Developing a Motion Planner for Highway Lane Change Maneuvers

Seo-Wook Park
Motion planner for Highway Lane Change Maneuver

- Automated lane change maneuver (LCM) system for highway driving scenario
- Generates an optimal trajectory in Frenet space
- Implement driving maneuver behavior depending on surrounding traffic conditions
- Collision checking using dynamic capsule-based objects
Motion planner for Highway Lane Change Maneuver

Learn fundamentals
- Motivation
- Schematic of Motion Planner
- Three MATLAB functions
  - referencePathFrenet
  - trajectoryGeneratorFrenet
  - dynamicCapsuleList

Implement motion planner and controller
- Learn through reference example
- Architecture of highway lane change planer
- Closed-loop controller

Simulate test bench
- Scenarios in straight and curved roads
- Scenarios imported from HERE HD map
Lane Change Maneuver (LCM) system – example

- Feedback
  - Lack of configurability and flexibility
  - How to customize?
Modular architecture to enable customized design

- Feedback
  - Lack of configurability and flexibility
  - How to customize?

- Modular architecture
  (Grey box means purely custom implementation)
Modular architecture to enable customized design

Initial version of LCM

Major upgrade for planner (MATLAB version)

- Modular architecture
  (Grey box means purely custom implementation)
Evolution of examples for Lane Change Maneuver (LCM) system

**Initial version of LCM**
- Highway Lane Change
  - Simulate an automated lane change maneuver system for highway driving scenario.
  - Open Script

**Major upgrade for planner (MATLAB version)**
- Highway Trajectory Planning Using Frenet Reference Path
  - Demonstrates how to plan a local trajectory in a highway driving scenario. This example uses a reference path and dynamic list of
  - Open Live Script

**Motion planner (Simulink Model)**
- Generate Code for Highway Lane Change Planner
  - Design, test, and generate C++ code for a lane change planner for highway driving. This example closely follows the Highway

**Closed-loop system (Simulink Model)**
- Highway Lane Change
  - Design and test the planner and controller components of a lane change maneuver system designed for highway driving.

Modular architecture
Incrementally learn about lane change planning and controls

**referencePathFrenet**
- Reference Path Generation
- Coordinate Transform

**trajectoryGeneratorFrenet**
- Trajectory Generation

**dynamicCapsuleList**
- Collision Checking
- Feasibility Checking
- Cost Evaluation

Generate Code for Highway Lane Change Planner
Design, test, and generate C++ code for a lane change planner for highway driving. This example closely follows the Highway

Motion planner (Simulink Model)

Closed-loop system (Simulink Model)
Schematic of Motion Planner for Lane Change Maneuver

- **dynamicCapsuleList**
- **checkCollision**
- **referencePathFrenet**
- **frenet2global**
- **Motion Prediction**
- **trajectoryGeneratorFrenet**
- **Terminal State Sampler**
- **Feasibility Check**
- **Cost Evaluation**

**Key Components:**
- Map
- Waypoints
- Reference Path Frenet
- Ego Global State
- MIO Global States
- Ego Frenet State
- MIO Frenet States
- Terminal States
- Map Info

**Tasks:**
- Find MIOs
- Global to Frenet
- Frenet to Global
- Predicated Trajectories
- Target Trajectories
- Candidate Ego Trajectories
- Optimal Trajectory
Advantage of trajectory planning in Frenet coordinate

- Frenet system represents an object and its trajectory with respect to the reference path (road center or lane center).

- This approach dramatically simplifies the trajectory planning task when a car is traveling on a curved road.

\[ s \ L = \frac{1}{\kappa}(0, \theta) \]

Global States: \([x \ y \ \theta \ \kappa \ v \ a]\)

Frenet States: \([s \ \delta s \ \delta^2 s \ L \ \delta L \ \delta^2 L]\)

- \(s\): Arc length
- \(L\): Perpendicular deviation from the direction of the reference path
Schematic of Motion Planner for Lane Change Maneuver

- Destination
- Map
- Waypoints
- referencePathFrenet
- dynamicCapsuleList
- checkCollision
- candidate ego trajectories
- predicted target trajectories
- Feasibility Check
- Motion Prediction
- trajectoryGeneratorFrenet
- terminalStates
- Terminal State Sampler
- MapInfo
- Frenet Space
**referencePathFrenet**

- Fits a smooth, piecewise, continuous curve to a set of waypoints given as \([x \ y]\) or \([x \ y \ \theta]\)

- Convert trajectories between global and Frenet coordinate systems (frenet2global, global2frenet)

**Path Points:** \([x \ y \ \theta \ \kappa \ \delta \kappa \ s]\)
  - \(\kappa\): Curvature
  - \(\delta \kappa\): Derivative of curvature
  - \(s\): Arc length

**Global States:** \([x \ y \ \theta \ \kappa \ v \ a]\)

**Frenet States:** \([s \ \delta s \ \delta^2 s \ L \ \delta L \ \delta^2 L]\)
  - \(s\): Arc length
  - \(L\): Perpendicular deviation from the direction of the reference path
Schematic of Motion Planner for Lane Change Maneuver

- **Destination**
- **Map**
- **Waypoints**
- **dynamicCapsuleList**
- **checkCollision**
- **candidate ego trajectories**
- **Feasibility Check**
- **Cost Evaluation**
- **Motion Prediction**
- **trajectoryGeneratorFrenet**
- **terminalStates**
- **frenet2global**
- **Find MIOs**
- **global2frenet**
- **mioGlobalStates**
- **egoGlobalState**
- **terminalStates**
- **egoFrenetState**
- **mioFrenetStates**
- **egoGlobalState**
- **terminalStates**
- **MapInfo**

**Frenet Space**

**Global to Frenet Conversion**

**Frenet to Global Conversion**

**Optimal Trajectory**
Generates trajectory by solving 4th or 5th order polynomials that satisfy boundary conditions in Frenet space relative to a given reference path

```
refPath = referencePathFrenet(waypoints);
connector = trajectoryGeneratorFrenet(refPath);
[frenetTraj,globalTraj] = ...
connect(connector,initState,termState,timeSpan);
```

initState = [0 0 0 0 0 0];  % [S ds ddS L dL ddL]
termState = [50 0 0 3 0 0];  % [S ds ddS L dL ddL]

![Graph showing trajectory generation](image_url)
Schematic of Motion Planner for Lane Change Maneuver

- **Destination**
- **Waypoints**
- **Map**
- **dynamicCapsuleList**
- **checkCollision**
- **referencePathFrenet**
- **MIOs**
- **global2frenet**
- **egoGlobalState**
- **mioGlobalStates**
- **frenet2global**
- **Feasibility Check**
- **Cost Evaluation**
- **Motion Prediction**
- **trajectoryGeneratorFrenet**
- **terminalStates**
- **Terminal State Sampler**
- **MapInfo**
- **Frenet Space**
dynamicCapsuleList

- Manages lists of capsule-based objects for collision checking

Why Capsules?
- Automatically build a buffer on front/rear of rectangular objects
- Avoid inflating the sides of rectangular objects
- Computationally efficient for collision checking
- Check for collisions between ego and obstacles using `dynamicCapsuleList`

```matlab
isColliding = checkCollision(capList)
```
Motion planner for Highway Lane Change Maneuver

Learn fundamentals
- Motivation
- Schematic of Motion Planner
- Three MATLAB functions
  - referencePathFrenet
  - trajectoryGeneratorFrenet
  - dynamicCapsuleList

Implement motion planner and controller
- Learn through reference example
- Architecture of highway lane change planer
- Closed-loop controller

Simulate test bench
- Scenarios in straight and curved roads
- Scenarios imported from HERE HD map
Highway lane change planner test bench

- Scene
- Target vehicles
- Map data
- Planner Configuration Parameters
- Highway Lane Change Planner
- Metric assessment
- Visualization
Schematic of Motion Planner for Lane Change Maneuver

- **Map**
  - Waypoints
  - Destination

- **Schematic Diagram**
  - **dynamicCapsuleList**
  - **checkCollision**
  - **Terminal State**
    - **Sampler**
      - **trajectoryGeneratorFrenet**
      - **Cost Evaluation**
      - **Motion Prediction**
        - **frenet2global**
        - **terminalStates**
  - **Feasibility Check**
  - **Find MIOs**
    - **referencePathFrenet**
    - **egoGlobalState**
    - **mioGlobalStates**
  - **global2frenet**
    - **egoFrenetState**
    - **mioFrenetStates**
  - **Frenet Space**

- **Key Components**
  - **ego**
  - **MapInfo**
Highway Lane Change Planner: Terminal State Sampler

- Maneuver Mode
  - Find Preferred Lane
  - Maneuver Mode

- Terminal State Sampler
  - Cruise Control
  - Lead Car Following
  - Lane Change

- Motion Prediction
  - trajectoryGeneratorFrenet
  - Terminal States

- Frenet
  - egoFrenetState
  - mioFrenetStates

- Map Info

- Ego Frenet State
- MIO Frenet States
- Map Info
Preferred lane and maneuver mode

- **Cruise Control Mode**
  - $NaN|V_{set}|0|L_{ego}|0|0$

- **Lead Car Following Mode**
  - $NaN|V_{lead}|0|L_{ego}|0|0$

- **Lane Change Mode**
  - $NaN|V_{ego}|0|L_{left}|0|0$
  - $NaN|V_{ego}|0|L_{right}|0|0$

- **Terminal Frenet States**

<table>
<thead>
<tr>
<th>s</th>
<th>$\delta s$</th>
<th>$\delta^2 s$</th>
<th>$L$</th>
<th>$\delta L$</th>
<th>$\delta^2 L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{set}$</td>
<td>0</td>
<td></td>
<td>$L_{ego}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$V_{lead}$</td>
<td>0</td>
<td></td>
<td>$L_{ego}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$V_{ego}$</td>
<td>0</td>
<td></td>
<td>$L_{left}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$V_{ego}$</td>
<td>0</td>
<td></td>
<td>$L_{right}$</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- **Unrestricted longitudinal position**
  - → use 4th-order polynomial

- **Frenet States**: $[s \, \delta s \, \delta^2 s \, L \, \delta L \, \delta^2 L]$
  - $s$ : Arc length
  - $L$ : Perpendicular deviation from the direction of the reference path

**ego**

**unsafe target** ($TTC_{ego-mio} < TTC_{safe}$)

**safe target** ($TTC_{ego-mio} \geq TTC_{safe}$)

where $TTC = Time-to-collision$
Examples for finding optimal trajectory

Candidate trajectories with higher cost

Optimal trajectory

Predicted target trajectories using dynamicCapsuleList

Colliding trajectories

Infeasible trajectory (excessive yaw rate)

Preferred lane = 3

Optimal trajectory with min cost

safe

unsafe

Preferred lane = 3

Optimal trajectory with min cost

safe
Highway lane change: closed-loop system

**Highway Lane Change Test Bench**

- **Sensors and Environment**
- **Vehicle dynamics**
- **Metric assessment**
- **Visualization**

**Components:**
- Highway Lane Change Planner
- Lane Change Controller
- System Time
- EgoActor
- MapInfo
- Target Actors/World
- Planner Params
- Visualization
- Configuration Parameters

**Metrics Assessment:**
- Ego Velocity (m/s)
- Time Gap
- Lateral Jerk
- Longitudinal Jerk

**Vehicle Dynamics:**
- Acceleration
- Lateral Velocity
- Longitudinal Velocity

**Planning and Control:**
- RefPointOnPath
- Acceleration
- Steering Angle
- Longitudinal Velocity

**Visualization:**
- Ego Actor
- Map Info
- Planning Status
- Obstacle Profiles
- HD Video

**System Time:**
- [T]
- [A]

**Planner Configuration Parameters:**
- [B]
Lane Change Planner

Trajectory Generator → Path Analyzer

Optimal trajectory → Reference trajectory (virtual lane center)

Reference trajectory

Reference point $[x_k, y_k, \theta_k, \kappa_k, v_k]$

 yaw angle deviation ($\Delta \theta$)

Lateral deviation

MATLAB EXPO
Lane Change Planner + Controller

Lane Change Planner

- Trajectory Generator
- Path Analyzer

Optimal trajectory
→ Reference trajectory
   (virtual lane center)

Lane Change Controller

- Set velocity
- Path Following Controller
  (Adaptive MPC)

- Acceleration
- Curvature
- Lateral Deviation
- Relative Yaw Angle
- Steering Angle
Motion planner for Highway Lane Change Maneuver

Learn fundamentals
- Motivation
- Schematic of Motion Planner
- Three MATLAB functions
  - `referencePathFrenet`
  - `trajectoryGeneratorFrenet`
  - `dynamicCapsuleList`

Implement motion planner and controller
- Learn through reference example
- Architecture of highway lane change planer
- Closed-loop controller

Simulate test bench
- Scenarios in straight and curved roads
- Scenarios imported from HD map
Simulate highway lane change planner with test scenarios

```matlab
function helperSLHighwayLaneChangeSetup(nvp)

arguments
nvp.scenarioFcnName {mustBeMember(nvp.scenarioFcnName,...
    "scenario_LC_01_SlowMoving";...
    "scenario_LC_02_SlowMovingWithPassingCar";...
    "scenario_LC_03_DisabledCar";...
    "scenario_LC_04_CutInWithBrake";...
    "scenario_LC_05_SingleLaneChange";...
    "scenario_LC_06_DoubleLaneChange";...
    "scenario_LC_07_RightLaneChange";...
    "scenario_LC_08_SlowmovingCar_Curved";...
    "scenario_LC_09_CutInWithBrake_Curved";...
    "scenario_LC_10_SingleLaneChange_Curved";...
    "scenario_LC_11_MergingCar_HighwayEntry";...
    "scenario_LC_12_CutInCar_HighwayEntry";...
    "scenario_LC_13_DisabledCar_Ushape";...
    "scenario_LC_14_DoubleLaneChange_Ushape";...
    "scenario_LC_15_StopnGo_Curved")}

end
```

Scenarios in straight road

Scenarios in curved road

Scenarios imported from HD map
scenario_LC_14_DoubleLaneChange_Ushape
Collision avoidance against a disabled car

- Disabled car detected
- Trajectory for lane change still not collide against the unsafe target, but it’s marginal
- Unsafe target still exists during lane change
- Lane change again immediately after reaching next lane to avoid collision
Closed-loop system simulation: scenario_LC_15_StopnGo_Curved
Recap: Motion planner for Highway Lane Change Maneuver

**Learn fundamentals**
- Motivation
- Schematic of Motion Planner
- Three MATLAB functions
  - referencePathFrenet
  - trajectoryGeneratorFrenet
  - dynamicCapsuleList

**Implement motion planner and controller**
- Learn through reference example
- Architecture of highway lane change planer
- Closed-loop controller

**Simulate test bench**
- Scenarios in straight and curved roads
- Scenarios imported from HERE HD map
Key takeaways

- Demonstrated how to design and simulate an automated lane change maneuver (LCM) system for highway driving.
- These reference examples can be used as a good framework for developing a custom LCM system.
- Navigation and Automated Driving Toolbox provide necessary components for the LCM system.

Generate Code for Highway Lane Change Planner

Design, test, and generate C++ code for a lane change planner for highway driving. This example closely follows the Highway Lane Change planner.

Highway Lane Change

Design and test the planner and controller components of a lane change maneuver system designed for highway driving.

Closed-loop system + MPC controller + vehicle dynamics

R2021a

Automated Driving Toolbox™
Navigation Toolbox™
Model Predictive Control Toolbox™

referencePathFrenet
trajectoryGeneratorFrenet
dynamicCapsuleList
Thank you