Aim

To develop a control system for autonomous lane changing of a vehicle using Model Predictive Control (MPC) in MATLAB.

Objective

- Implement a Model Predictive Control (MPC) algorithm for autonomous lane changing.
- Simulate the vehicle's response to lane-changing maneuvers.
- Validate the control system's performance through trajectory tracking and input analysis.

Summary

This MATLAB project focuses on designing a control system for an autonomous vehicle to perform lane changes effectively. It uses Model Predictive Control (MPC) to generate and follow a reference trajectory. The simulation involves generating reference signals, setting up the vehicle's initial state, and iteratively solving an optimization problem to determine the control inputs. The results are visualized through plots of the vehicle's trajectory, state variables, and control inputs.

Tools and Libraries Used

- MATLAB
- Optimization Toolbox (for quadprog)
- Control System Toolbox (for state-space and trajectory analysis)

Procedure

1. **Load Constants**: Initialize constants and parameters such as sampling time, controlled states, and trajectory information.

CODE:

```
constants=initial_constants();

Ts=constants('Ts');

controlled_states=constants('controlled_states');

hz = constants('hz');

x_dot=constants('x_dot');

trajectory=constants('trajectory');
```

2. **Generate Reference Signals**: Create reference signals for the trajectory including desired yaw angles and positions.

CODE:

```
t = 0:Ts:7;
```

[psi_ref,X_ref,Y_ref]=trajectory_generator(t);

3. **Setup Initial State**: Define the initial state of the vehicle and initialize state and input arrays.

CODE:

```
y_dot=0;

psi=psi_ref(1,2);

psi_dot=0;

Y=Y_ref(1,2);

states=[y_dot,psi,psi_dot,Y];

statesTotal=zeros(length(t),length(states));

statesTotal(1,:)=states;
```

4. **Controller Initialization**: Initialize the Model Predictive Control (MPC) parameters and matrices for optimization.

CODE:

```
[Ad, Bd, Cd, Dd]=LPV_cont_discrete(states);

[Hdb,Fdbt,Cdb,Adc]=MPC_simplification(Ad,Bd,Cd,Dd,hz);
```

5. **Simulation Loop**: Iteratively solve the optimization problem to compute control inputs and simulate the vehicle's response.

CODE:

```
for i =1:sim_length-1
   % Generate reference vector and update matrices if necessary
   % Solve optimization problem using `quadprog`
   % Update states and inputs
end
```

6. **Plot Results**: Visualize the simulation results including trajectory, state variables, and control inputs.

CODE:

```
% Plot trajectory
% Plot position states and yaw
% Plot inputs
```

Highlights

- **Model Predictive Control (MPC)**: Utilized for predicting and optimizing control inputs over a moving time horizon.
- **Optimization with quadprog**: Solves the quadratic programming problem to determine optimal control actions.
- **Adaptive Horizon**: Adjusts the prediction horizon dynamically based on the simulation progress to improve efficiency.

Conclusion

The MATLAB simulation successfully demonstrates the application of Model Predictive Control (MPC) for autonomous lane changing. The vehicle accurately follows the desired trajectory, and the control inputs are within feasible limits. The MPC approach effectively manages the trade-off between control performance and computational complexity, making it suitable for real-time autonomous driving applications. Future work could involve extending the control system to handle more complex driving scenarios and integrating it with real-world vehicle dynamics.