Lateral Control of Autonomous Vehicles

Lab report - I

Muhammad Ali (MARS)

Melaku Mesihu (MARS)

Lab objectives:

The objectives of this lab are to understand how to generate a reference trajectory for an autonomous car, propose a method, and simulate it while utilizing both kinematic and dynamic car models for analysis and control purposes. Additionally, designing and analysing Linear Time-Invariant (LTI) and Linear Parameter-Varying (LPV) controllers in state feedback form to achieve robust lateral control of autonomous vehicles.

1. Vehicle Modelling

1.1.Kinematic Vehicle Model

The kinematic model facilitates the analysis of a car's trajectory relative to the Inertial frame (X, Y) by mapping its speeds onto the Body frame (x, y). Illustrated in Figure 1, this model represents the vehicle's motion with respect to a fixed coordinate system, specifically the Earth frame. It is geometrically derived under the assumption that no forces are acting on the vehicle, treating it as a point object located at its center of mass. The resulting kinematic model, expressed mathematically, connects the global position (X, Y) and orientation in radians to the longitudinal velocity (v_x) and yaw rate (ψ).

The kinematic model equations:

$$\begin{cases} \dot{X} = v_x \cos(\psi) - v_y \sin(\psi) \\ \dot{Y} = v_x \sin(\psi) + v_y \cos(\psi) \end{cases} \quad (1)$$

where X, Y and ψ specifies the global position in meters (m) and the orientation in radians (rad) respectively. v_x and $\dot{\psi}$ represents the longitudinal velocity in m/s and the yaw rate in Rad/s, respectively. The velocity v_y represents the lateral velocity of the car that we will consider equal to zero for car path planning. β is the Sideslip angle.

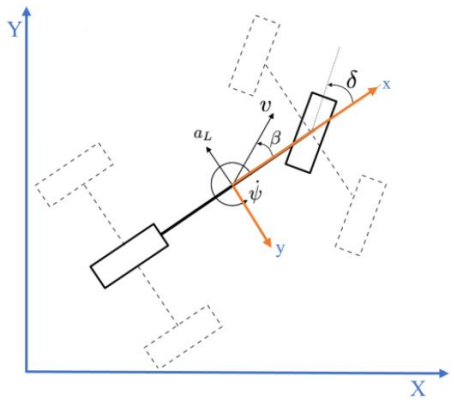

Figure 1: A vehicle into Inertial and Body Frames

By assuming v_y = 0, so equation (1) can be simplified to:

$$\begin{cases} \dot{X} = v_x \cos(\psi) \\ \dot{Y} = v_x \sin(\psi) \end{cases} \tag{2}$$

1.2.Dynamic Vehicle Model

The 2-DOF linear bicycle model is fundamental in control synthesis, capturing essential aspects of lateral vehicle dynamics. Despite its simplicity, it provides a fast and efficient method for analysing vehicle behaviour. Engineers frequently use the bicycle model in the initial phases of vehicle design to assess handling and stability before committing to more complex simulations or physical prototypes. In autonomous vehicle systems, where accurate and rapid predictions of vehicle movements are critical, the bicycle model is widely utilized for real-time control and trajectory planning. Acting as a simplified representation of the full vehicle model, it serves as the primary reference for controlling and stabilizing lateral motion. This model consolidates the two wheels on each axle into a single point at the axle's center, as illustrated in Figure 2, and describes the vehicle's yaw rate and side-slip angle at its center of gravity through the following relationship:

$$\begin{cases} \mathrm{m}a_y = \mathrm{m}(\dot{v}_y + \dot{v}_x\dot{\psi}) = F_{yf} + F_{yr} \\ I_z\ddot{\psi} = L_f F_{yf} - L_r F_{yr} \end{cases} \tag{3}$$

where ay is the lateral acceleration, m the mass and Iz the car inertia.

Figure 2 shows the resulting vehicle model which is expressed by the parameters where vx and v_y are the longitudinal and lateral velocity accordingly. $\dot{\psi}$ is the yaw rate of the car. α_f, α_r are the tire side-slip angles of the front and rear wheels respectively. β is the side slip angle of the vehicle body. δ is the steering wheel angle. L_f, L_r are the distances of the front

and rear wheel from the center of the gravity of the car and C_f, C_r are the front and rear cornering stiffness. The value for all variables is provided in table1 in the appendix.

Linear functions for the vehicle model:

$$F_{yf} = C_f \alpha_f \tag{4}$$
$$F_{yr} = C_r \alpha_r$$

Using small angles approximations, the side slip angles can be written as:

$$\alpha_f = \delta - \frac{v_y + L_f \dot{\psi}}{v_x} \tag{5}$$

$$\alpha_r = \delta - \frac{v_y + L_r \dot{\psi}}{v_x}$$

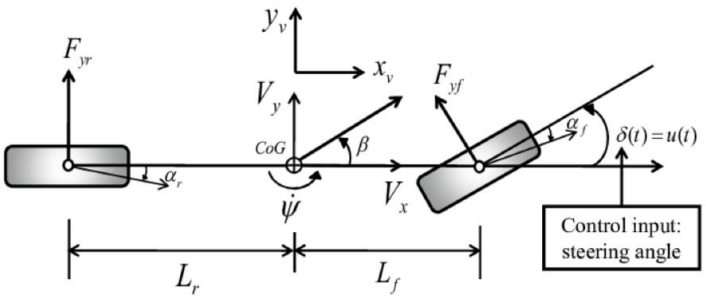

Figure 2: Two-wheeled bicycle model representing

Considering, v_y and $\dot{\psi}$ as state variables and combining the equations (3), (4), (5) the linear vehicle model in state space form is derived:

$$\begin{bmatrix} \dot{v_y} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} -\frac{c_{f+}\, c_r}{m v_x} & -v_x + \frac{CrLr - C_f L_f}{m v_x} \\ \frac{-L_f C_f + L_r C_r}{I_z v_x} & -\frac{L_f^2 C_f + L_r^2 C_r}{I_z v_x} \end{bmatrix} \begin{bmatrix} v_y \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \frac{C_f}{m} \\ \frac{L_f C_f}{I_z} \end{bmatrix} \delta \tag{6}$$

But for a control analysis we use the extended system that contains yaw rate (ψ) as a state variable that we can measure from the sensor and δ (the steering angle is the input "u"). So, the extended system is given as:

$$\begin{bmatrix} \dot{v_y} \\ \ddot{\psi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -\frac{c_{f+}\, c_r}{m v_x} & -v_x + \frac{CrLr - C_f L_f}{m v_x} & 0 \\ \frac{-L_f C_f + L_r C_r}{I_z v_x} & -\frac{L_f^2 C_f + L_r^2 C_r}{I_z v_x} & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_y \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} \frac{C_f}{m} \\ \frac{L_f C_f}{I_z} \\ 0 \end{bmatrix} \delta \tag{7}$$

As seen, the above matrices do depend on the longitudinal speed v_x. So, this model is a Linear Parameter Varying (LPV) model as follows:

$\dot{x}$ = A(v_x)x + B δ

y = Cx

$$A = \begin{bmatrix} -\frac{c_{f+}\, c_r}{m v_x} & -vx + \frac{CrLr - C_f L_f}{m v_x} & 0 \\ \frac{-L_f C_f + L_r C_r}{I_z v_x} & -\frac{L_f^2 C_f + L_r^2 C_r}{I_z v_x} & 0 \\ 0 & 1 & 0 \end{bmatrix}, B = \begin{bmatrix} \frac{C_f}{m} \\ \frac{L_f C_f}{I_z} \\ 0 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, D = 0 \quad (8)$$

2. Reference Generation

The objective of this section is to propose and simulate a car reference trajectory using MATLAB. To illustrate and analyze the example, we will define a trajectory composed of three main elements, utilizing a "Droite-Clothoid-Circle" approach for trajectory generation. A clothoid, in simple terms, is a planar curve where the curvature increases proportionally to the distance along the curve.

Clothoids are particularly important when a vehicle traveling at a constant speed on a straight path transitions into a circular curve. Without a transitional element, this shift causes a sudden change in centrifugal force, which can lead to hazardous side impacts for drivers and potential damage to equipment. To address these risks, a clothoid segment provides a smooth connection between the straight line and the circular arc, eliminating lateral shocks and ensuring a gradual, linear increase in centrifugal force for improved safety. In highway design, this smooth transition is referred to as a "progressive connection," offering a gradual variation in slope between the initial and full curve values. Additionally, the path planning must account for the limitations of contact forces and the steering angle to ensure feasibility and safety.

$$\dot{\psi} = \frac{v_x}{R} = \rho v_x \quad (9)$$

$$F_c = m v_x \dot{\psi} = \frac{m v_x^2}{R}$$

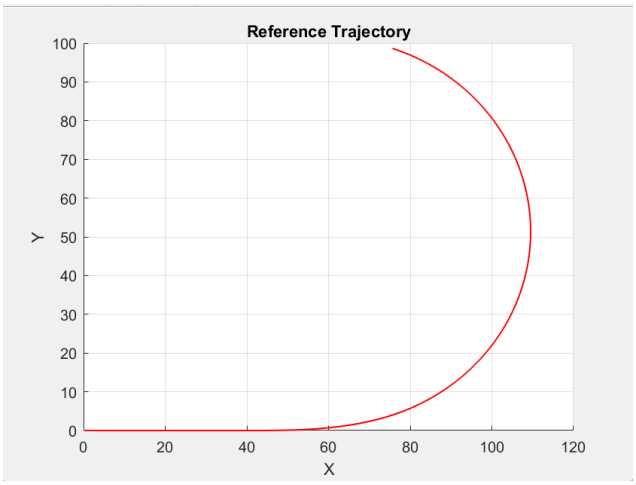

Figure 3: Reference Trajectory using Droite clothoide circle

3. Control Design

This section addresses the design and simulation of various controllers, ranging from Linear Time-Invariant (LTI) to Linear Parameter-Varying (LPV), all implemented in discrete time. The primary control objective is for the car to track a reference trajectory, specifically focusing on tracking a yaw rate reference. Accordingly, the linear bicycle model is selected for control design, with the steering input (δ) as the input and the yaw rate as the output.

Before proceeding with controller design, it is crucial to verify that the system is both controllable and observable. This can be tested using MATLAB commands such as rank (ctrb(A, B)) for controllability and rank(obsv(A, C)) for observability. Based on the MATLAB results, both the controllability and observability matrices have full rank (i.e., a rank of 3), confirming that the system is controllable and observable. With this validation, a control law can now be designed.

3.1. LTI (Linear Time-Invariant)

An LTI system is a type of dynamic system where the relationship between its input and output is both linear and constant over time. Although our system is initially a linear parametric variant system, for the first step, we consider v_x as a constant, treating the system as an LTI system.

LQR Design

The Linear-Quadratic Regulator (LQR) is a state-feedback controller designed to optimize the control of Linear Time-Invariant (LTI) systems. By leveraging the system's current state, LQR minimizes a cost function that balances a weighted sum of the system's state variables and control inputs.

The weighting matrices Q (state cost) and R (control effort cost) are selected based on the relative importance of specific state variables and control efforts for the application. In our case, the Q matrix is configured to heavily penalize high yaw rates, reflecting their critical impact, while assigning lower penalties to the car's lateral velocity and yaw angle. For the R matrix, a value of R=2 is chosen.

To design the discrete-time LQR, first we describe our system using the MATLAB command $c2d$ (SS, Ts), where the sampling time Ts is 0.01 sec and SS is the continuous time state space, then by using this discrete system, we find the gain k using discrete LQR, k=$dlqr$ (ad, bd, Q_lqr, R_lqr). so, the control input (the stirring angle) U=$-kx$.

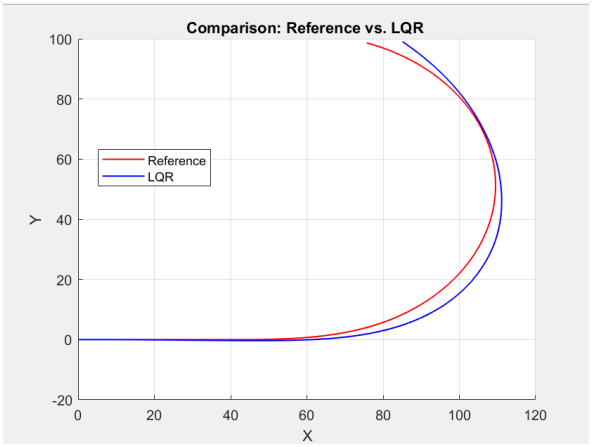

Figure 4: Comparison of Reference and Trajectory followed by LQR controller

3.2. LPV (Linear Parametric Variant)

The modelling of the LPV system involves a dynamic system framework that accounts for parameter variations within its state-space representation, utilizing a polytopic approach.

A polytopic system represents its dynamics as a convex combination of multiple linear subsystems, each corresponding to a specific region in the parameter space.

The discrete-time polytopic system can be expressed as follows:

$$X_{k+1} = (\sum_{i=1}^{N} \alpha_k(i) A_i) X_k + B_{u_k} + E w_k \quad (10)$$

where X_k is the state vector, at instant k, and X_{k+1} its successor, u_k and w_k are the control input and the process disturbance input, respectively. Matrices $\boldsymbol{A_i}$ with i = {1, 2,…,N}, **B** and **E** are the state-space matrices with appropriated dimensions.

The convex set for scheduling parameters in a polytopic LPV model is determined by factors such as vehicle speed, steering angle, road curvature, or road slope, depending on the system dynamics and the parameters influencing its behaviour.

In this case, the system has two varying parameters, $\rho_k \in [\underline{\rho_k}, \overline{\rho_k}]$, specifically $\rho_1 = v_x$ and $\rho_2 = \frac{1}{v_x}$, which appear in affine form within the state-space matrices. Assuming the longitudinal speed is limited to the range of 5 m/s to 20 m/s, the convex set representing the possible evolution of the scheduling parameters is ρ_1 = [5, 20] and ρ_2 = [0.05, 0.2], which are ordered in the min max order sequence.

$$A_{\rho_k} = \left(\sum_{i=1}^{N} \alpha_k(i) A_i \right)$$

The convex optimization problem is formulated as a polytope with four corners (since we have 2 varying parameters), as shown in Figure 5.

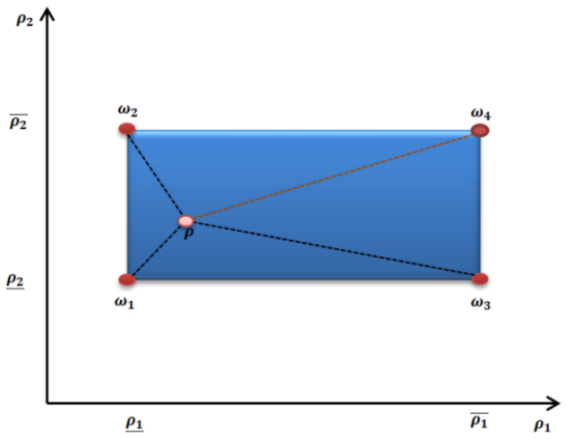

Figure 5: Polytopic modelling approach

The interpolation between the four vertices allows for the determination of the LPV model by combining the contributions of the four vertices in a weighted manner.

$$\begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \alpha_k(1) \\ \alpha_k(2) \\ \alpha_k(3) \\ \alpha_k(4) \end{bmatrix} = \begin{bmatrix} \rho_k \\ 1 \end{bmatrix} \tag{11}$$

where the column vectors ω_i, i = {1,2,3,4} are the a priori known vertices of the polytopic set Ω_ρ. the column vector α_k can be obtained as a solution of a LP problem allowing the implementation of polytopic controller.

The LPV model is constructed by solving the LTI control problem at each vertex of the convex set. Each vertex represents a specific configuration of the varying parameters. Once the actual desired varying parameter ρ_k is known, the appropriate controller is scheduled.

```
% Four corner A-matrices (continuous):
A1_ct = [ -(Cf+Cr)*rho2min/m, -rho1min + (Cr*Lr - Cf*Lf)*rho2min/m, 0;
        (-Lf*Cf + Lr*Cr)*rho2min/Iz, -(Lf^2*Cf + Lr^2*Cr)*rho2min/Iz, 0;
         0, 1, 0 ];

A2_ct = [ -(Cf+Cr)*rho2max/m, -rho1min + (Cr*Lr - Cf*Lf)*rho2max/m, 0;
        (-Lf*Cf + Lr*Cr)*rho2max/Iz, -(Lf^2*Cf + Lr^2*Cr)*rho2max/Iz, 0;
         0, 1, 0 ];

A3_ct = [ -(Cf+Cr)*rho2min/m, -rho1max + (Cr*Lr - Cf*Lf)*rho2min/m, 0;
        (-Lf*Cf + Lr*Cr)*rho2min/Iz, -(Lf^2*Cf + Lr^2*Cr)*rho2min/Iz, 0;
         0, 1, 0 ];

A4_ct = [ -(Cf+Cr)*rho2max/m, -rho1max + (Cr*Lr - Cf*Lf)*rho2max/m, 0;
        (-Lf*Cf + Lr*Cr)*rho2max/Iz, -(Lf^2*Cf + Lr^2*Cr)*rho2max/Iz, 0;
         0, 1, 0 ];

B_ct_lpv = [ Cf/m; Cf*Lf/Iz; 0];
```

The state trajectory of our system is below the diagonal made from ω_2 to ω_3 (Figure 5) and adding additional forth vertex ω_4 is to make our system too much robust and decreasing the response of the system. So, to mitigate this problem we also tested the

simulation by removing the fourth vertex ω_4. Then the geometry of our polytope will be a triangle instead of rectangular region as shown in Figure 6.

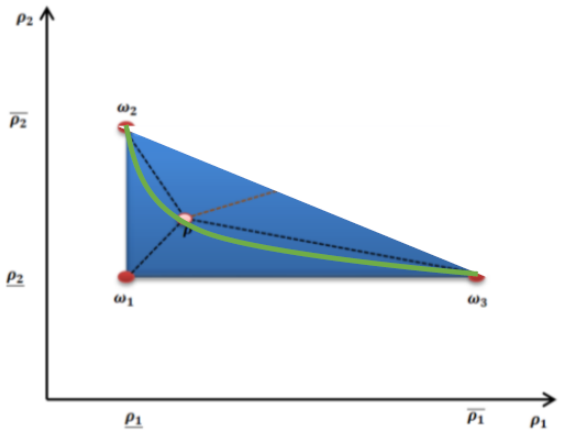

Figure 6: Reduced Vertex Polytopic modelling approach

Polytopic LQR synthesis

LQR problem can be solved by finding a state feedback gain K and a matrix P $>$ 0 such that:

$$min\ trace\ ((C - DK)P(C - DK)^T)$$
$$P > 0 \qquad (12)$$
$$(A - BK)P(A - BK)^T - P + Q \leq 0$$

And equation 12 can be numerically solved by writing the problem as a LMI.

$$(A - BK)PP^{-1}(A - BK)^T - P + Q \leq 0$$

Using Schur complement, and by considering a change of variables, $Y = KP$ the condition becomes:

$$\begin{pmatrix} -P + Q & (AP - BY) \\ * & -P \end{pmatrix} \leq 0 \qquad (13)$$

The state feedback gain will be computed as $K = Y\ P^{-1}$.

Therefore, to implement this control law in discrete LQR LPV we first have to find the discrete form of our system in affine form. So, instated of using MATLAB command "c2d" we use a Tylor approximation of our state space.

$$\dot{x} = Ax + Bu$$

$$\frac{x_{(k+1)} - x_{(k)}}{T_s} = Ax_{(k)} + Bu_{(k)} \Longrightarrow\ x_{(k+1)} = x_{(k)} + T_sAx_{(k)} + T_sBu_{(k)}$$

So $$x_{(k+1)} = (I_{3X3} + T_sA)x_k + T_sBu_{(k)} \qquad (14)$$

Therefore, the discrete state matrix will be $A_d = (I_{3X3} + TsA)$ and $B_d = T_s B$ (Euler method)

```
% Discretize corners:
Ad1 = eye(3) + Ts*A1_ct;
Ad2 = eye(3) + Ts*A2_ct;
Ad3 = eye(3) + Ts*A3_ct;
Ad4 = eye(3) + Ts*A4_ct;
Bd  = Ts * B_ct_lpv;
```

A Discrete Varying State Space is used to simulate the system in Simulink and a varying discrete version of state matrix A and output matrix B is given in to it. The varying state matrix A is dependent on time.

4. Results and Discussions

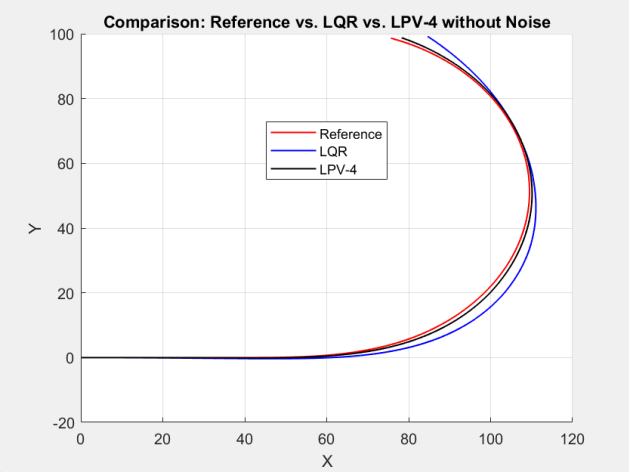

Figure 7: Comparison of Reference and Trajectory followed by LQR and LPV with 4 vertexes controller without Noise

As shown in the previous figure LPV-4 can better handle non-linearities and parameter variations, leading to improved tracking than LQR.

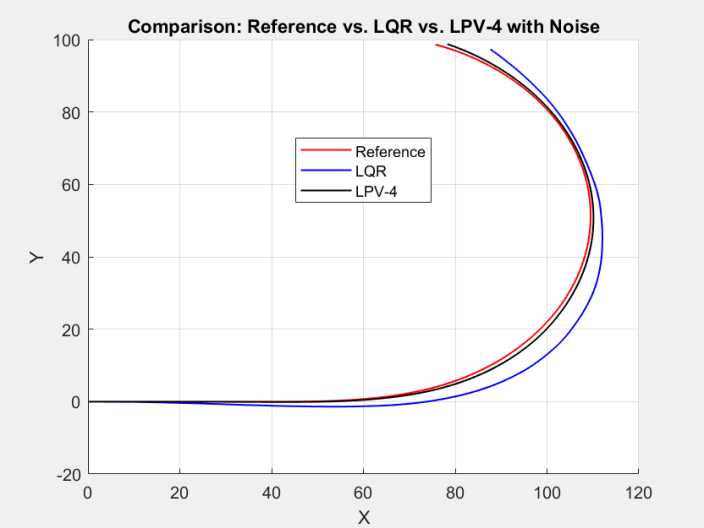

Figure 8: Comparison of Reference and Trajectory followed by LQR and LPV with 4 vertexes controller with Noise

As shown from the previous figure, LPV-4 can handle noise better than LQR, maintaining closer alignment with the desired path. This suggests LPV's adaptability and robustness in noisy environments.

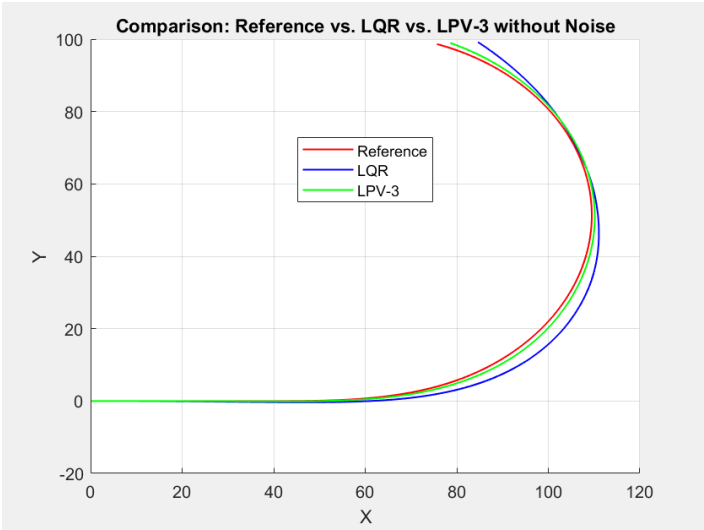

Figure 9: Comparison of Reference and Trajectory followed by LQR and LPV with 3 vertexes controller without Noise

LPV-3 still effectively tracks the reference path, indicating that even with one less vertex, it maintains good performance by adapting to system dynamics. However, it might be slightly less robust than LPV-4 in complex scenarios.

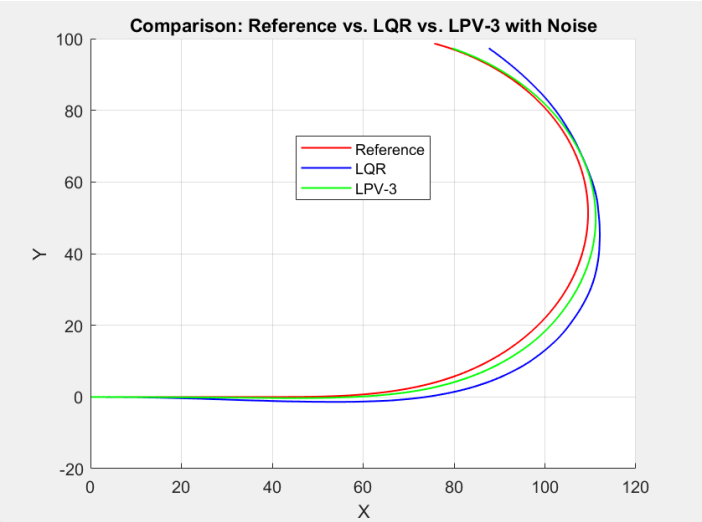

Figure 10: Comparison of Reference and Trajectory followed by LQR and LPV with 3 vertexes controller with Noise

LPV-3 effectively handles noise, maintaining better path tracking than LQR. It shows that even with fewer vertices, LPV-3 is robust in noisy conditions, but less robustness than LPV-4.

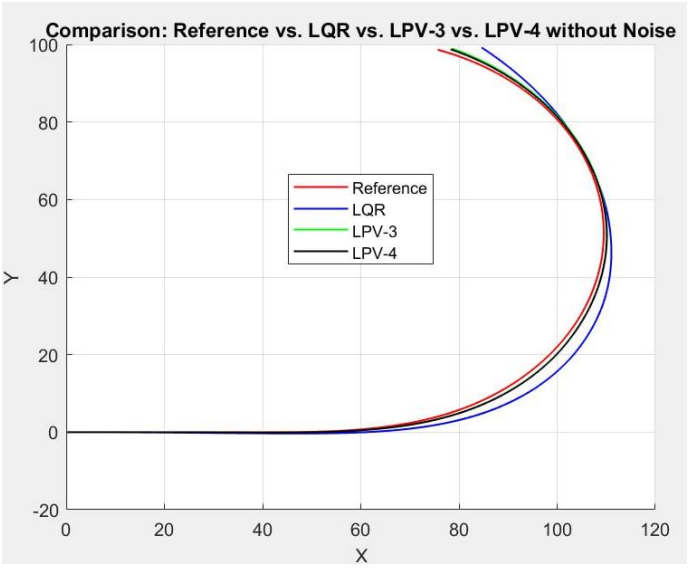

Figure 11 Comparison of Reference and Trajectory followed by LQR and LPV with 3 and 4 vertexes controller without Noise

This plot compares the performance of different controllers in tracking a reference trajectory. The red line represents the desired trajectory. The LQR controller (blue line) shows noticeable deviations, especially in curves, while the LPV controllers (green for LPV-3 and black for LPV-4) demonstrate much better accuracy. Among them, LPV-4

performs the best, closely matching the reference, showcasing its superior ability to handle system dynamics effectively.

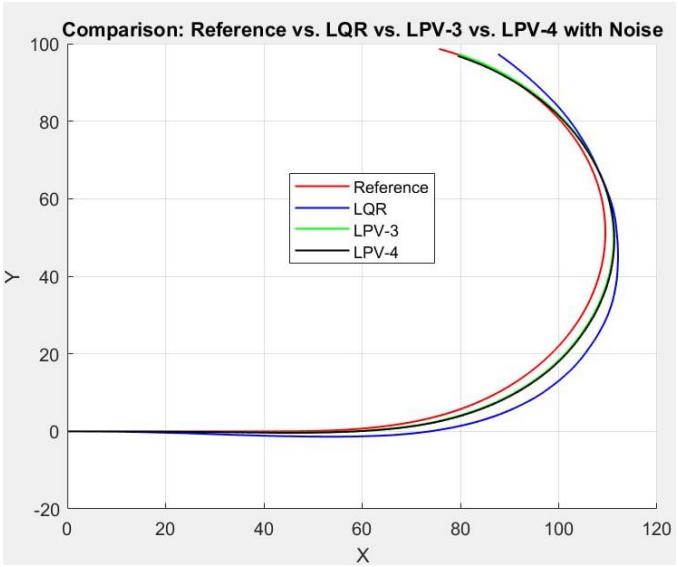

Figure 12 Comparison of Reference and Trajectory followed by LQR and LPV with 3 and 4 vertexes controller with Noise

This plot illustrates the performance of different controllers in tracking a reference trajectory under the influence of noise. The red line represents the desired trajectory. The LQR controller (blue line) shows significant deviations, particularly in curved sections, due to its reduced robustness against noise. The LPV controllers (green for LPV-3 and black for LPV-4) perform better, with LPV-4 demonstrating superior noise handling and closely following the reference trajectory. This highlights the robustness and adaptability of the LPV-4 controller in noisy conditions.

5. Conclusion

This lab report examines the lateral control of autonomous vehicles, emphasizing the design and simulation of controllers for an autonomous car using both kinematic and dynamic models. The primary objectives include generating a reference trajectory and implementing controllers in Linear Time-Invariant (LTI) and Linear Parameter-Varying (LPV) forms.

A clothoid-based approach is employed for reference trajectory generation, ensuring smooth transitions between straight and curved paths while accounting for limits on contact forces and steering angles. In the control design section, both LTI LQR and LPV LQR controllers are developed and analysed. The LPV case leverages a polytopic system, enabling the incorporation of varying parameters within a state-space framework.

Simulation results are provided for both controllers, comparing the system's trajectory to the reference trajectory. The findings indicate that the LPV LQR controller outperforms the LTI LQR controller, particularly under conditions with system noise.

This comprehensive analysis and the accompanying simulations offer valuable insights into the lateral control of autonomous vehicles, highlighting the superior adaptability and performance of LPV control strategies in dynamic environments.

Appendix

Parameter	Symbol	Value	Units
Vehicle mass	m	1500	kg
Front cornering stiffness	C_f	30000	N/rad
Rear cornering stiffness	C_r	40000	N/rad
Yaw inertia	I_z	2734	$kg\,m^2$
Distance from front axle to COG	L_f	1.3	m
Distance from rear axle to COG	L_r	1.4	m

Table 1: Vehicle parameters

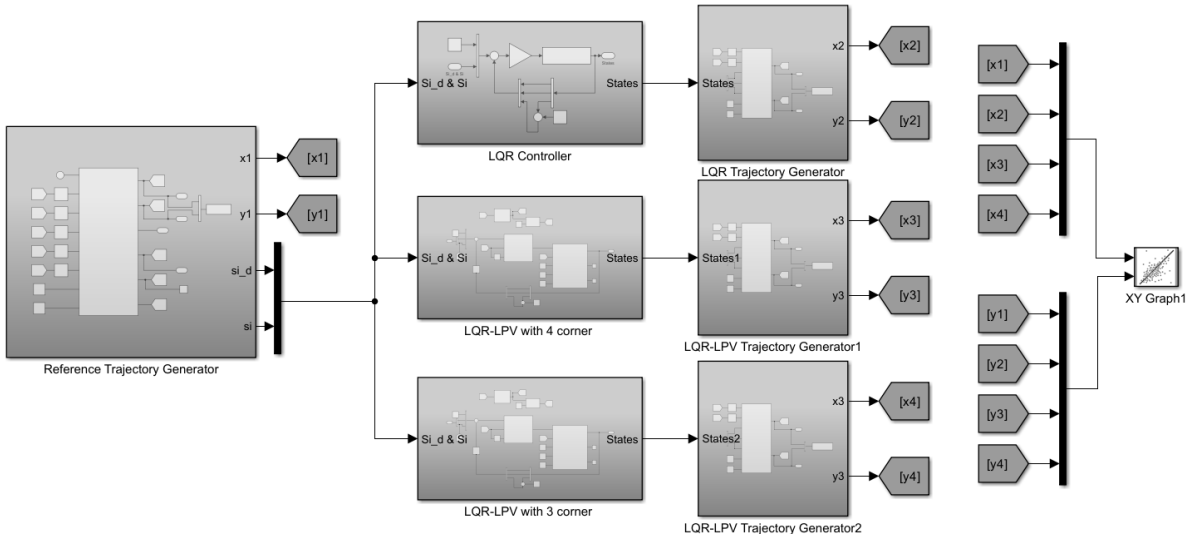

Figure 13: Simulink model of the system