**Lab Title**: Adaptive Cruise Control (ACC)

**Objective/Aim:**

To implement an Adaptive Cruise Control (ACC) system that automatically adjusts the vehicle speed based on the relative distance to the vehicle ahead, maintaining a safe following distance while ensuring comfort and fuel efficiency.

**Equipment/Software Used:**

* Python 3.12
* MATLAB/Simulink or any vehicle simulation software
* Sensors (Radar/LIDAR) for distance measurement (simulated data)
* Vehicle dynamics model for speed and distance control

**Required Libraries:**

* NumPy for numerical computations
* Matplotlib for data visualization
* Control systems library (in Python or MATLAB) for PID controller implementation
* Sensor simulation libraries like PySimu or MATLAB equivalents for simulating radar/LIDAR data

**Theory:**

Adaptive Cruise Control (ACC) is an advanced driver assistance system (ADAS) designed to automate speed control and maintain a safe distance between a vehicle and the one ahead. Unlike conventional cruise control systems, which maintain a set speed regardless of traffic conditions, ACC dynamically adjusts the speed of the vehicle to adapt to the behaviour of the lead vehicle.

At the core of ACC is a feedback control system that uses sensors like radar or LIDAR to measure the distance and relative speed between vehicles. These sensors continuously send data to the ACC control unit, which processes this information and adjusts the vehicle’s speed by applying throttle or brakes as necessary.

Working Principles:

The ACC system can be broken down into three primary components:

* Sensors: ACC relies on radar or LIDAR sensors placed at the front of the vehicle to measure the distance to the vehicle ahead. These sensors are capable of detecting objects, calculating their speed, and determining the distance.
* Control System: The most common control algorithm used in ACC is the Proportional-Integral-Derivative (PID) controller. The PID controller computes the error between the desired and actual distance and adjusts the vehicle's acceleration or deceleration accordingly. The Proportional (P) component deals with the present error, the Integral (I) component sums past errors, and the Derivative (D) component predicts future errors. The combination of these terms allows for smooth control over the vehicle's speed.

Traffic Scenarios:

ACC systems are especially useful in highway driving and stop-and-go traffic. In highway driving, ACC ensures that the vehicle can follow the lead vehicle at a safe distance, smoothly adjusting speed as necessary. In stop-and-go traffic, the system can bring the vehicle to a complete stop and resume motion when traffic starts moving again, thus reducing driver fatigue.

Challenges and Considerations:

* Response Time: The accuracy and efficiency of the ACC depend on the response time of the sensors and the controller. Lag in sensor readings or delays in processing can cause safety issues, especially in dynamic traffic conditions.
* Complex Traffic Scenarios: ACC must also handle edge cases, such as sudden braking by the lead vehicle, lane changes, or stop-and-go traffic, without causing discomfort or compromising safety.
* Environmental Factors: External factors like weather conditions (e.g., rain, snow) can affect the performance of radar or LIDAR sensors, requiring additional considerations for robust implementation.

**Procedure:**

1. Define the vehicle’s kinematics using a simplified model such as the point-mass or bicycle model. This model will simulate the motion of the vehicle, taking inputs from the controller to adjust speed based on throttle or braking actions.
2. Simulate radar or LIDAR sensor readings to detect the distance and speed of the vehicle ahead. For simplicity, you can create a virtual environment where the lead vehicle's position and speed are generated. These sensor values will be the input to the ACC controller.
3. Create a PID controller to calculate the difference (error) between the desired distance and the actual distance from the vehicle ahead. Tune the controller's parameters (Kp, Ki, and Kd) to provide smooth and stable speed adjustments.
4. Implement a variety of traffic scenarios where the lead vehicle’s speed changes dynamically. This could include accelerating, decelerating, or sudden stops. Ensure that the ACC system responds effectively to these changes.
5. Visualize the performance of the ACC system by plotting vehicle speed, distance from the lead vehicle, and control signals (acceleration or braking). This will help assess the system’s response to various traffic situations. Analyse whether the ACC system maintains the desired distance effectively and how quickly it reacts to changes.

**Code:**

Below is a Python code example for implementing a PID controller and visualizing the ACC system’s behaviour:



**Output:**

1. **Distance Graph**: The first graph shows the distance between the controlled vehicle and the lead vehicle over time. The system maintains the desired safe distance (e.g., 30 meters) with some initial fluctuations, which stabilize as the PID controller adjusts the speed.
2. **Speed Graph**: The second graph compares the speed of the controlled vehicle and the lead vehicle. The controlled vehicle speed smoothly follows the lead vehicle's speed changes, adjusting throttle and braking as needed to maintain the desired distance.

**Results and Observation:**

1. **Distance Maintenance**: The ACC system successfully maintains a safe following distance from the lead vehicle. Initially, there are small fluctuations as the PID controller corrects the error, but the system stabilizes and keeps the vehicle at the target distance. This illustrates the effectiveness of the PID controller in dynamic traffic conditions.
2. **Smooth Speed Transitions**: The system adjusts the vehicle speed smoothly, responding to changes in the lead vehicle's speed. This is important for maintaining comfort and safety. If the lead vehicle suddenly decelerates, the ACC system reacts by applying braking, but in a controlled manner to avoid abrupt stops.
3. **Control System Efficiency**: The results show that tuning the PID controller parameters (Kp, Ki, and Kd) is crucial for system performance. Higher values of Kp lead to faster responses but may cause overshooting, while increasing Ki helps eliminate steady-state error. The derivative term (Kd) helps predict future errors and smooths out the control action.
4. **Challenges in Stop-and-Go Traffic**: While the ACC performs well in highway scenarios, stop-and-go traffic presents challenges. If the lead vehicle stops suddenly, the ACC system must be tuned to apply adequate braking force to avoid collisions. Further refinement of the controller would be necessary to handle such scenarios effectively without compromising safety or passenger comfort.