Evaluate Path Planner and Controller for Automated Parking

Shusen Zhang

Application Engineering, MathWorks
Learn about path planning with these examples

**MATLAB example**

**Automated Parking Valet**

Construct an automated parking valet system using path planning, trajectory generation, and vehicle control techniques.

*Open Script*

**Simulink test bench**

**Automated Parking Valet in Simulink**

Construct an automated parking valet system in Simulink with Automated Driving Toolbox.

*Open Model*
Learn about path planning with these examples

- How robust is the algorithm?
- How can I handle moving pedestrian?
- How can I automate the tests?
Evaluate Path Planner and Controller for Automated Parking

- Explore system robustness with simulation
- Improve design to handle moving pedestrian
- Test automation for regression tests
Requirement 1: vehicle can only move forward
Requirement 2: vehicle can’t cross parking lanes
Requirement 3: algorithm must be able to handle loop.
Baseline model in the product
Single test case in the baseline model
Software model
Main modules

Planning

Inflation

Collision Checking
Main modules

Planning

Path Analyzer

Control
Explore baseline behavior with multiple goal poses
Explore behavior with single goal pose
Explore behavior with single goal pose
Changing testing condition

Block Parameters: BehaviorPlanner

Subsystem (mask)

Activate a sequence of navigation tasks from global route plan.

Parameters

Route plan: routePlan

Initial vehicle speed [m/s]: 0

[GUI elements: OK, Cancel, Help, Apply]

Diagram showing navigation tasks and inflated areas.
Explore behavior with goal pose that generates a loop in path
Isolate issue based on simulation results

Planner finds the goal

Path Analyzer

Controller is mostly tracking
Existing path analyzer

- Find a point on the path for the vehicle to follow

% Find the closest point on the reference path
[closestIdx, ~] = find(dis2PointsSquare == min(dis2PointsSquare), 1);

% Enforce to be a scalar in Simulink
closestIdx = closestIdx(1);
Existing path analyzer
Modified path analyzer

- Find a point on the path for the vehicle to follow

% Normalized distance between current position and section starting point
u = (RXY.*DeltaXY)/(DeltaXY.*DeltaXY);

% Find section ending point
indexIncrement = ceil(u-1);
Modified path analyzer
Explore behavior with improved path analyzer
Reduce turning radius and speed
Evaluate Path Planner and Controller for Automated Parking

- Explore system robustness with simulation
  - Remove intermedia points
  - Specify different parking maps and spots
  - Identify design issue and improve the design.

- Improve design to handle moving pedestrian

- Test automation for regression tests
Build test bench to test dynamic scenario
Identify changes to react to pedestrian

1. Update map based on scenario
2. Aware of potential collision
3. React to imminent collision
Update map based on scenario

1. Update map based on scenario
2. Aware of potential collision
3. React to imminent collision
Update map based on scenario
Identify changes to react to pedestrian

1. Update map based on scenario

2. Awareness of potential collision

3. Action with potential collision
Awareness of potential collision

Time-to-collision calculation
Time-to-collision with known path
Identify changes to react to pedestrian

1. Update map based on scenario

2. Awareness of potential collision

3. Action with potential collision
Action with potential collision

- Reduce speed when TTC is low
Test in dynamic environment
Improve design to handle pedestrians
Evaluate Path Planner and Controller for Automated Parking

- Explore system robustness with simulation
  - Remove intermedia points
  - Specify different parking maps and spots
  - Identify design flaws and improve the design.

- Improve design to handle moving pedestrian
  - Add moving pedestrian
  - Create costmap from ground truth
  - Reduce speed based on time-to-collision

- Test automation for regression tests
Automate regression testing
Test assessment metrics

Input:
1. DistanceMetricAcceptable
2. YawDiffMetricAcceptable
3. IsTrajValid

Output:

Local

Constant

Step:
- VerifyMetrics
  - VerifyActualVSRefMetrics
    - verify(duration(DistanceMetricAcceptable==false)<0.5)
    - verify(duration(YawDiffMetricAcceptable==false)<0.5)
    - verify(IsTrajValid==true)
Test assessment metrics

Distance Metric

Yaw Metric

TTC Metric

Reference Point

Pedestrian

Trajectory
Test iteration

- Tests definition and test management
Testing multiple goal poses
Automate regression testing

- Use test manager to inspect reason of failed test.

Yaw Metric
Automate regression testing

Less deviation with higher gain
Automate regression testing

- Re-planning and latency
- Environment model uncertainty
- Complex parking lot
- Backward motion
Automate regression testing
Evaluate Path Planner and Controller for Automated Parking

- Explore system robustness with simulation
  - Remove intermediate points
  - Specify different parking maps and spots
  - Identify design flaws and improve the design

- Improve design to handle moving pedestrians
  - Add moving pedestrian
  - Create costmap from ground truth
  - Reduce speed based on time-to-collision

- Test automation for regression tests
  - Add metrics for planner and controller
  - Add test case definition/ management