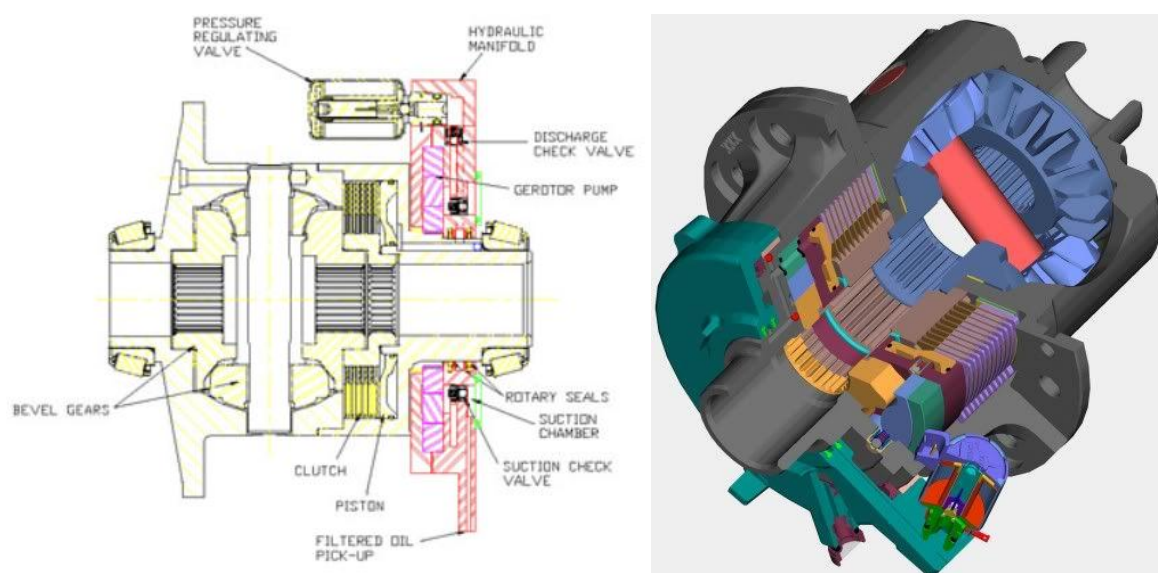


Figure 1: Electronic Limited Slip Differential Design Schematic for Implementation in Yaw Stability Control System (Left) Components (Piyabongkarn, et al., 2006), (Right) Cutaway eLSD Design (Jeep, 2016).

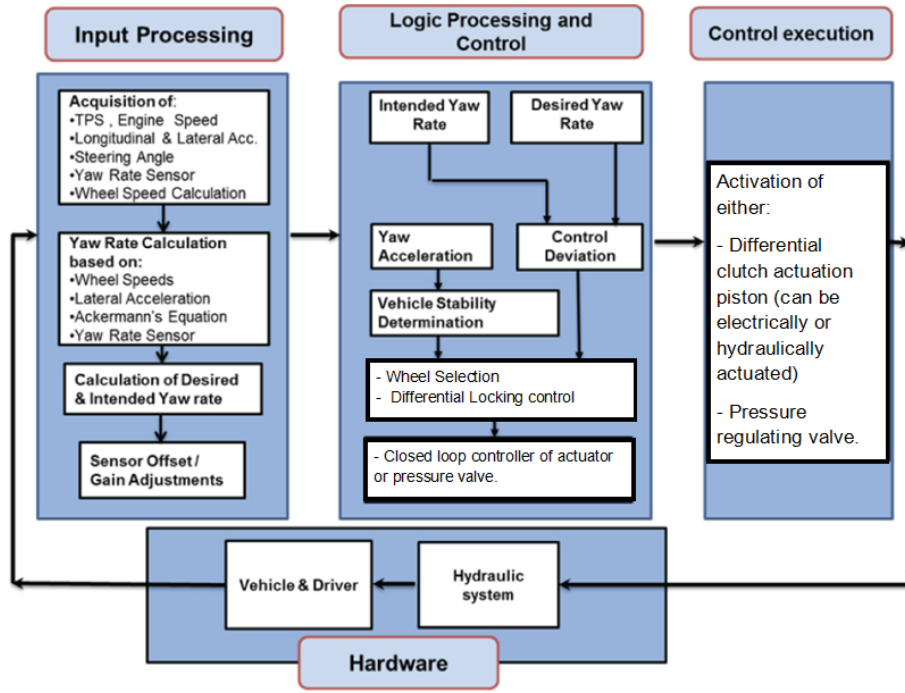


Figure 2: Electronic Limited Slip Differential and Yaw Stability Control Flowchart for Input Processing, Logic Control and Processing, Output and Feedback System. Image Retrieved and Adapted from (Annamalai, et al., 2013).

In a YSC closed-loop system, the input is the desired yaw rate, calculated from sensor data (driver inputs/requests). A comparator evaluates the difference between the desired and measured yaw rates, producing an error signal (Aripin, et al., 2014). This signal enters a controller algorithm, typically a proportional-integral-derivative (PID) or model-predictive-control (MPC) system is used, depending on input quantity and algorithm design/complexity the most suitable of the two is selected (Li, et al., 2020), to compute a suitable response. The controller sends a signal to the actuator, such as the eLSD pressure regulating valve or clutch actuation piston (Eaton, n.d.), which adjusts the pressure on the clutch pack within the differential (controlled variable) to better match the setpoint. The process/plant in this system is vehicle dynamics, specifically yaw rate. Disturbances include vehicle dynamics (e.g., exceeding tyre limits, load transfer), road conditions (e.g., friction changes due to weather or surface type, bumps), and external forces (e.g., aerodynamic changes from wind gusts). The output, vehicle yaw rate, is measured by a yaw rate sensor in the feedback loop and relayed to the comparator. Figure 3 illustrates this closed-loop control system.

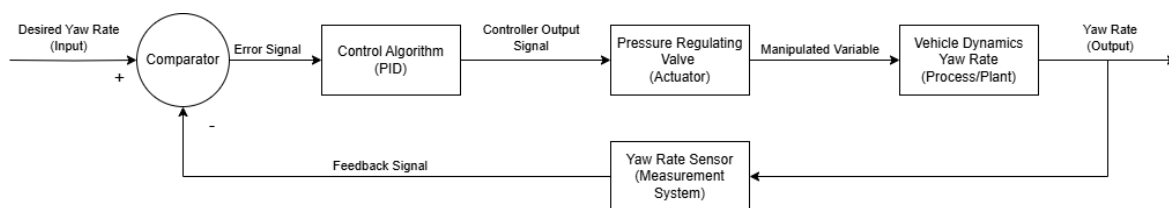


Figure 3: YSC Closed Loop System Diagram.

1.2 Hardware and Communication

Sensors and actuators are the main pieces of hardware for a YSC system. The schematic diagram in figure 4 demonstrates a communication system diagram detailing the linked control modules. At the centre of this system is either a controller-area-network (CAN) or FlexRay bus enabling efficient communication between the different control modules. A YSC system is often implemented into an electronic control unit (ECU) for the differential system, or powertrain control module rather than being a standalone module. This allows for faster more efficient data processing and response times. The benefits of using CAN and FlexRay buses have excellent communication reliability in trying environments, excellent error detection systems and noise rejection, both with high-speed data transfer (Voss, 2008). FlexRay tends to have a higher data rate than CAN bus which makes it more beneficial in driver assistant systems. The system utilises the sensors shown in figure 4 which are processed in associated control modules then relayed to the eLSD ECU. The YSC within calculates an output based on error and triggers an appropriate response at the differential.

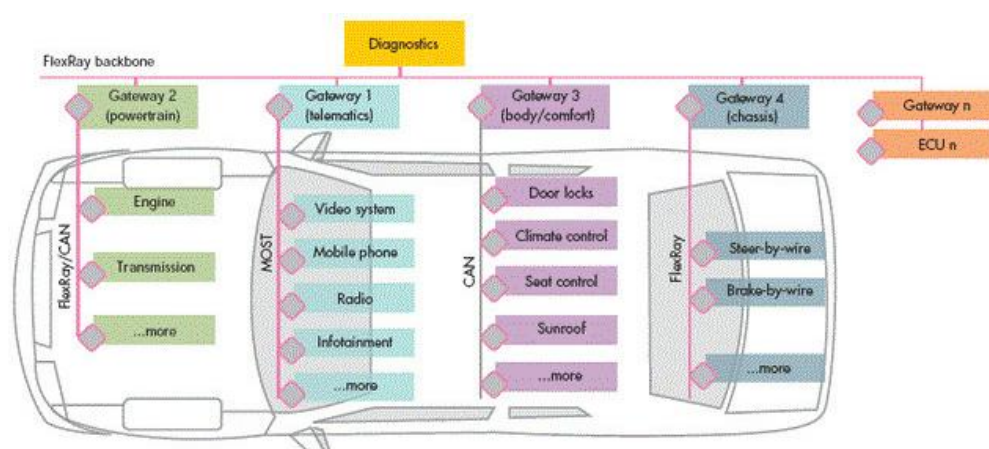
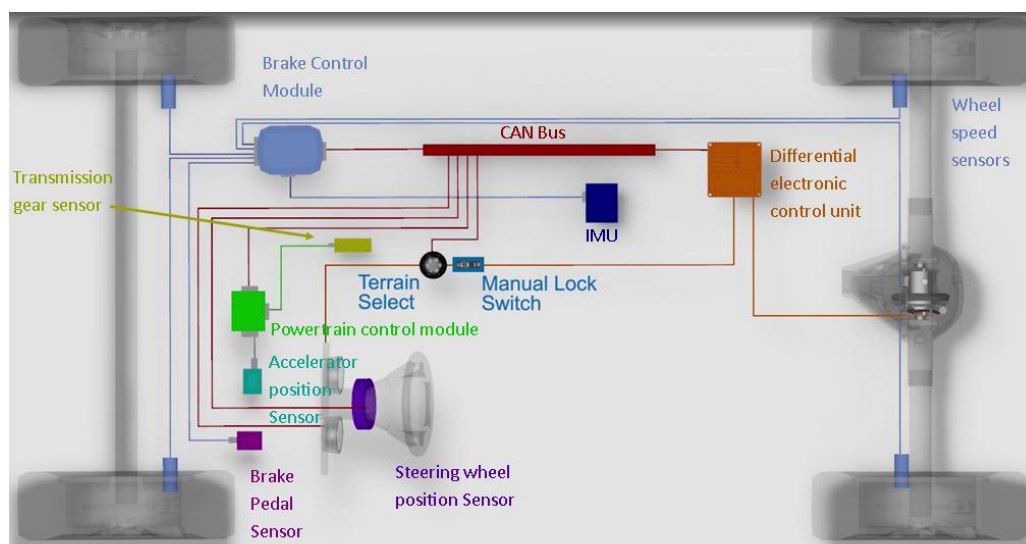


Figure 4: Vehicle Communication Diagram for eLSD YSC (Top) CAN Bus Usage (Eaton, n.d.), (Bottom) Modern FlexRay System (Guidara, 2016).

1.2.1 Sensor Inputs and Sample Rates

The system in this application utilises a variety of input sensors from around the vehicle (figure 4), the most common of which are: wheel speed, steering wheel position, accelerator position, brake position and a sensor for measuring yaw. This yaw measuring sensor can either be a: inertial measurement unit (IMU), yaw rate sensor or a more advanced global positioning system (GPS) (Piyabongkarn, et al., 2006). Due to the vehicle dynamic safety aspect and critical importance of an up-to-date logic control system, a high accuracy and sample rate is required to account for disturbances. The International Organisation for Standardisation (ISO) outlines key design principles for safety systems in production road vehicles for control design in the ISO 26262 standard (Schildbach, 2018). Although a specific sample rate is not stated within this standard, the general acceptance within industry is a higher sampling frequency (100-200 Hz) is used for vehicle safety systems. However, higher sampling rates require higher processing power to deal with the extra data which may not be required. For an inertial sensor sample rate for ride quality measures, a stated frequency of 64 Hz is tested and proved to be most effective (Bridgelall, 2015). Contradicting this sample rate frequency, 100 Hz is stated as a more suitable sample rate for wheel speed sensors, steering wheel position sensor and IMU systems (Li, et al., 2014). Both these investigations however, focused on the lower costing element of sampling rates whilst maintaining an effective safety system. Contrary to this, a higher sampling rate with greater costs would improve system response but may see only marginal benefits for this additional price.

1.2.2 Actuators and Outputs

In this eLSD system, the actuator is an electronically activated hydraulic pressure-regulating valve positioned at the end of the differential case. Different to traditional systems that rely solely on pressure generated by the gerotor pump from wheel speed differences, this valve provides precise control over the hydraulic pressure applied to pistons that engage the clutch pack (Piyabongkarn, et al., 2006). This enables the eLSD to function as an open, limited slip, or fully locked differential. The system operates based on inputs from wheel speed sensors and other vehicle data, processed by the ECU's YSC algorithm (demonstrated figure 5). Additionally, this eLSD design is more energy-efficient than alternative eLSDs and outperforms passive LSDs, which cannot be integrated into YSC systems due to their fixed mechanical responses.

The outputs of this system are the controlled variable itself (yaw rate), the controller output signal which is sent to the electronically controlled hydraulic pressure regulating valve. Outputs of the eLSD itself are distributed torque to the drive wheels (Piyabongkarn, et al., 2006), whilst the feedback system utilises outputs of measured yaw rates from the IMU to provide values for output error.

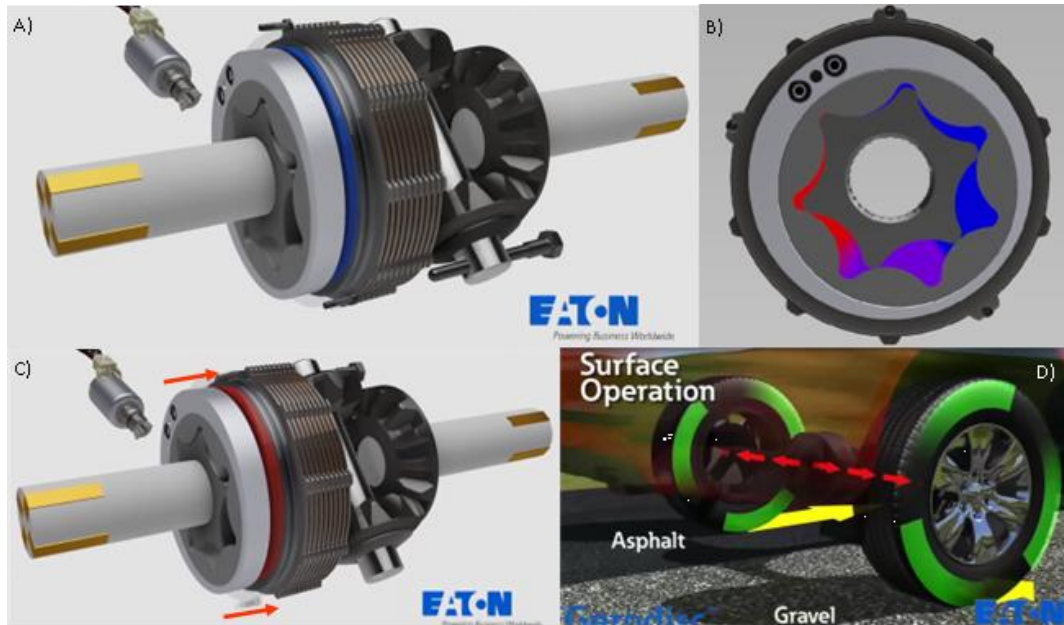


Figure 5: Principle Workings of eLSD with YSC, (A) Overview of System as Open Differential, (B) Wheel Speed Differences Cause Gerotor Pump to Rotate Building Hydraulic Pressure, (C) YSC System Controls Pressure on Clutch Pack using Pressure Regulating Valve, (D) Change in Differential Locking and Distributes Torque to Wheels Accordingly (Eaton, n.d.).

1.3 Software

Development software for vehicle YSC systems with eLSDs can be completed in initial stages using vehicle dynamics simulation software such as VI-Grade and MATLAB/Simulink. YSC system algorithms constructed in Simulink can be run in co-simulations with VI-Grade to assess virtual vehicle dynamic responses of initial developments of control algorithms (VI-GRADE, 2020). From these results, adaptations can be made to the control algorithms in MATLAB/Simulink which can then be implemented into testing units within vehicles. Figure 6 demonstrates a co-simulation for ABS.

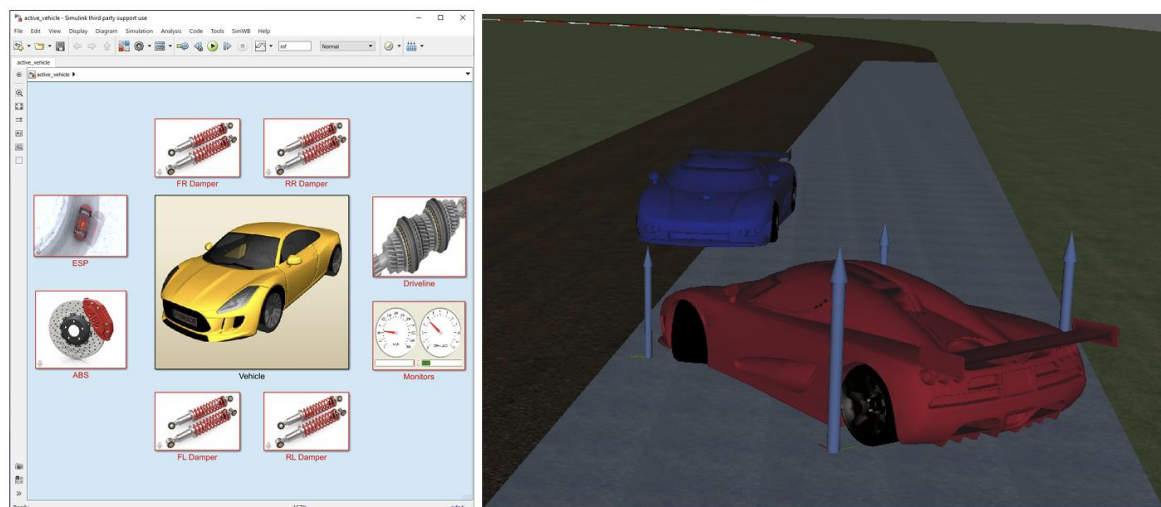


Figure 6: Co-simulation for ABS (Left) Simulink System, (Right) VI-CarRealTime Graphical Results (VI-GRADE, 2020).

The primary methods of vehicle diagnostic systems is the on-board-diagnostic 2 (OBD 2) programme with a standardised set of protocols for basic vehicle data such as live engine parameters, emissions data and access to diagnostic trouble codes (DTC) (McCord, 2011). Automotive manufacturers are required to adhere to specific protocols for these systems, however, other vehicle dynamics control modules/units tend to use manufacturer specific protocols. To access these specific modules, the Bosch GRADE X (other systems are available) is a diagnostic software that uses a vehicle communication interface (VCI) to retrieve the vehicle identification number (VIN) in order for correct protocol communication and diagnostics (Bosch, 2025). With the use of an OBD 2 diagnostic system live parameters can be monitored and analysed, however accuracy will be down to the vehicles sensor resolutions and calibration, instead of an externally calibrated measuring device. A combination of all these different software's and Microsoft Excel can be used for system effectiveness analysis.

1.4 Future Enhancements

Future developments of this software and system could see eLSD with YSC system integration into electric vehicles and modern powertrain designs (including hybrids) for a kinetic energy recovery system. Adaptations to design could see conversion from a speed sensing device to torque sensing, where reaction times are significantly quicker/instant. Due to the design of the eLSD, more servicing intervals are required for clutch pack design differentials due to wear; this therefore implicates that alternative designs should be considered to reduce operating costs for consumers (Palazzolo, 2009). Higher sample rate sensors could also be implemented for more responsive reactions to disturbances and advanced safety systems could see beneficial cooperation between ABS, traction control and launch control systems with eLSD YSC systems.

2 Task 2 – MATLAB Practical

2.1 Encoding and Decoding

The decoding of signal data is predominantly required for humans to easily process decimal numbers rather than binary. It also enables the PID controller to function as a simpler written code in MATLAB where calculations can be completed with double-precision floating-numbers (binary) for high levels of accuracy (Mathworks, 2023). The standard setup configurations for the sensor, demand and actuator signals are 16 bits or 2 bytes (Henderson, 2009). Output signals from sensors can often be analogue, for this signal to be processed in a controller, it must be digitalised into binary format. This uses an analogue to digital converter (ADC) which samples values at specific sample times (Patronis, et al., 2013). The effects of increasing sample rates and bit depth are demonstrated in figure 7. If sample rate is too low, the data is not updated regularly enough for some applications, when bit depth is too low, data clipping can occur as the analogue signal exceeds the bit depth capabilities of the ADC converter. Utilising the correct sample rate and bit depth is vital to ensure the required resolution and data capturing intervals fit within the sensor outputs and account for disturbances frequently enough for controller outputs.

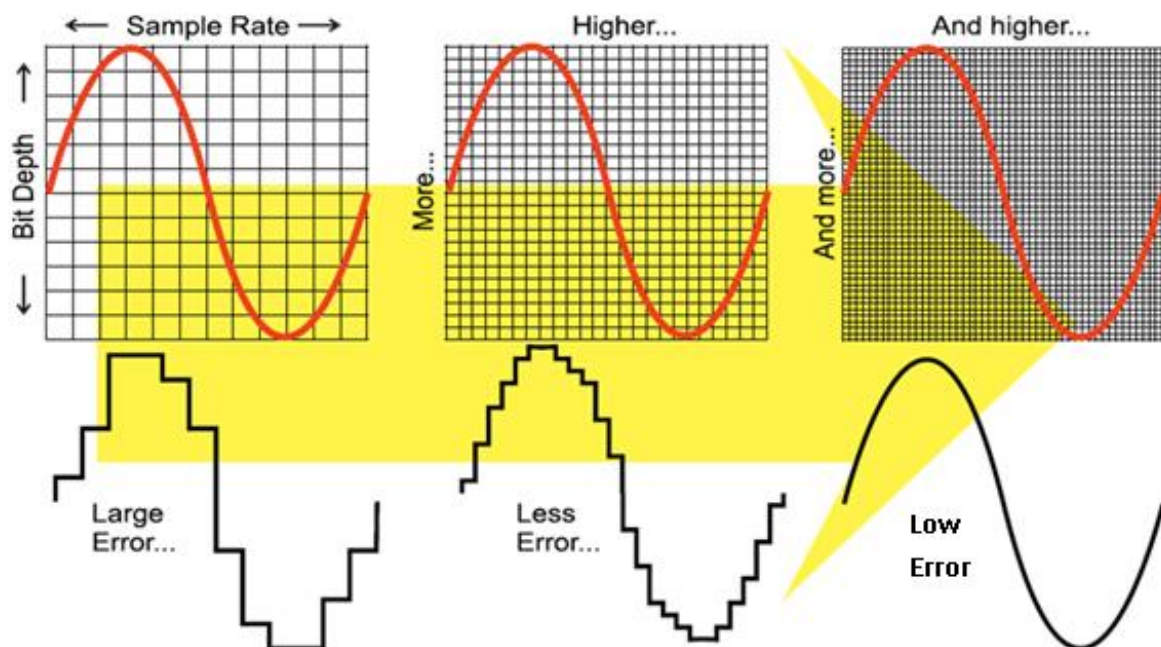


Figure 7: Sample Rate and Bit Depth Effects on Error (Macatee, 2010).

Within the cruise control system, data signals from the cruise control driver demand element ‘demand_raw’ must be converted from binary to decimal for coding within the controller system. This is achieved by the binary to decimal function ‘bin2dec’ in MATLAB (Mathworks, 2025). The raw signal can then be converted into a physical signal using the formula below (MATLAB code for this is demonstrated figure 8), where ‘m’ is minimum demand, ‘M’ is maximum demand and ‘R’ is the range for raw signal values:

$$X_{Physical} = \frac{X_{Raw} \times (M - m)}{R} + m$$

70	%% decode raw signal (from x-bit binary to mph)
71	
72	demand = ((bin2dec(demand_raw).*(M_demand-m_demand))./R)+m_demand;

Figure 8: MATLAB Conversion for Demand Raw Signal to Physical Signal.

The encoding of the physical controller signal to raw (binary) signal utilises the ‘dec2bin’ function built into MATLAB (Mathworks, 2025). The encoding is based on the formula shown below where ‘control’ is the physical signal, ‘m_ControlSig’ refers to the controller signal minimum demand, ‘M_ControlSig’ the controller signal maximum demand and ‘R’ the signal range. This mathematical equation is applied in MATLAB as demonstrated in figure 9, where the raw signal is then used to trigger actuators.

$$X_{Raw} = \frac{(X_{Physical} - m) \times R}{M - m}$$

98	%% encode control signal into x-bit binary number
99	
100	control_raw = dec2bin((((control)-m_ControlSig).*(R))./(M_ControlSig-m_ControlSig));

Figure 9: MATLAB Conversion for Control Physical Signal to Raw Signal for Actuator.

2.2 Controller Coding

Task 2 consisted of implementing a PID controller into a cruise control system. A PID controller benefits over other more basic systems due to inclusion of a proportional, integral and derivative component. The proportional control action is relative and proportional to the current control error, if only a proportional controller is present, a steady state error can occur. An integral component action is proportional to the integral of previous error, the addition of an integral error is to reduce the steady state error close to if not zero. The final aspect derivative focuses on the predicted control error values where it can anticipate incorrect trends and help prevent this (Visioli, 2006). The PID controller can be summarised by saying the proportional element accounts for ‘present’ error, integral ‘past’ error and derivative ‘future’ error. The base equation for a PID controller is demonstrated in the equation below, which outlines the fundamental principle within the baseline MATLAB code in figure 10, line 84. The addition of the if loop prevents the controller from exceeding the throttle and braking percentages of 100% being full throttle and ‘-100%’ for full braking.

$$I_{out} = K_P \left(e + K_I \int e \, dt + K_D \frac{de}{dt} \right) + I_0$$

```
73      %% calculate control signal and error (range of control [-100,100])
74
75      error = demand-velocity;
76
77      Int_vec(1)=0;
78      e_vec(1)=0;
79      y_vec(1)=velocity;
80
81      Integral = Int_vec(j)+error.*sim_step;
82      Derivative = (error-e_vec(j))./sim_step;
83
84      control = P.*(error+I*Integral+D.*Derivative)+demand;
85      if control>100
86          control =100;
87      elseif control<-100
88          control = -100;
89      end
```

Figure 10: Baseline MATLAB PID Cruise Controller code.

The initial controller, as shown in figure 10, is designed to minimize steady-state error by adjusting PID gain parameters. This is achieved through a triple nested for loop, where each gain: proportional, integral, and derivative, value is tested in every combination of the three ranges. The configuration with the smallest error, least overshoot and most reasonable representation of vehicle capabilities is selected.

Initially, a broad range of values is used to define the test parameters. Based on these results, a more refined range is tested to identify optimal values. Due to computational limitations, narrowing the range after an initial evaluation was necessary. Ideally, evaluating the entire range at a consistent step size would be preferred, but this was impractical due to computational constraints.

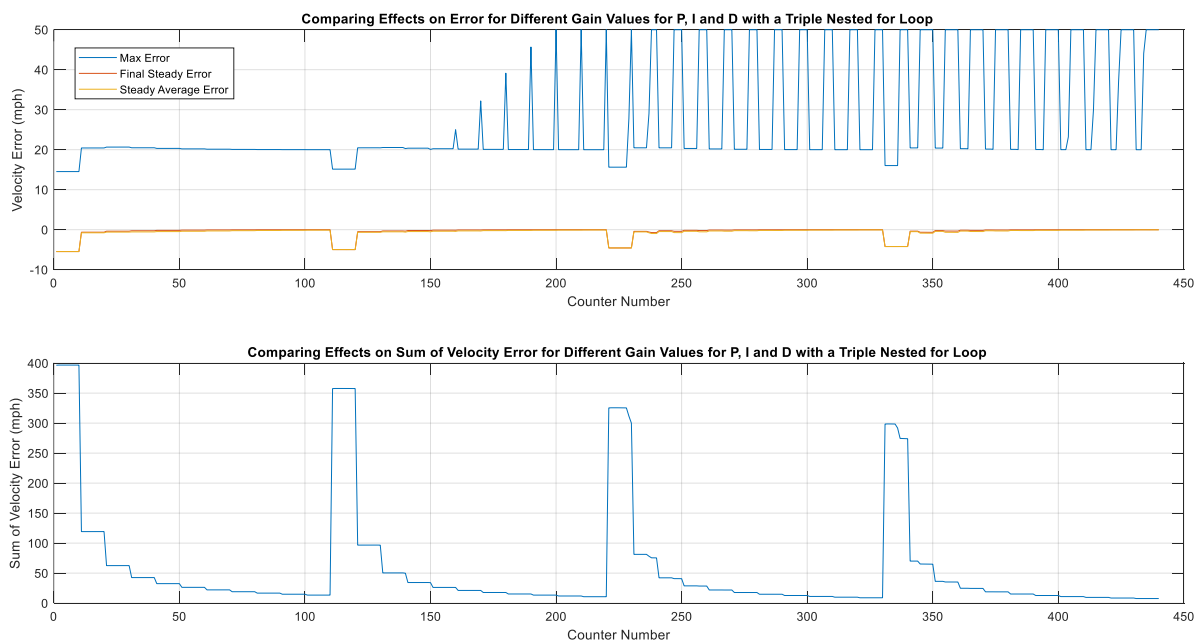
The final for loop configuration is illustrated in figure 11, derived from previously defined larger ranges. Results in graph 1 display the maximum error, average steady-state error, and cumulative error for each loop iteration (counter). However, while maximum and steady-state error values may appear low in certain cases, excessive overshoot, as depicted in graph 2, remains a concern. Graph 2 also provides opportunity to analyse sample point errors, overshoot, undershoot, and steady-state error for each simulation interval and loop/counter.

```

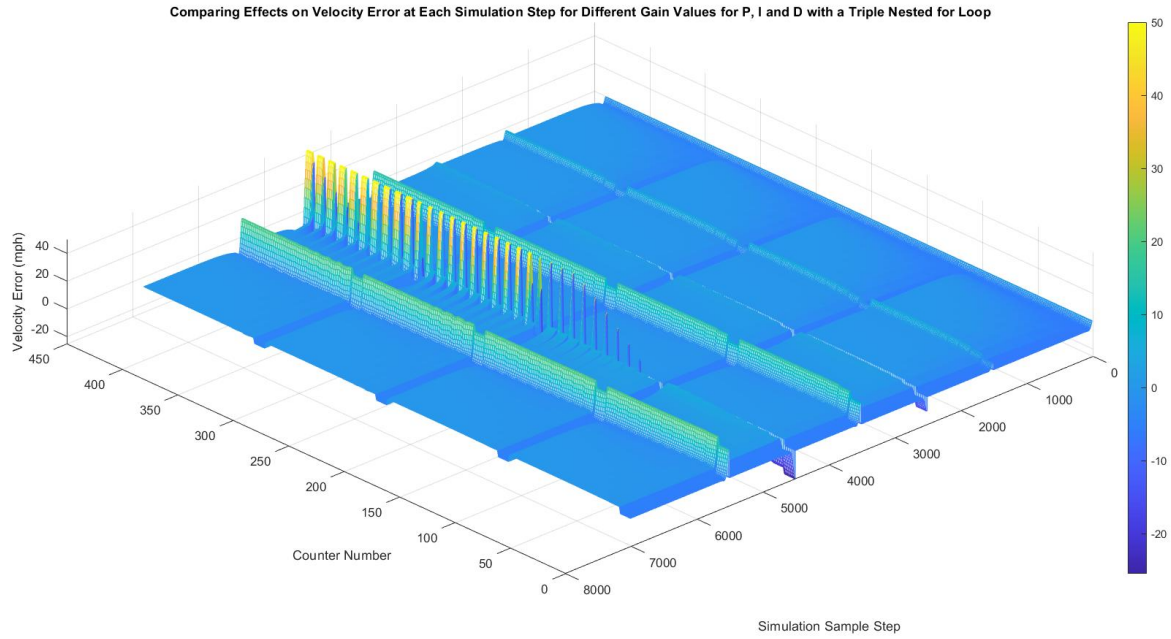
13  [ ] for P=0.8:0.2:1.4
14  [ ] for I=0:0.5:5
15  [ ] for D=0.001:0.001:0.01

```

Figure 11: For Loop Range Values for Gain Tests.



Graph 1: Comparing Error Values for Triple Nested for Loop Gain Parameters.



Graph 2: A Graph Plotting Error at Each Simulation Step for a Triple Nested For Loop Comparing the Effects of Different Gain Parameter Values for P, I and D.

The initial values are retrieved from counter number 44 (demonstrated figure 12), when studying the results, overshoot is too high for a suitable cruise control system and therefore in accordance with table 1, the values for counter number 49 are tested instead resulting in a lower overshoot value (Hussien, et al., 2015), further refinement led to the ‘refined’ values in table 2 with a reduced overshoot and lowest steady state error of -0.11271, the results are illustrated in graph 3.

```
Counter Number 44 Steady State Error -0.470767 Max Error 20.328064 P is 0.8000 I is 2.0000 D is 0.0040
Counter Number 45 Steady State Error -0.470847 Max Error 20.328064 P is 0.8000 I is 2.0000 D is 0.0050
Counter Number 46 Steady State Error -0.470893 Max Error 20.328064 P is 0.8000 I is 2.0000 D is 0.0060
Counter Number 47 Steady State Error -0.470961 Max Error 20.328064 P is 0.8000 I is 2.0000 D is 0.0070
Counter Number 48 Steady State Error -0.471064 Max Error 20.330353 P is 0.8000 I is 2.0000 D is 0.0080
Counter Number 49 Steady State Error -0.471167 Max Error 20.330353 P is 0.8000 I is 2.0000 D is 0.0090
Counter Number 50 Steady State Error -0.471247 Max Error 20.330353 P is 0.8000 I is 2.0000 D is 0.0100
Counter Number 51 Steady State Error -0.385241 Max Error 20.209045 P is 0.8000 I is 2.5000 D is 0.0010
```

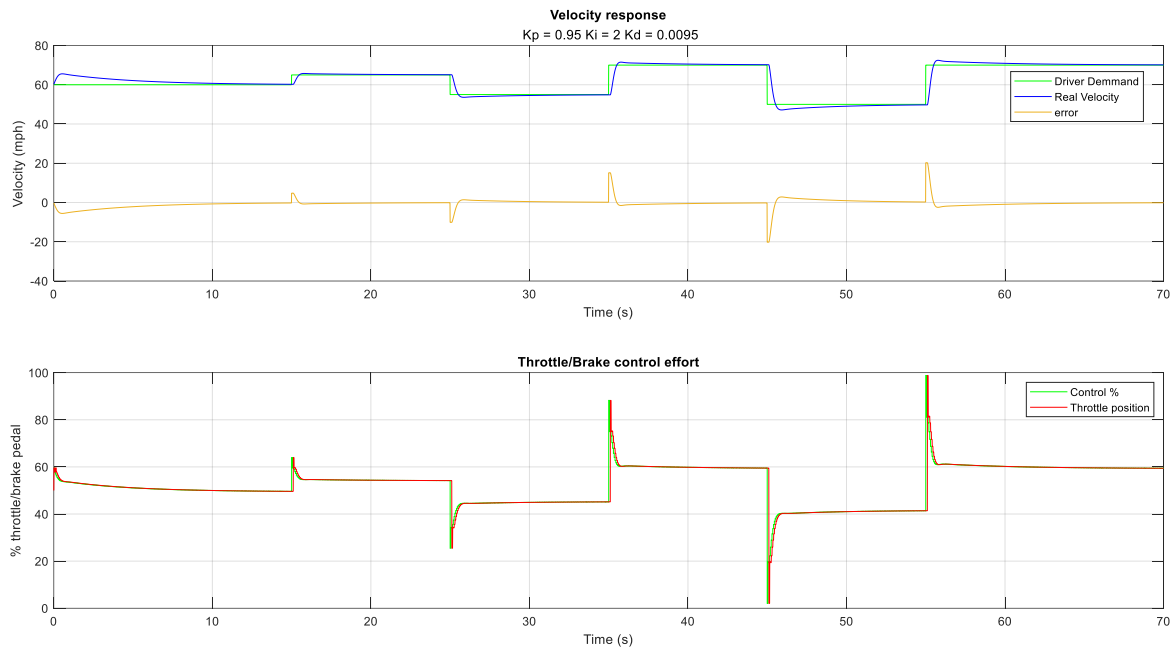
Figure 12: For Loop Iteration Values for Gain Parameter Adjustments.

Table 1: Effects on Control Signal Due to PID Gain Adjustments, Where T_i is Rise Time, T_s is Settling Time, OS is Overshoot and SSE is Steady State Error (Hussien, et al., 2015).

	Effect of Performance			
	T_r	T_s	OS	SSE
P	Decrease	Small Change	Increase	Decrease
I	Decrease	Increase	Increase	Eliminate
D	Small Change	Decrease	Decrease	Small Change

Table 2: PID Parameters

Gain Parameter (K)	Counter 44	Counter 49	Refined
P	0.8	0.8	0.95
I	2	2	2
D	0.004	0.009	0.0095



Graph 3: Refined Controller Results for Minimal Overshoot.

2.2.1 Regulating Acceleration and Deacceleration

A cruise control system is implemented into vehicles as a comfort feature, when the cruise control demand changes due to driver requests or with more modern systems radar in adaptive cruise control, the system is expected to respond in a timely but comfortable manner. This results in maximum acceleration/deacceleration rates defined for adaptive cruise control systems by the ISO norm 22179. At 20m/s the acceleration/deacceleration rate are not permitted to exceed $+4\text{ms}^{-2}/-5\text{ms}^{-2}$ whilst at 5m/s the rates are altered to $+2\text{ms}^{-2}/-3.5\text{ms}^{-2}$ respectively (Happee, et al., 2023). The addition of a while loop compares the calculated acceleration value to the maximum and minimum limits prespecified. If these values are greater than these limits, a small value is removed/added to the control value until acceleration is within the determined limits. Figure 13 demonstrates this additional code.

```

91         if control>100
92             control =100;
93         elseif control<-100
94             control = -100;
95         end
96
97     Accel = ((velocity/2.23693629)-(y_vec(j)/2.23693629))./sim_step;
98     throttle_settingest=throttle_setting;
99
100    if ((Accel>AccelMax) || (Accel<AccelMin))
101        while ((Accel>AccelMax) || (Accel<AccelMin))
102            if Accel > AccelMax
103                control = control-0.02;
104            elseif Accel < AccelMin
105                control = control+0.02;
106            end
107
108            [velocity_rawest, velocity_realest,throttle_settingest] = getRawVelocity...
109                |(n_bits_sensor_signal,velocity,throttle_settingest, sim_step,control);
110            Accel = ((velocity_realest/2.23693629)-(velocity/2.23693629))./sim_step;
111        end
112    end

```

Figure 13: MATLAB Code for Acceleration/Deacceleration Limiting of Cruise Controller Outputs.

The for-loop values are adjusted using the same methodology as before, the final range for testing is demonstrated in figure 14 in addition to the iteration counter values in figure 15. The counter number 291 demonstrates the best compromise between lowest steady state error and minimal overshoot, as also demonstrated in graphs 4 and 5. Therefore, the gain parameters are: P = 1.1, I = 1 and D = 0.003.

```

18     for P=1.04:0.01:1.1
19         for I=0.5:0.25:1.5
20             for D=0.001:0.001:0.009

```

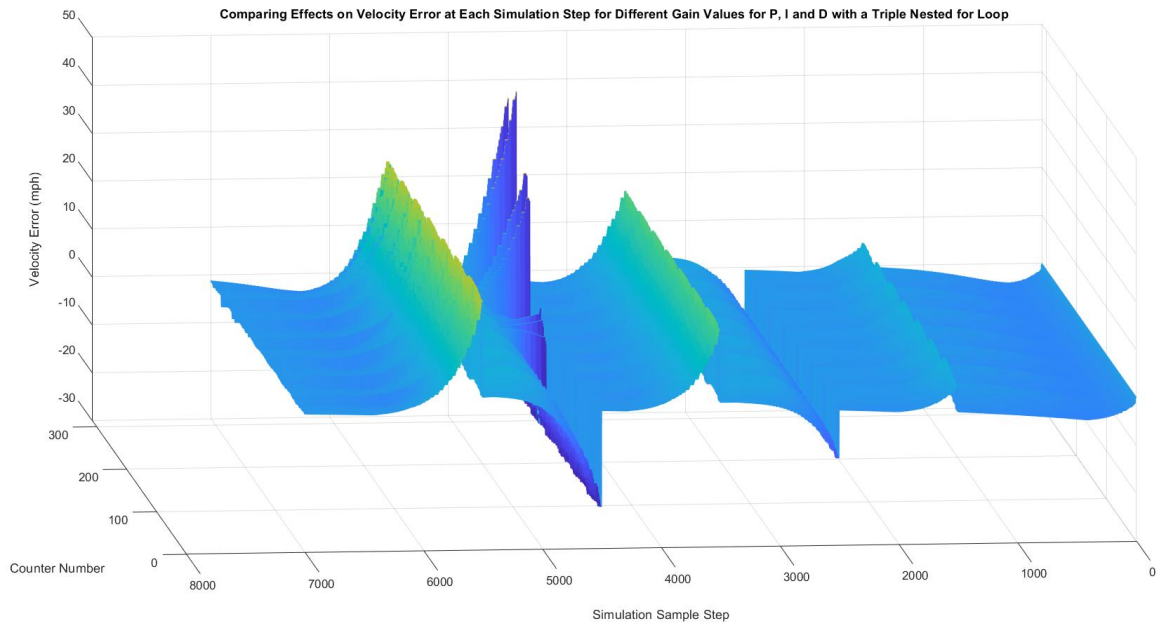
Figure 14: For Loop Value Ranges with Acceleration and Deacceleration Limits.

```

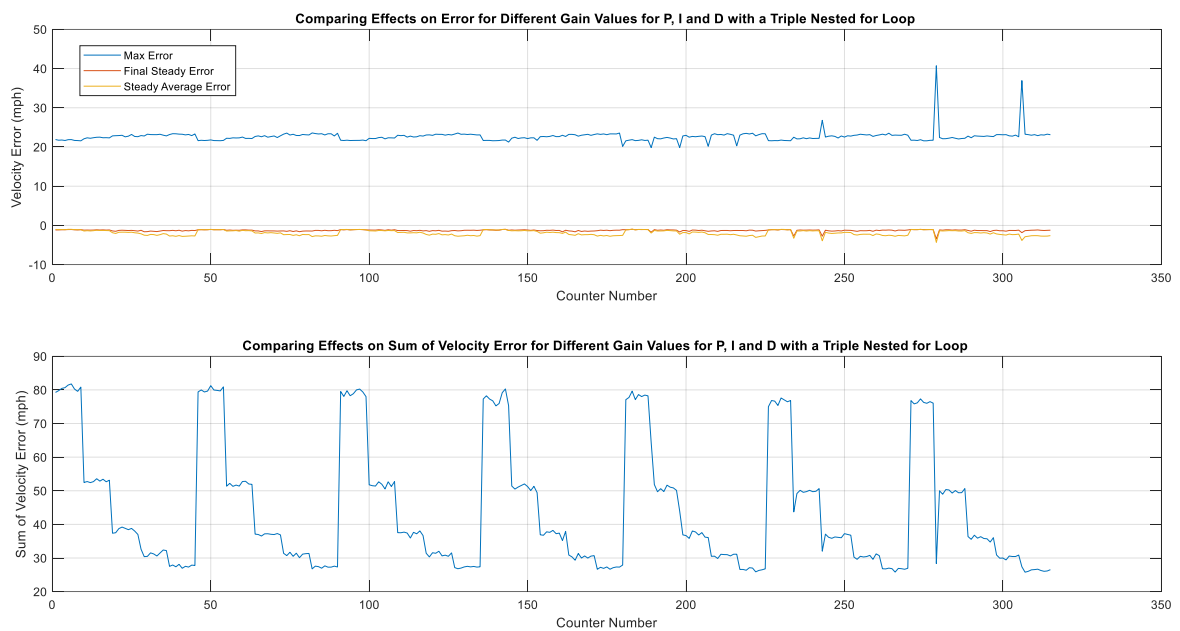
Counter Number 290 Steady State Error -2.027822 Max Error 22.355957 P is 1.1000 I is 1.0000 D is 0.0020
Counter Number 291 Steady State Error -1.822180 Max Error 22.859497 P is 1.1000 I is 1.0000 D is 0.0030
Counter Number 292 Steady State Error -1.840601 Max Error 22.777100 P is 1.1000 I is 1.0000 D is 0.0040
Counter Number 293 Steady State Error -1.928422 Max Error 22.738190 P is 1.1000 I is 1.0000 D is 0.0050
Counter Number 294 Steady State Error -1.844019 Max Error 22.692413 P is 1.1000 I is 1.0000 D is 0.0060
Counter Number 295 Steady State Error -1.917593 Max Error 22.816010 P is 1.1000 I is 1.0000 D is 0.0070
Counter Number 296 Steady State Error -2.207237 Max Error 22.722168 P is 1.1000 I is 1.0000 D is 0.0080
Counter Number 297 Steady State Error -1.856668 Max Error 22.655792 P is 1.1000 I is 1.0000 D is 0.0090

```

Figure 15: For Loop Iteration Values for Gain Parameter Adjustments with Acceleration and Deacceleration Limits.

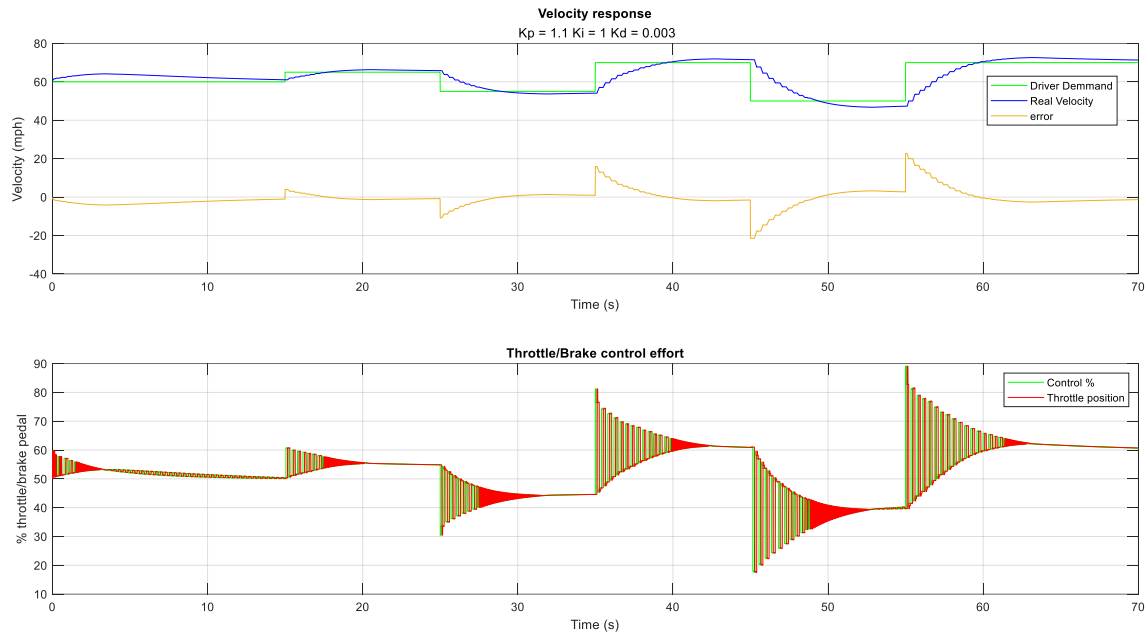


Graph 4: Comparing Effects on Velocity Error for Different Gain Values with Acceleration and Deacceleration Limits.



Graph 5: Comparing Error Values for Triple Nested for Loop Gain Parameters with Acceleration/Deacceleration Limits.

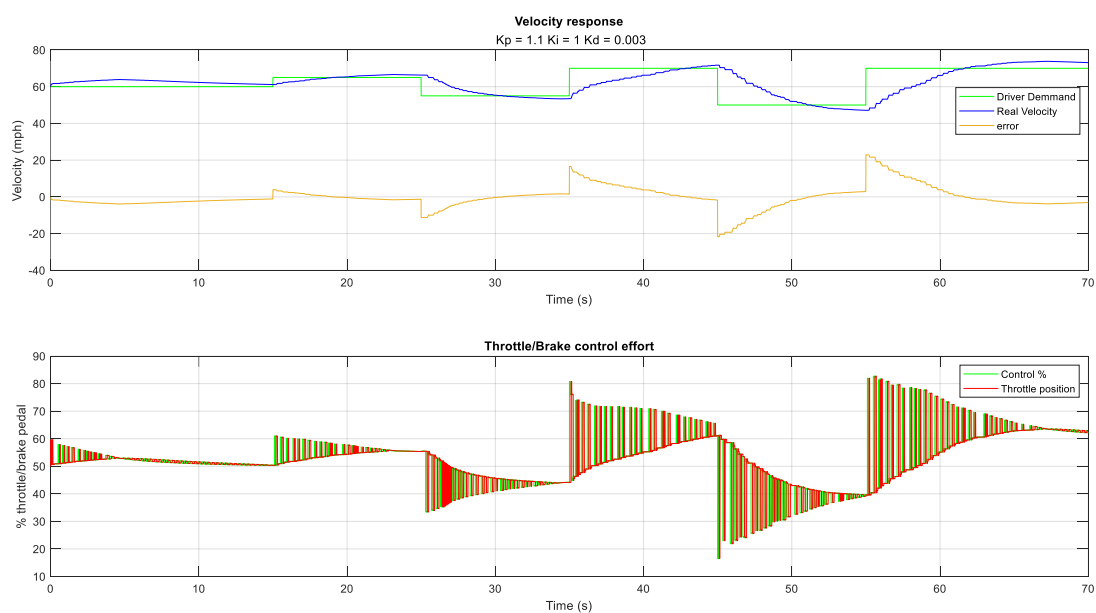
The results for the simulation with the selected gain parameters are demonstrated in graph 6. Whilst the rate of acceleration/deacceleration have been reduced significantly in comparison to graph 3, noticeable by the more gradual gradients, the steady state error has increased due to the systems increased response delay. However, the control signals are unsteady causing twitching during acceleration changes when trying to meet demands; ultimately impacting ride comfort and leaving room for system refinement.



Graph 6: Refined Controller Results for Reduced Overshoot and Acceleration Control.

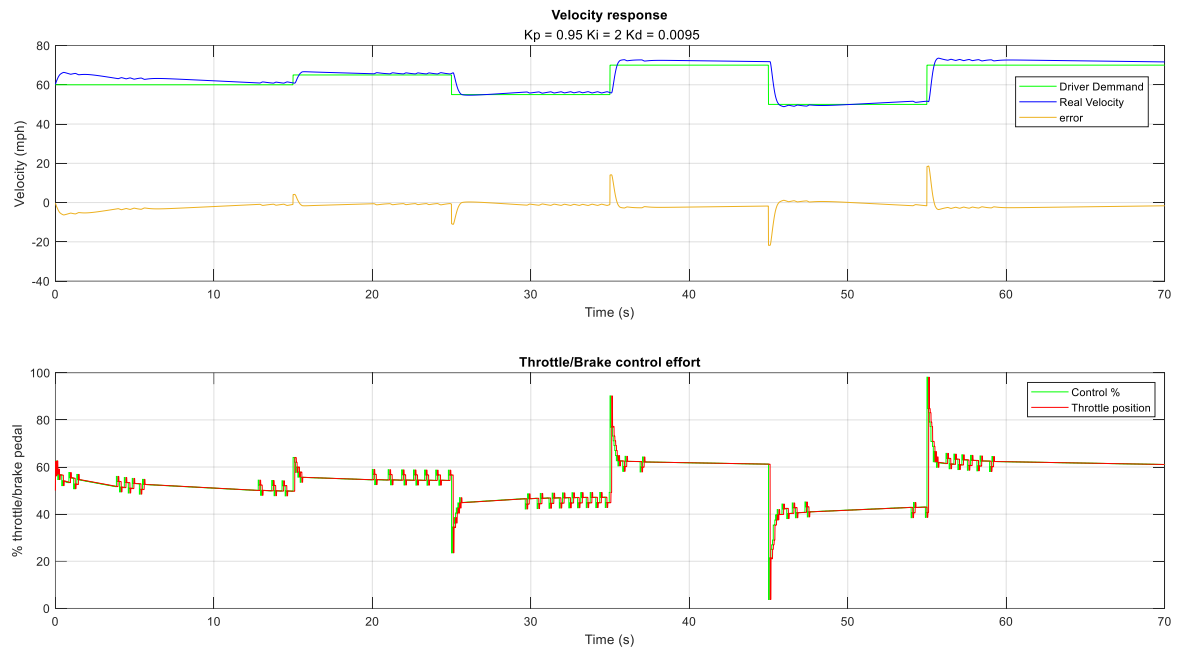
2.3 Coding Bits

Considering the PID controller with acceleration/deceleration limits, lowering the bit rate to below 16 causes the system to act in an unpredictable manor that would be unsafe for applications as demonstrated in graph 7 with a universal 14-bit rate. The steady state error increases and controller outputs become more unreliable and scattered. The cause of this is a reduced capture resolution where sensor and actuator signals are less precise causing abrupter changes (Henderson, 2009).

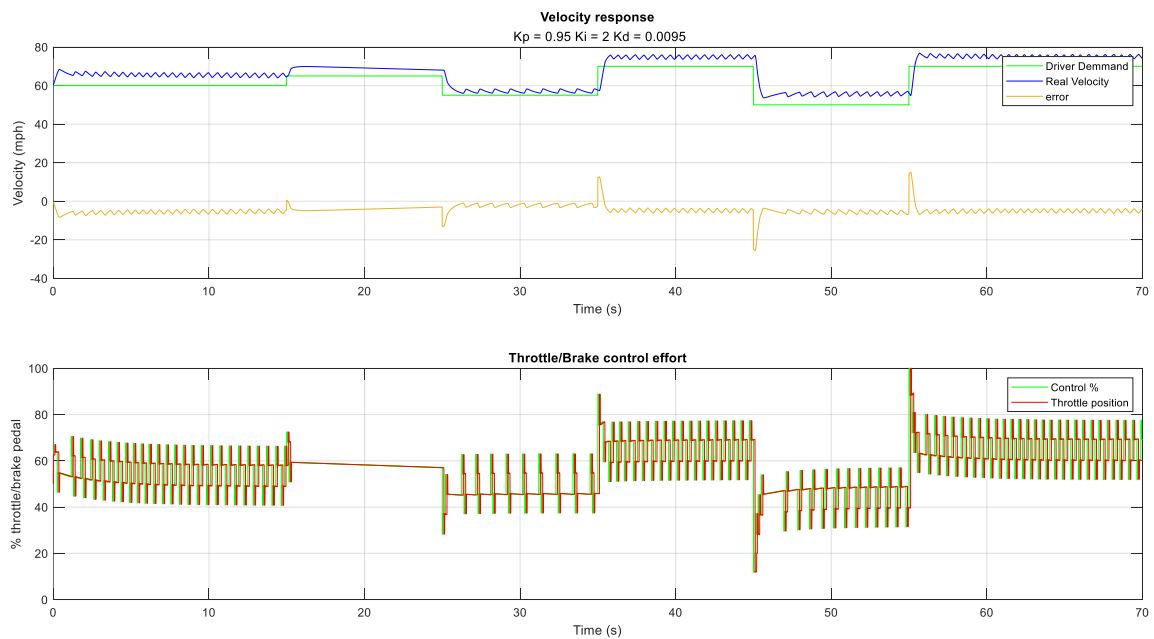


Graph 7: Accelerator/Decelerator Limited PID Controller with 14-Bit Rates.

On the other hand, the basic PID controller without acceleration/deacceleration limits is capable of operating at lower bit rates, demonstrated in graph 8 where the system is running at 6-bits. The control signal becomes less smooth and is starting to impact vehicle velocity, ride comfort and steady state error (Happee, et al., 2023). When bit rate is too low (4-bits), the steady state error no longer tends towards zero (graph 9) and the system is no longer fit for purpose without adaptations to the PID.



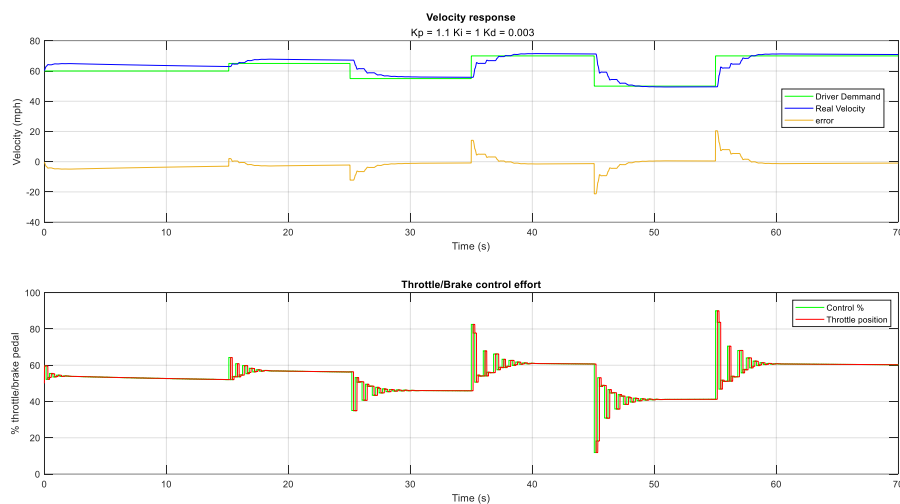
Graph 8: Basic Cruise Control PID Controller with 6-Bit Rate.



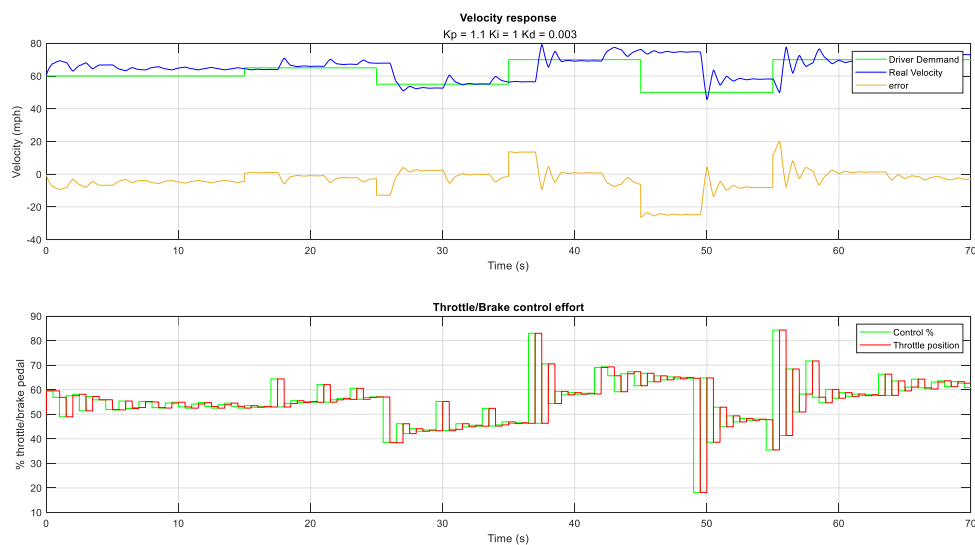
Graph 9: Basic Cruise Control PID Controller with 4-Bit Rate.

2.4 Sample Times

Increasing time periods between system sample times causes reduced captures of environmental disturbances acting on the measured parameters. The sensors therefore relay less frequent information causing the system to react to bigger changes causing a stepped controller output, as demonstrated in graph 10 and exemplified in figure 7, instead of a gradual gradient (Patronis, et al., 2013). The effects of this in a cruise control system are reduced ride comfort, larger errors and clunky responses. Graph 10 utilises a 0.14 second sampling time illustrating these issues. If the sample time increases further, the system response gets worse to the point at which acceleration limits can no longer effectively be implemented and a delayed system output causes lag between the vehicle response and the input driver demand. Graph 11 demonstrates the severity of increased sample time at 0.5 seconds, showing large delays between demand and outputs. The system also demonstrates more underdamping characteristics.



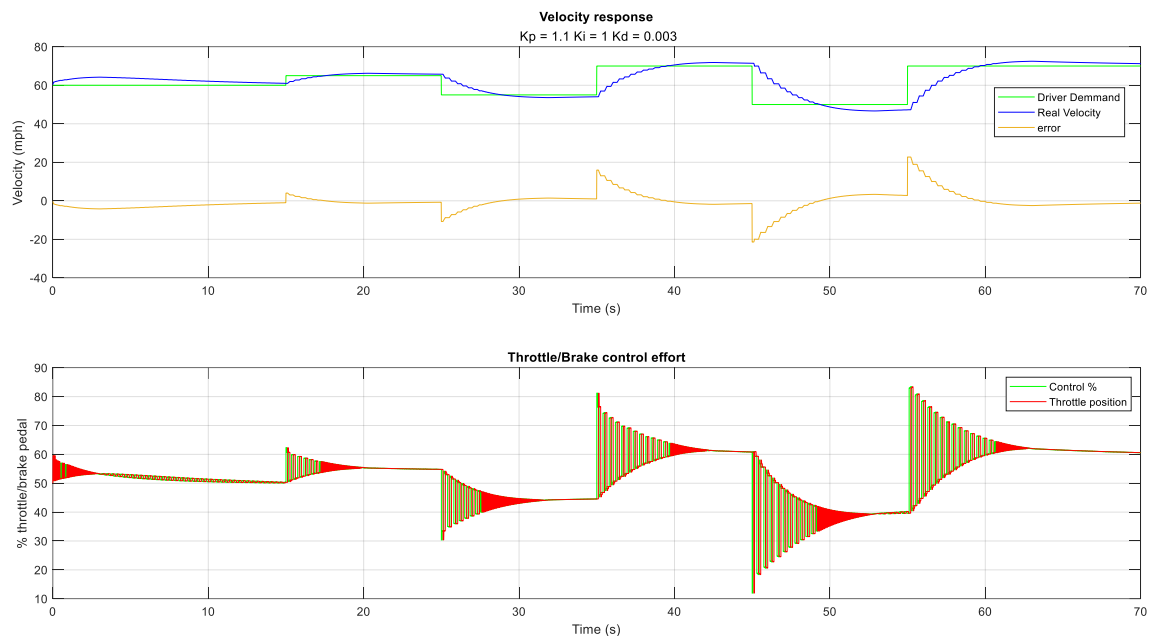
Graph 10: Acceleration/Deceleration Limited Cruise Controller PID Outputs with Time Sample of 0.14 Seconds.



Graph 11: Acceleration/Deceleration Limited Cruise Controller PID Outputs with Time Sample of 0.5 Seconds

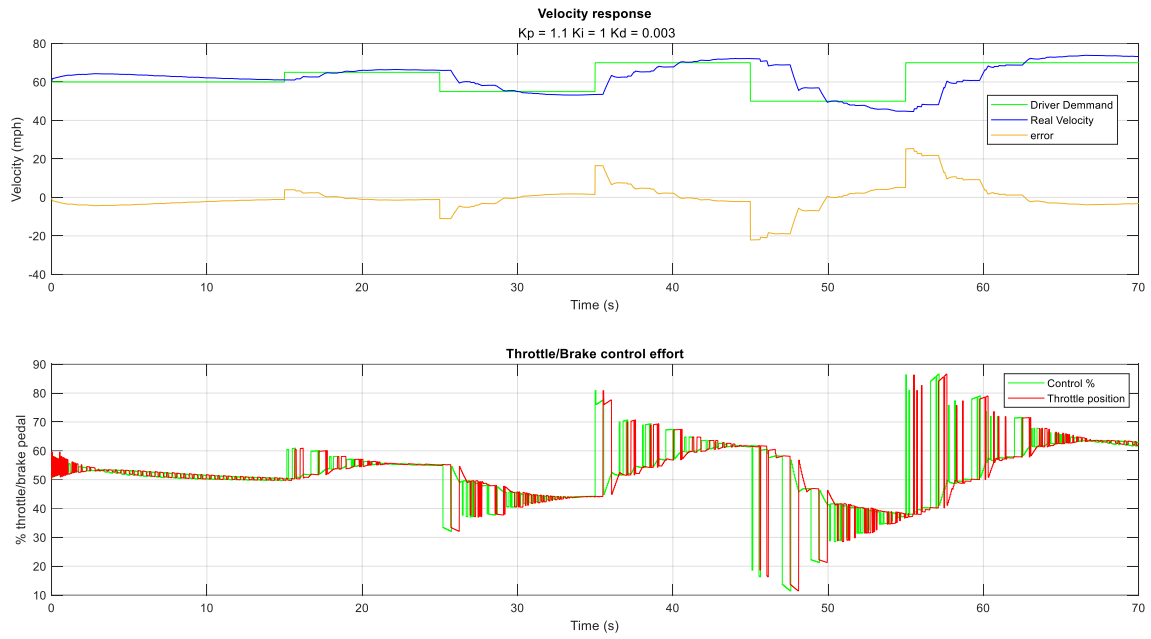
2.5 Communication Network Delays

Within systems, there is often delays within communication networks due to the vehicle communication system. The protocols used within a vehicle have limitations, for example, a vehicle fitted with CAN bus will transfer and receive signals based on priority, if systems are transferring higher priority signals/data, lower priority systems will be affected by a queuing delay (Voss, 2008). The controller itself (PID) also must process and compute the information and determine a suitable output value, the more complex the controller, the longer the processing time. According to Haas, et al., (2024), for automotive applications the critical limit of effective communication delay is approximately 100 milliseconds. The results in graph 12 are representative of a 0.1 second delay, the system still demonstrates good predictability, convergence to zero steady state error and effective controller output response time.

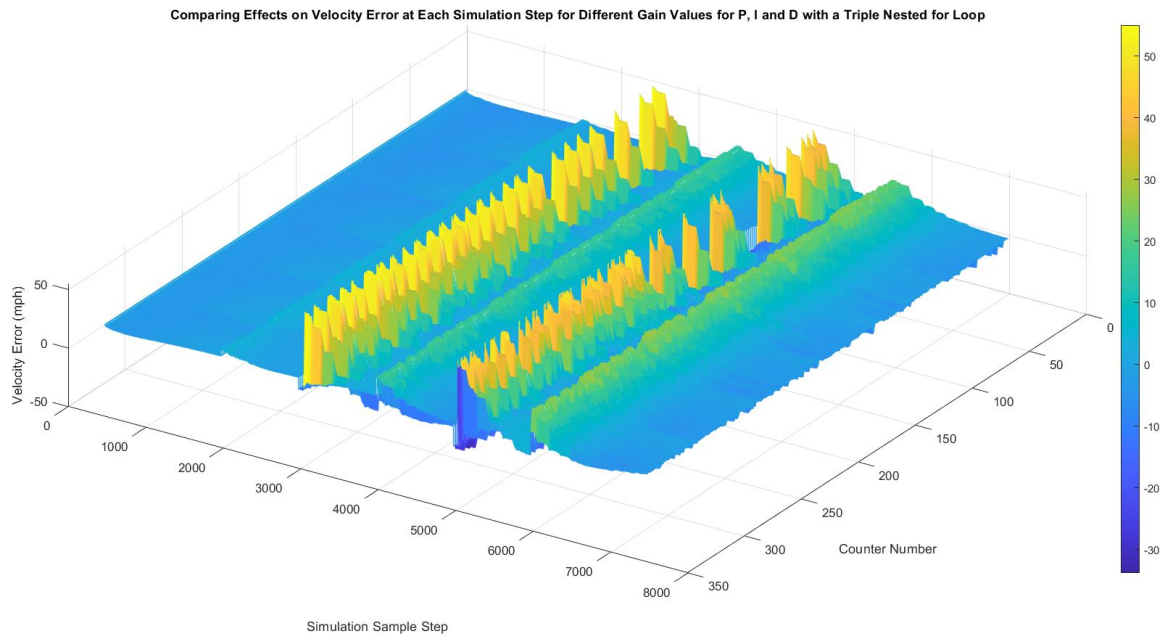


Graph 12: Acceleration/Deceleration Limited Cruise Controller PID Outputs with Communication Delays of 0.1 Seconds.

However, the limit of this controller is demonstrated at 0.5 seconds where the system has significant delays in response time (graph 13), larger step gaps and overshoot. The steady state error has also increased from -1.2784 to -3.51, showing double the increase. Due to this delay, the controller increases output corrections to eliminate error from previous control output delays. Attempts to reduce the significance of these errors had little to no effect when trialled in a triple nested for loop with gain parameter adjustments. This is illustrated in graph 14 by the high ranges in velocity error. Gain parameters were set as: $P = 0.8-1.5$, $I = 0.5-1.5$ and $D = 0-0.05$.



Graph 13: Acceleration/Deacceleration Limited Cruise Controller PID Outputs with Communication Delays of 0.5 Seconds.



Graph 14: Comparing Effects on Velocity Error for Different Gain Values with Acceleration and Deacceleration Limit when Communication Network Delays Reach 0.5 Seconds.

2.6 Concluding Comments

The PID cruise control system demonstrates high responsiveness to driver demands with minimal steady state error. However, overshoot during driver speed adjustment demands raise safety concerns and could benefit from critical to overdamped responses to mitigate underdamping. Adding an acceleration/deceleration limiter improves passenger comfort but increases error and intensifies overshooting. Lower communication network delays, bit rates, and sample rates further impair system predictability and safety. Refinement of the acceleration/deacceleration coding aspect, implementation of a model predictive controller or robust PID with delay compensation is suggested to improve response and decrease steady state error/trend to zero error (Camacho & Bordons, 2012).

Reference List

Annamalai, R. et al., 2013. *Development of Vehicle Yaw Stability Controller*, India: SAE Technical Paper 2013-26-0086.

Aripin, M. et al., 2014. A Review of Active Yaw Control System for Vehicle Handling and Stability Enhancement. *International Journal of Vehicular Technology*.

Bosch, 2025. *GRADE-X Vehicle Diagnostics*. [Online]
Available at: <https://www.bosch-mobility.com/en/solutions/software-and-services/grade-x-vehicle-diagnostics/>
[Accessed 24 March 2025].

Bridgelall, R., 2015. Inertial Sensor Sample Rate Selection for Ride Quality Measures. *Journal of Infrastructure Systems*, 21(2).

Camacho, E. & Bordons, C., 2012. *Model Predictive Control in the Process Industry*. Germany: Springer London.

Eaton, n.d.. *EGerodisc Differential*. [Online]
Available at: <https://videos.eaton.com/detail/video/5173824355001/egerodisc-differential>
[Accessed 22 March 2025].

Guidara, N., 2016. *Automotive FlexRay Bus*. [Online]
Available at: <https://www.linkedin.com/pulse/automotive-flexray-bus-nizar-guidara>
[Accessed 24 March 2025].

Haas, O. et al., 2024. Network Latency in Teleoperation of Connected and Autonomous Vehicles: A Review of Trends, Challenges, and Mitigation Strategies. *Sensors*, 24(12).

Happee, R., Irmak, T., Winkel, K. & Shyrokau, B., 2023. Standards for Passenger Comfort in Automated Vehicles: Acceleration and Jerk. *Applied Ergonomics*, Volume 106.

Henderson, H., 2009. *Encyclopedia of Computer Science and Technology*. United States: Facts On File.

Hussien, S., Jaafar, H., Ghazali, R. & Razif, N., 2015. The Effects of Auto-Tuned Method in PID and PD Control Scheme for Gantry Crane System. *International Journal of Soft Computing and Engineering (IJSCE)*, 4(6), pp. 121-125.

Jeep, 2016. *How Quadra-Drive-II ELSD diff works .. animated video..*. [Online]
Available at: <https://www.jeepcommander.com/threads/how-quadra-drive-ii-elsd-diff-works->

[animated-video.14195/](#)

[Accessed 23 March 2025].

Li, s. et al., 2020. Vehicle Yaw Stability Control at the Handling Limits Based on Model Predictive Control. *International Journal of Automotive Technology*, Volume 21, pp. 361-370.

Li, X., Song, X. & Chan, C., 2014. Reliable Vehicle Sideslip Angle Fusion Estimation using Low-Cost Sensors. *Measurement*, Volume 51, pp. 241-258.

Macatee, S., 2010. *Digital Audio at Last*. [Online]

Available at: <https://www.prosoundtraining.com/2010/03/21/digital-audio-at-last/>

[Accessed 27 March 2025].

Mathworks, 2023. *Documentation: Floating-Point Numbers*. [Online]

Available at: https://www.mathworks.com/help/releases/R2021a/matlab/matlab_prog/floating-point-numbers.html

[Accessed 27 March 2025].

Mathworks, 2025. *MATLAB Help Centre: bin2dec*. [Online]

Available at: <https://uk.mathworks.com/help/matlab/ref/bin2dec.html>

[Accessed 3 March 2025].

McCord, K., 2011. *Automotive Diagnostic Systems Understanding OBD I and OBD II*. United States: CarTech, Incorporated.

Palazzolo, J., 2009. *High-Performance Differentials, Axles & Drivelines*. United Kingdom: CarTech.

Patronis, E., Davis, D. & Brown, P., 2013. *Sound System Engineering 4e*. United Kingdom: Taylor & Francis.

Piyabongkarn, D., Lew, J., Grogg, J. & Kyle, R., 2006. *Stability-Enhanced Traction and Yaw Control using Electronic Limited Slip Differential*, Detroit: SAE Technical Paper 2006-01-1016.

Rajamani, R., 2006. *Vehicle Dynamics and Control*. United Kingdom: Springer.

Schildbach, G., 2018. On the Application of ISO 26262 in Control Design for Automated Vehicles. *Electronic Proceedings in Theoretical Computer Science*, pp. 74-82.

VI-GRADE, 2020. *VI-CarRealTime 20.1 Documentation*, Germany: VI-grade GmbH.

Visioli, A., 2006. *Practical PID Control*. Germany: Springer London.

Voss, W., 2008. *A Comprehensible Guide to Controller Area Network*. United Kingdom: Copperhill Technologies Corporation.