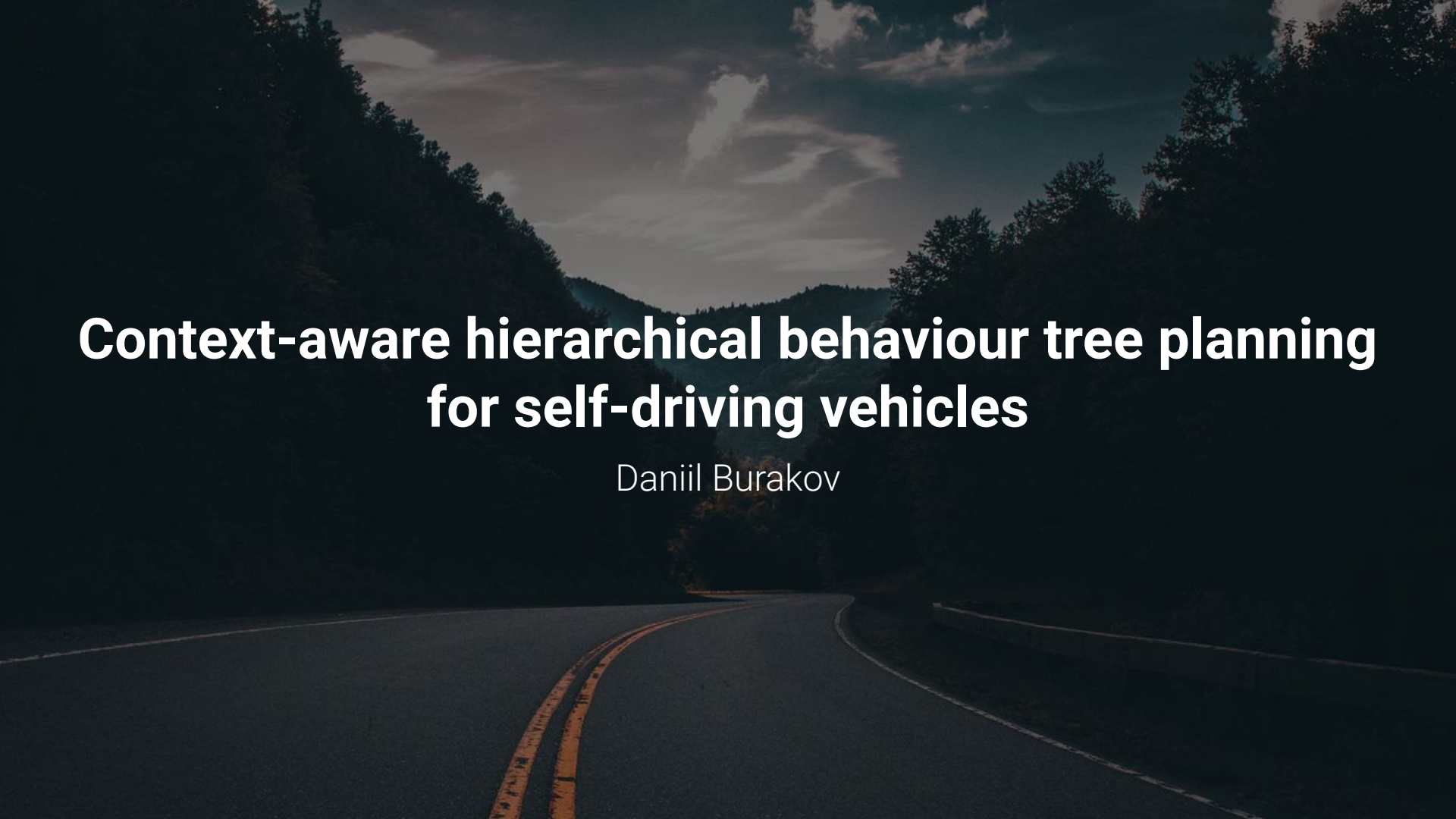


Context-aware hierarchical behaviour tree planning for self-driving vehicles

Daniil Burakov

Agenda

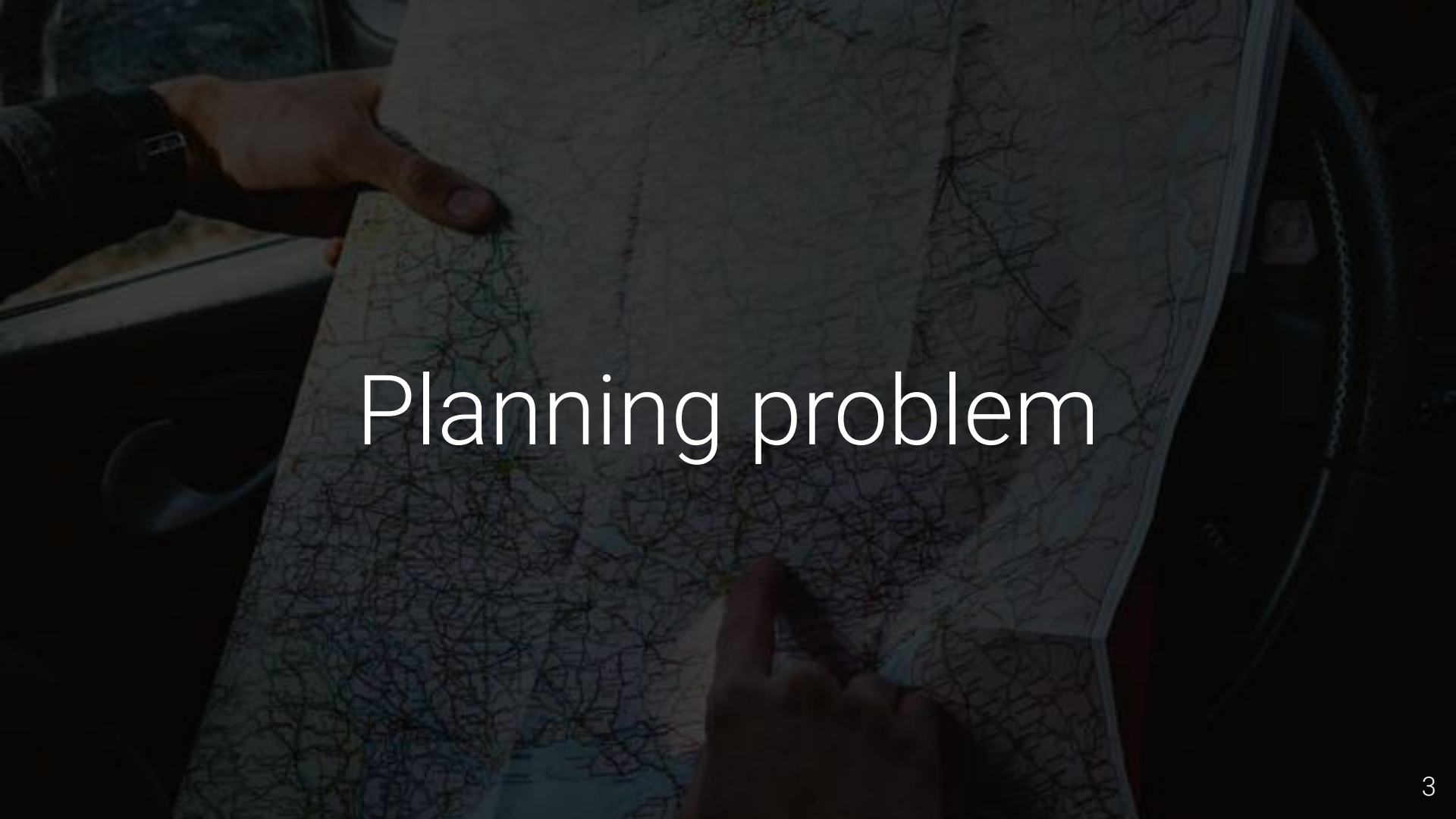
Planning Problem

Literature Review

Proposed Solution

Current Progress

Next Steps


Planning problem

Planning problem

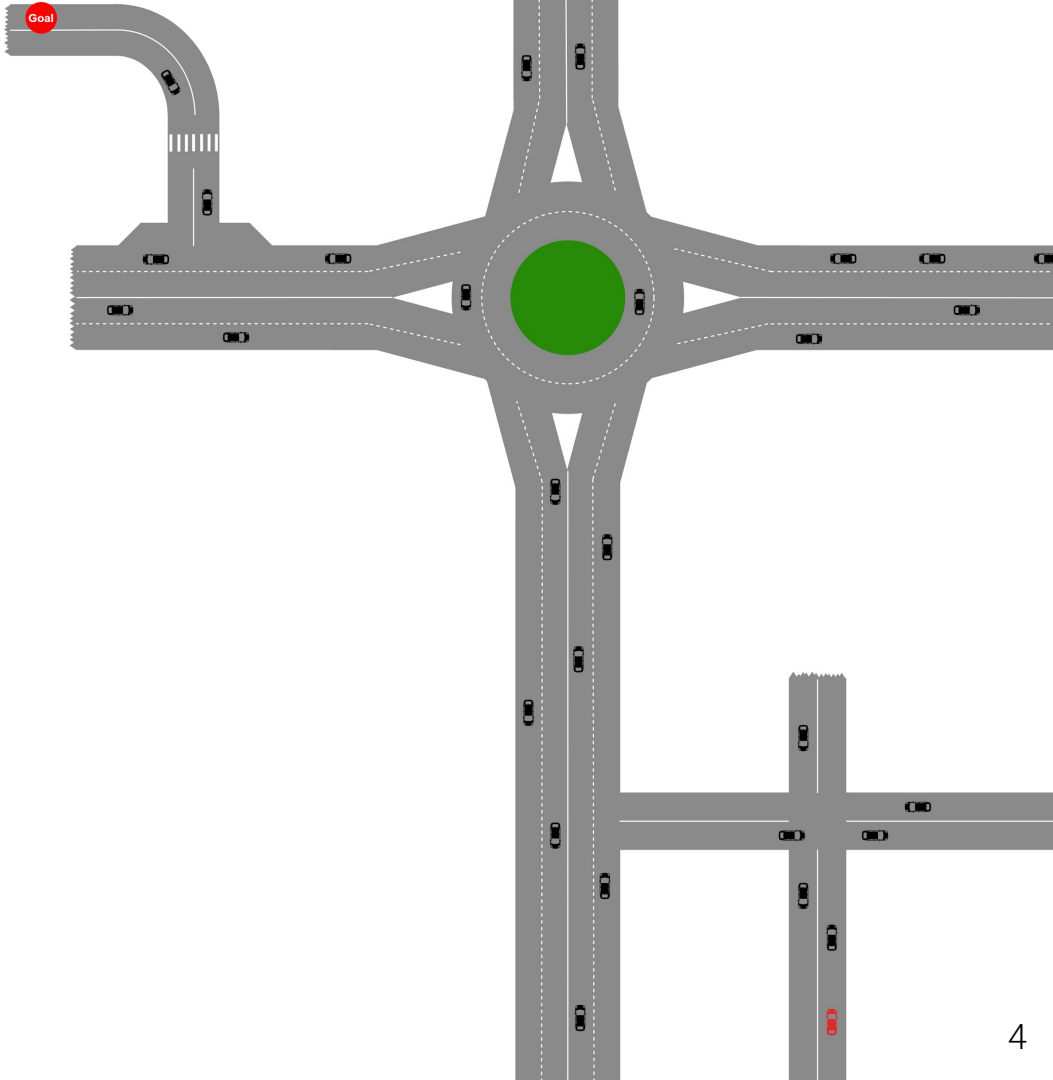
Safety

Comfort

Traffic Rules

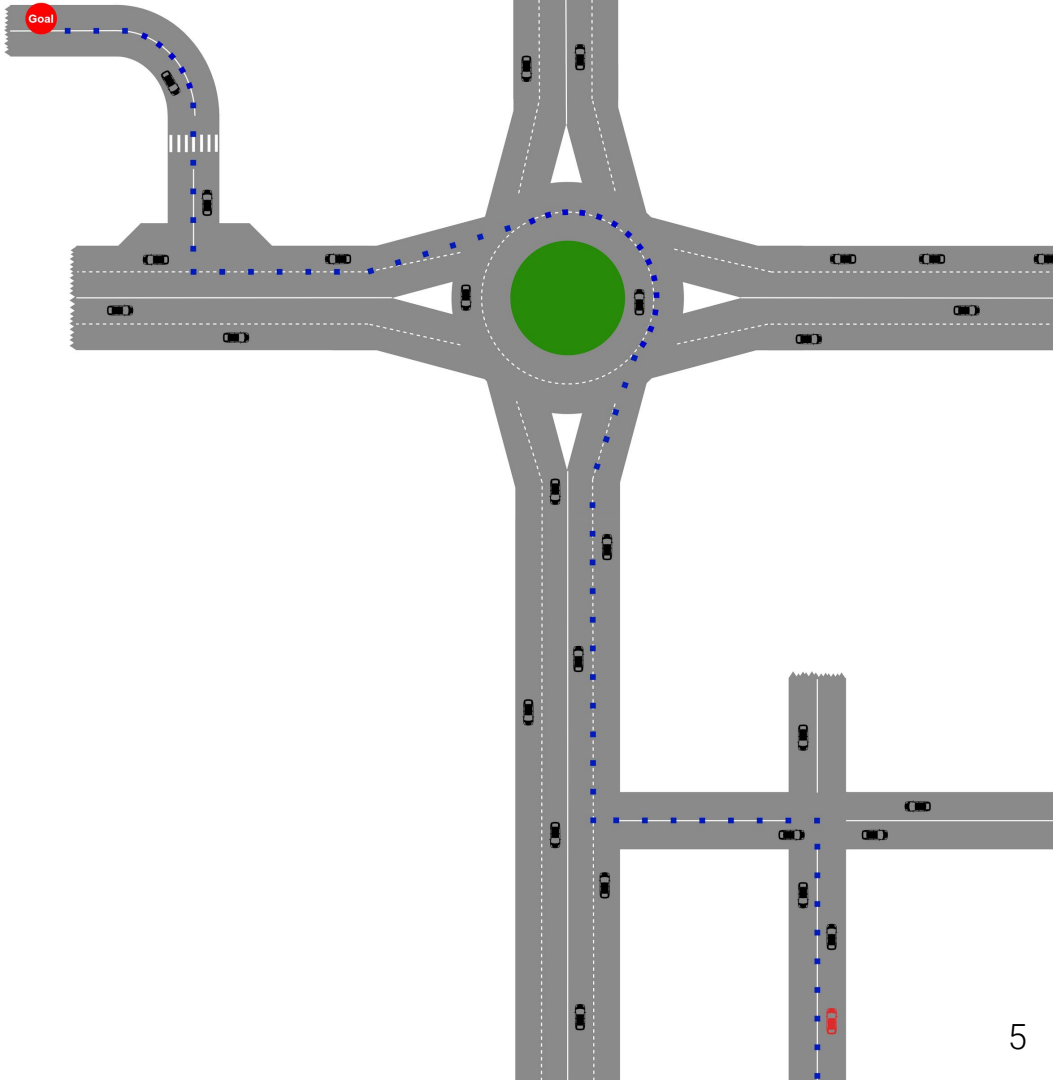
Predictability of arrival

Social interaction



Planning problem

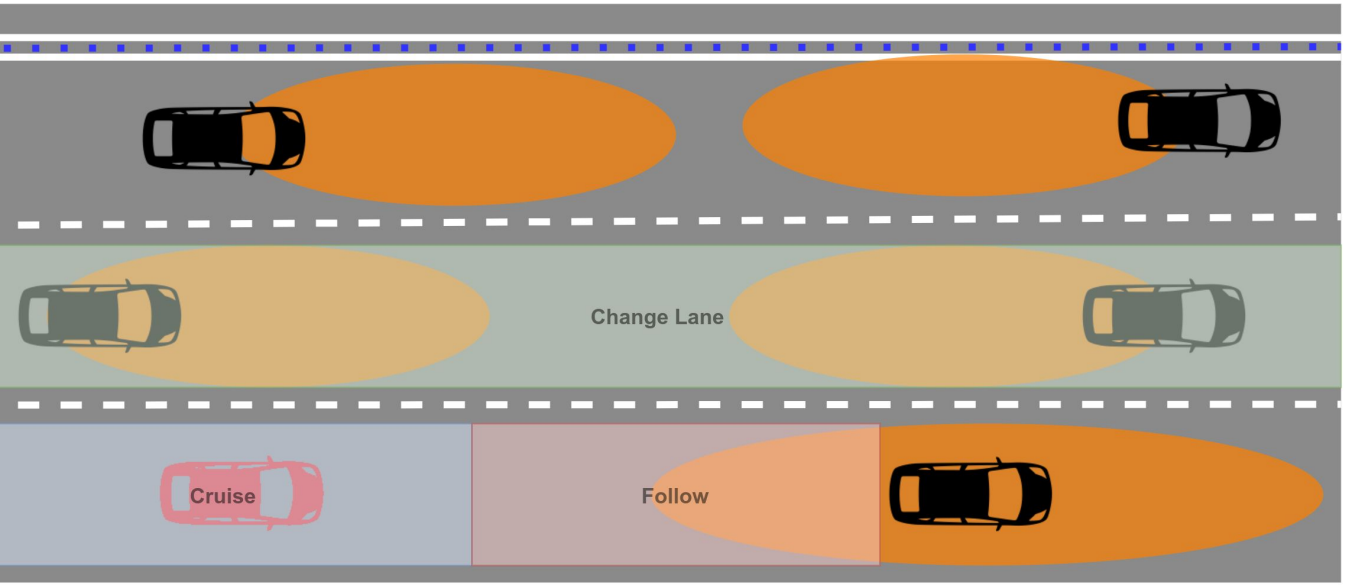
Mission Planning

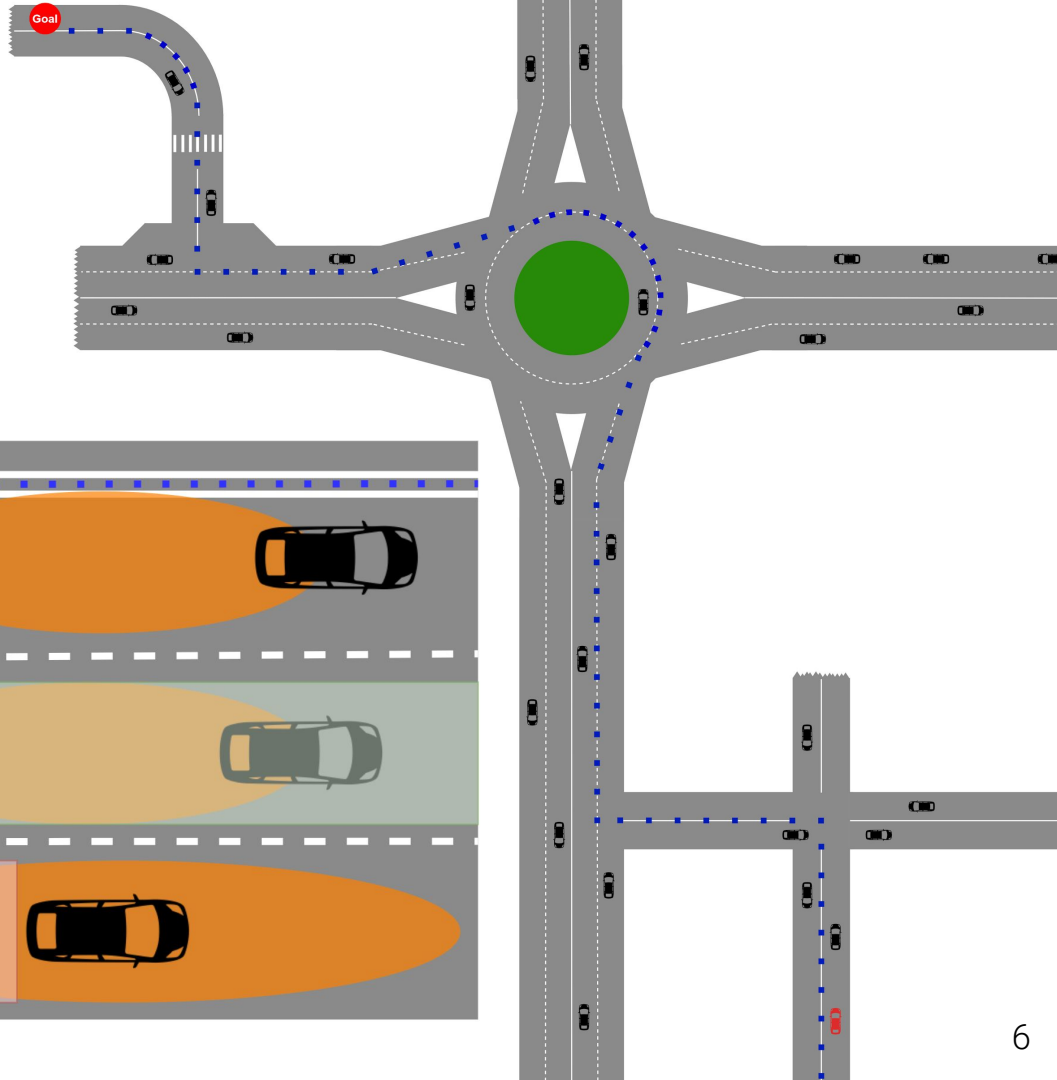


Planning problem

Mission Planning

Behaviour Planning



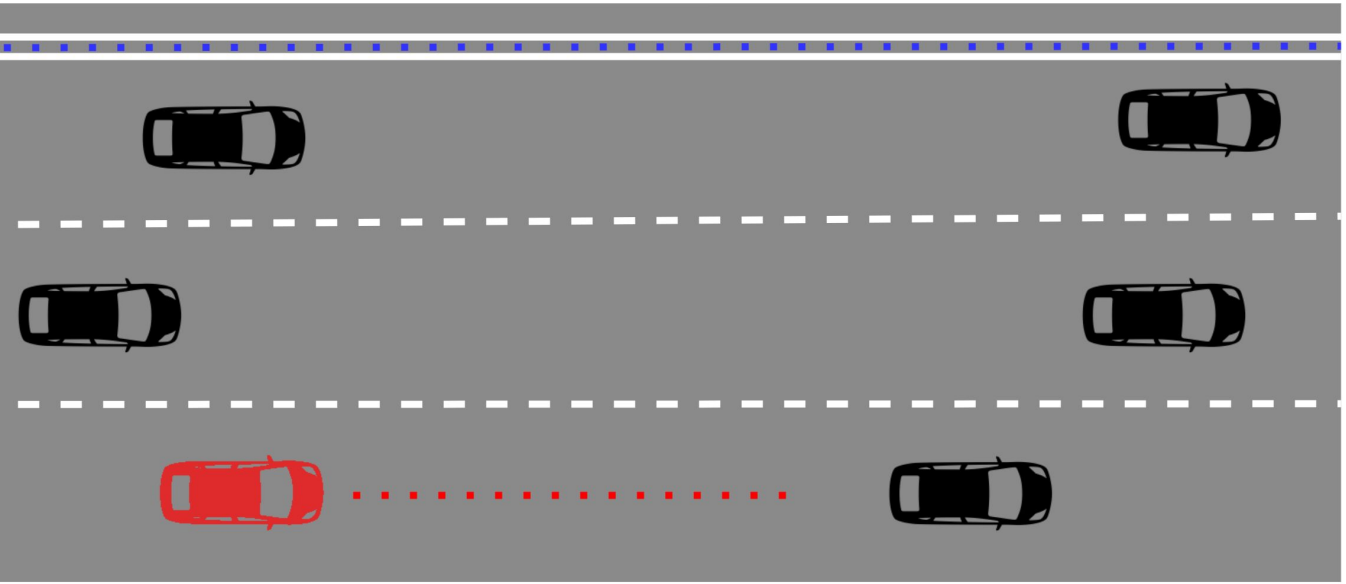


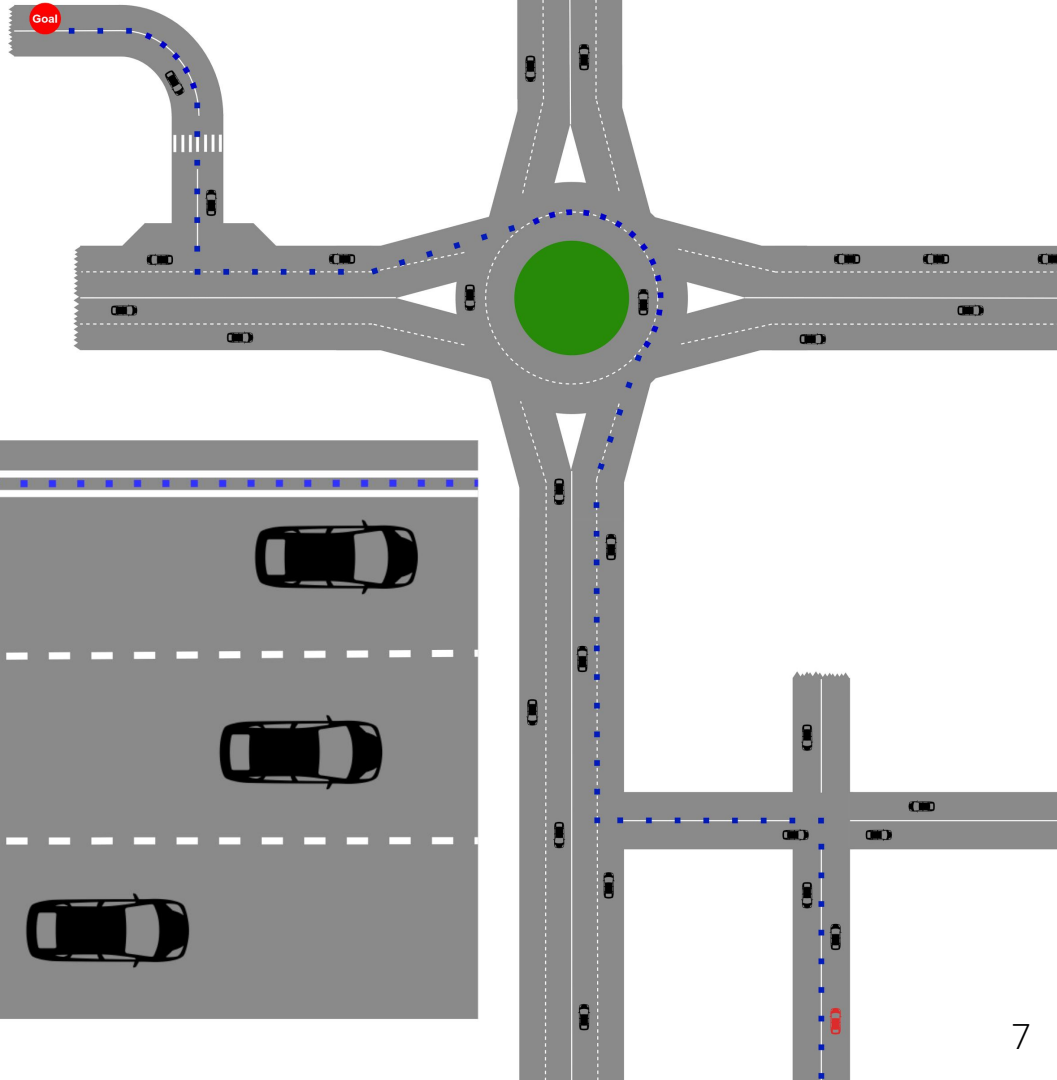
Planning problem

Mission Planning

Behaviour Planning

Local Planning






Literature review

DARPA Urban Challenge

Autonomous Driving in Urban Environments: Boss and the Urban Challenge

Tartan Racing: A Multi-Modal Approach to the DARPA Urban Challenge

Junior: The Stanford Entry in the Urban Challenge

Tartan: DARPA Urban Challenge Winner 2007

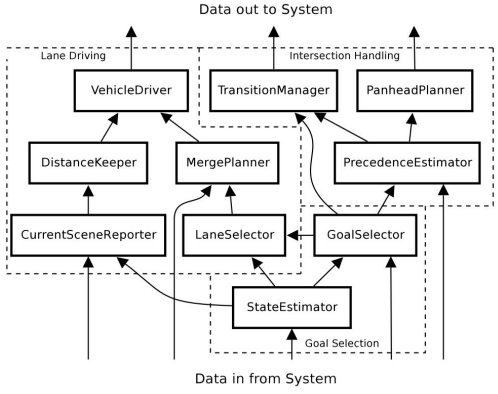


Figure 14. High-level behaviors architecture.

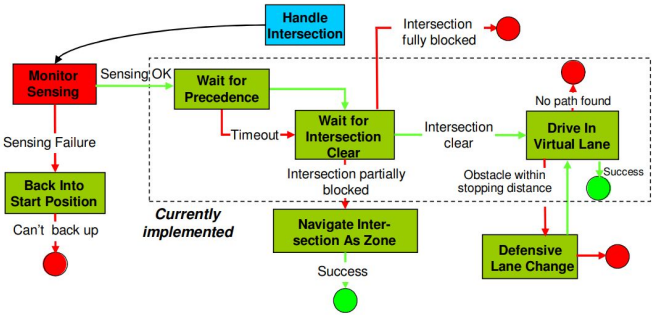


Figure 4. Schematic representation of the "Handle Intersection" behavior.

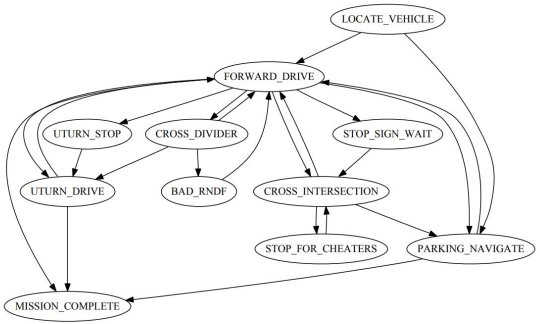
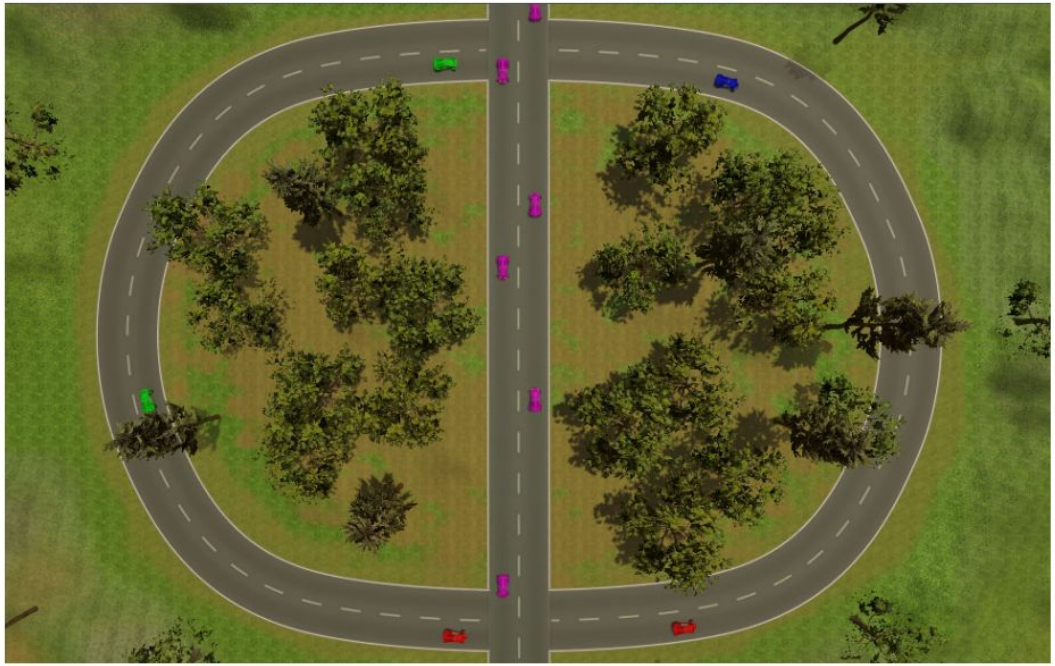


Figure 21: Finite State Machine that governs the robot's behavior.

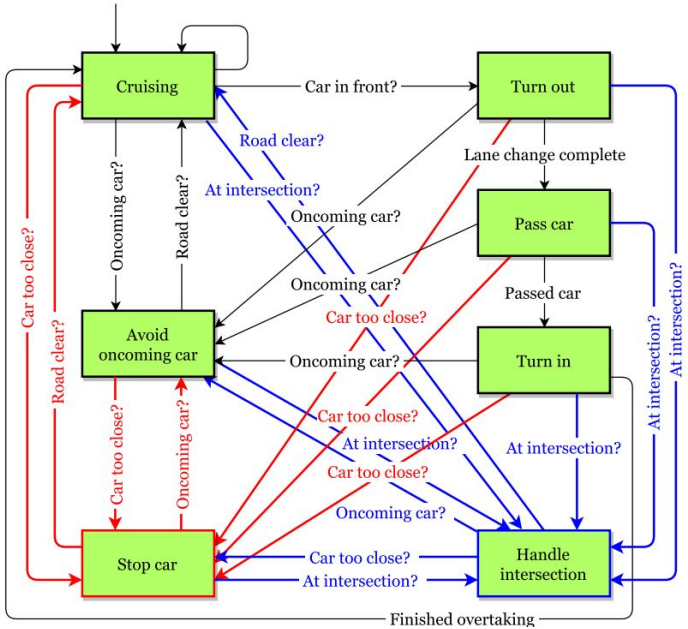
FSMs vs Behaviour Trees

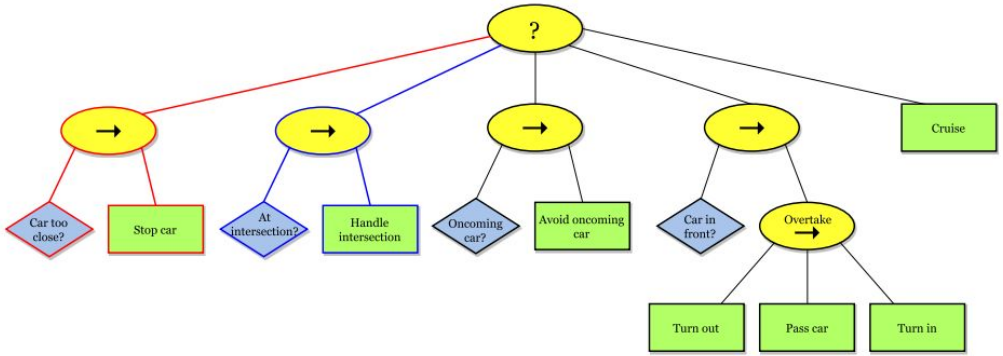
Behavior Trees for decision-making in autonomous driving



FSMs vs Behaviour Trees

Behavior Trees for decision-making in autonomous driving





FSMs vs Behaviour Trees

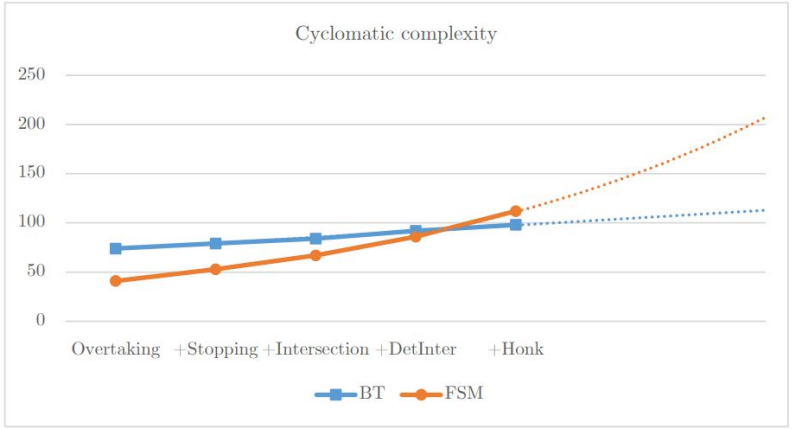


Figure 5.4. Plot of Cyclomatic complexity metrics with extrapolation (less is better).

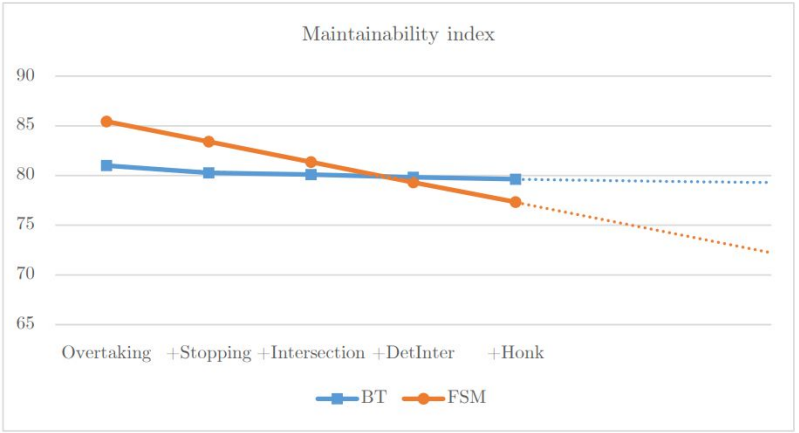


Figure 5.5. Plot of Maintainability index metrics with extrapolation (higher is better).

Petri Nets

Implementing Autonomous Driving Behaviors Using a Message Driven Petri Net Framework

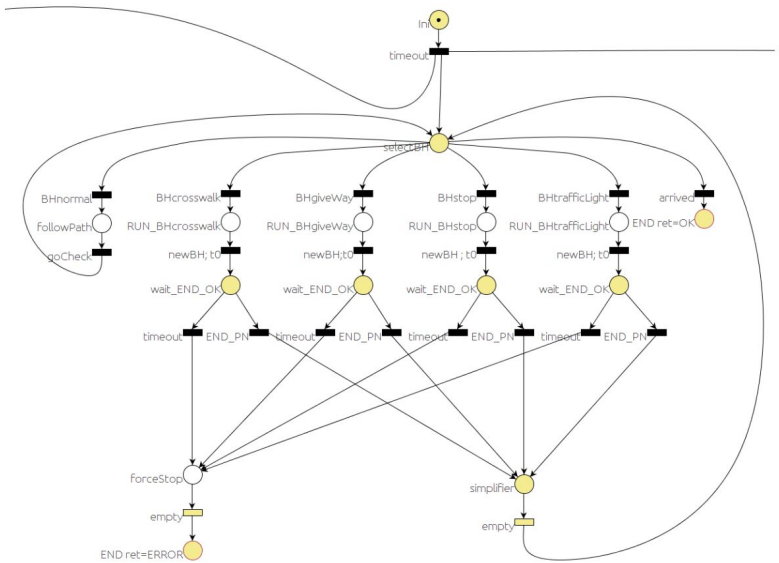


Figure 5. Selector PN: the behavior execution part.

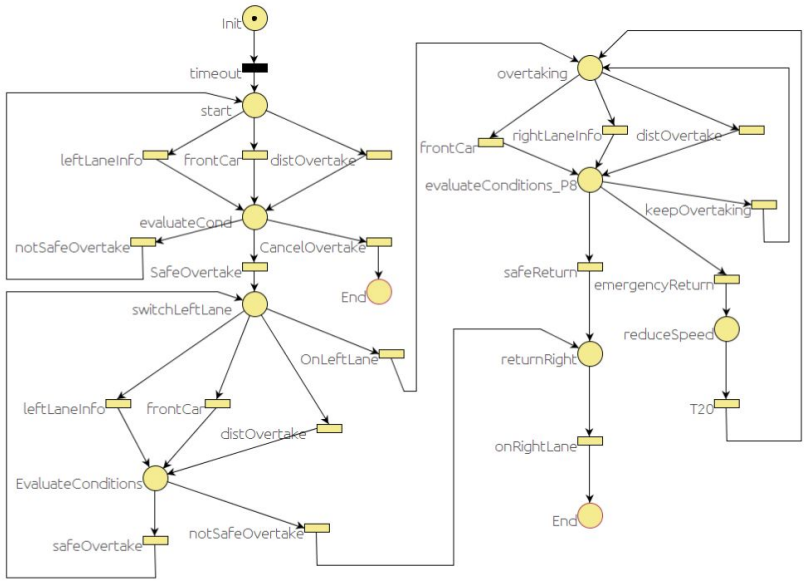


Figure 9. Overtake behavior PN.

Petri Nets

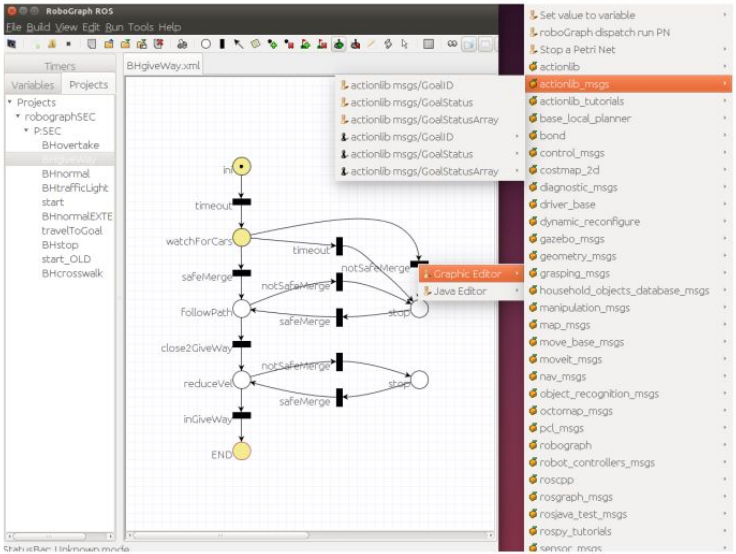


Figure 2. RG GUI editing the BHgiveWay Petri net (PN).

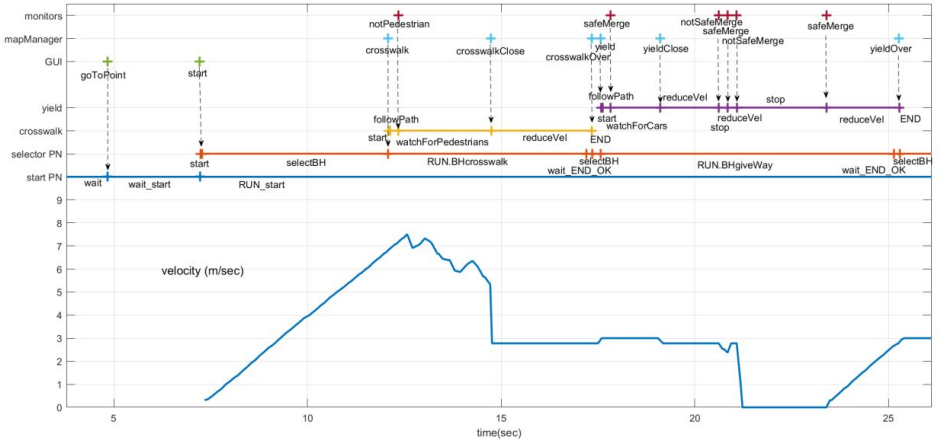
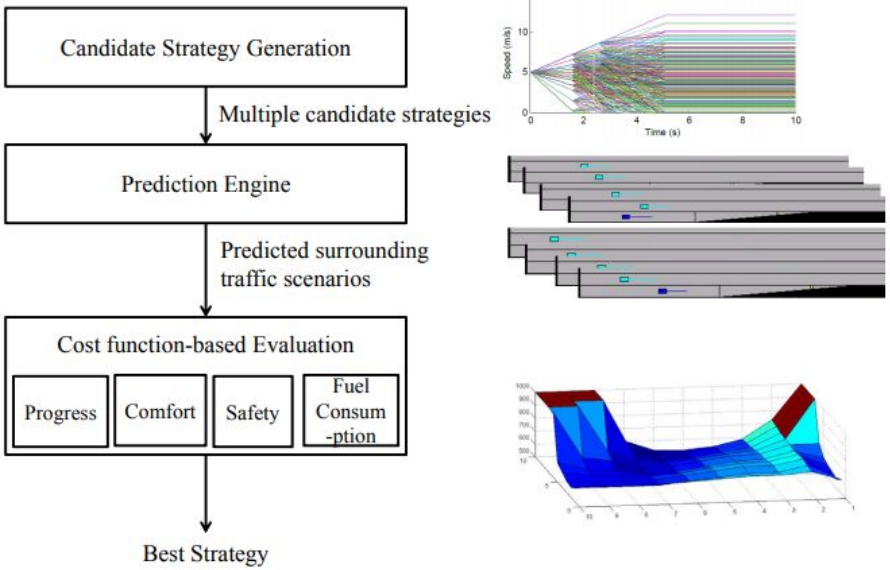


Figure 16. Temporal diagram for entering a roundabout scenario. At the top, the events produced by modules monitors, mapManager, and GUI. In the middle, the evolution of crossWalk, yield, selector, and start PNs. At the botton, the velocity commanded to the car.

Candidate path behaviour selection

A Behavioral Planning Framework for Autonomous Driving

Baidu Apollo EM Motion Planner



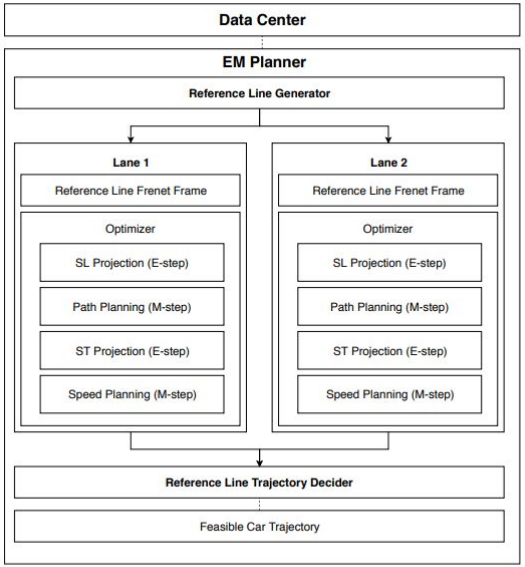
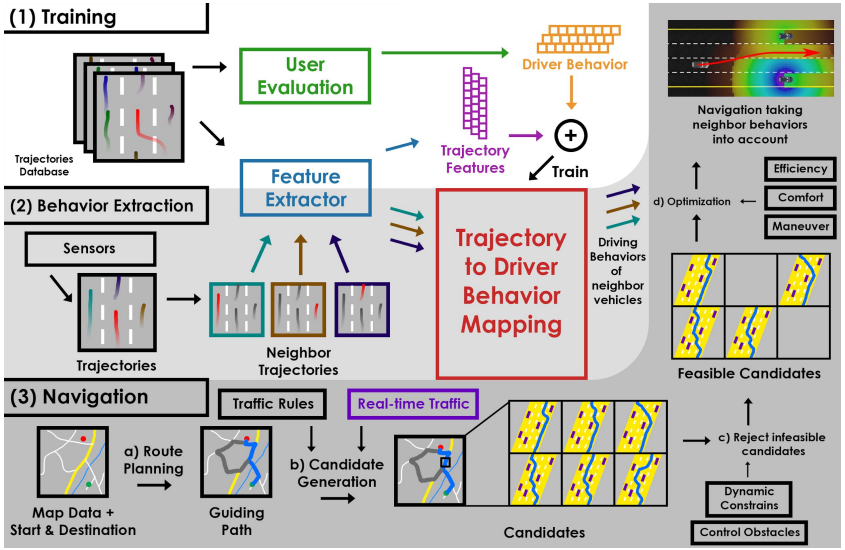


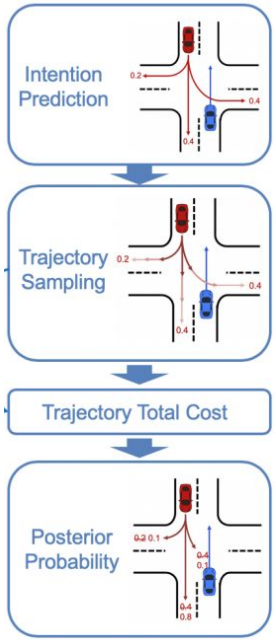
Fig. 2: EM Framework

Data-driven approaches

Behavior Modeling for Autonomous Driving

Data Driven Prediction Architecture for Autonomous Driving and its Application on Apollo Platform





Proposed solution

Proposed solution

Modified hierarchical approach

Scenarios (Contexts) - impose constraints, set desired state goals

Stages (Hierarchies) - set local goals

Behaviours - implement stages. Use Behaviour trees

Scenario

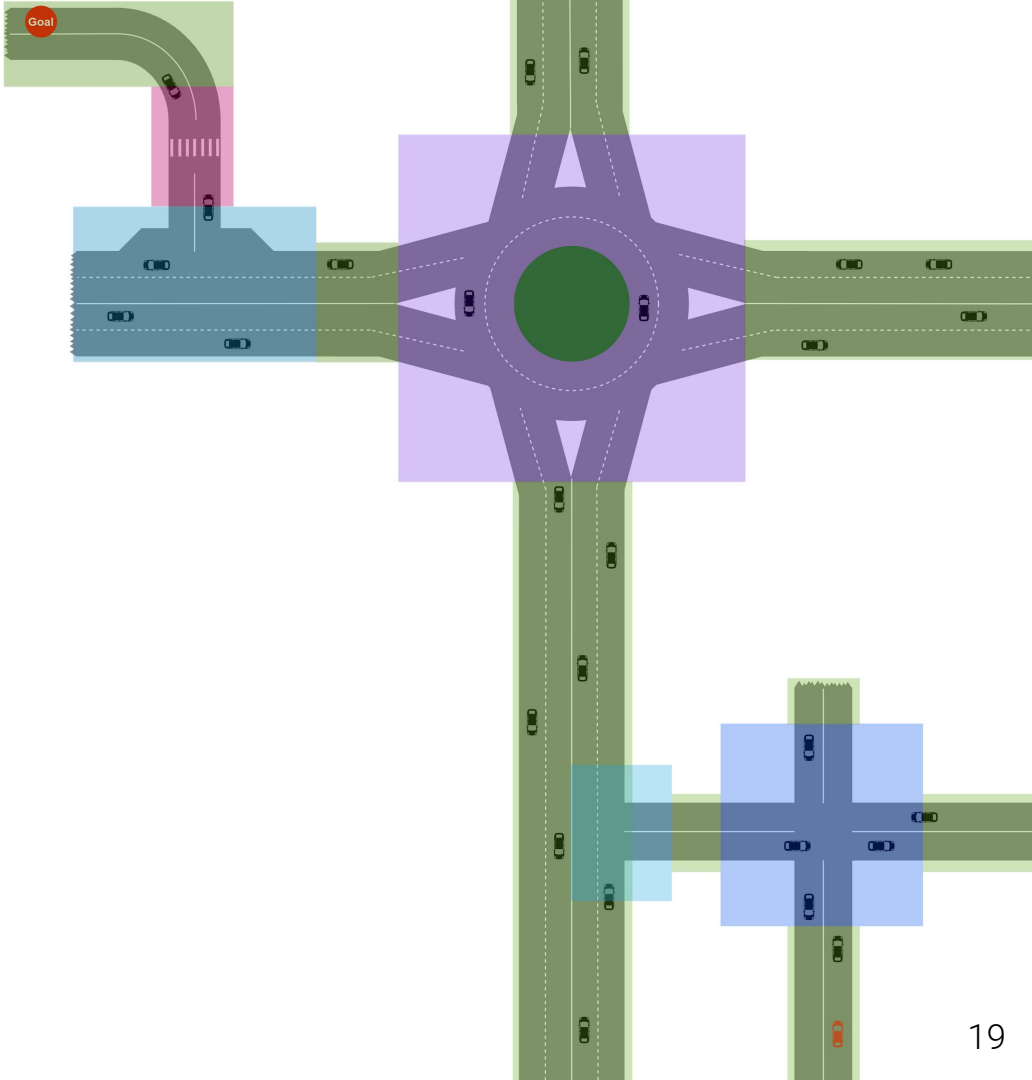
Imposes requirements and constraints

for example, enable traffic light detection module when in controlled intersection scenario and set new acceleration constraints

Next scenario can request the desired state for itself from the current scenario

for example, takeover scenario can request highway driving scenario to finish on the rightmost lane

Can be determined via map or perception



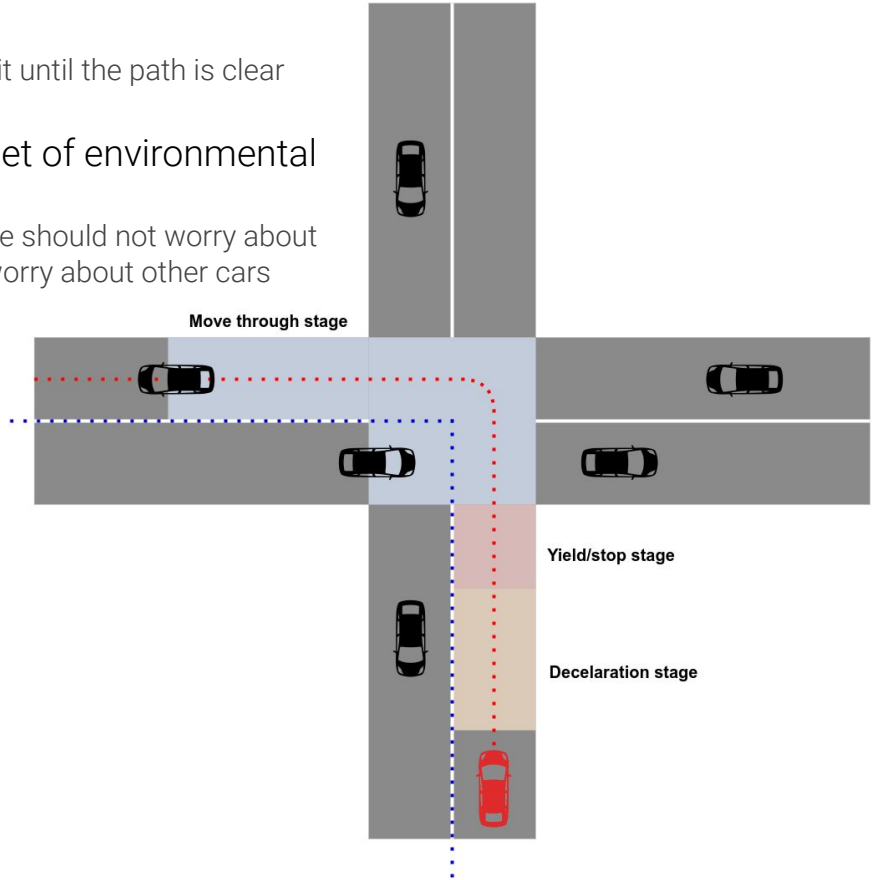
Stage

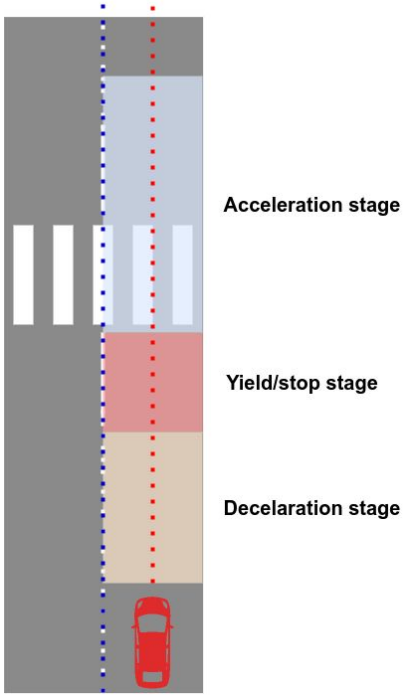
Determines local goals

for example: decelerate to stop, wait until the path is clear

Narrows focus to reduced set of environmental features

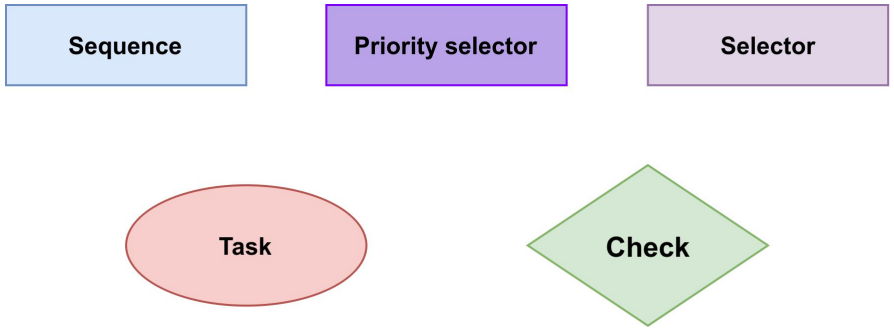
in deceleration stage the ego vehicle should not worry about cars on the intersection, it should worry about other cars when it is in the yield/stop stage





Behaviours: Behaviour tree

Implements Stages

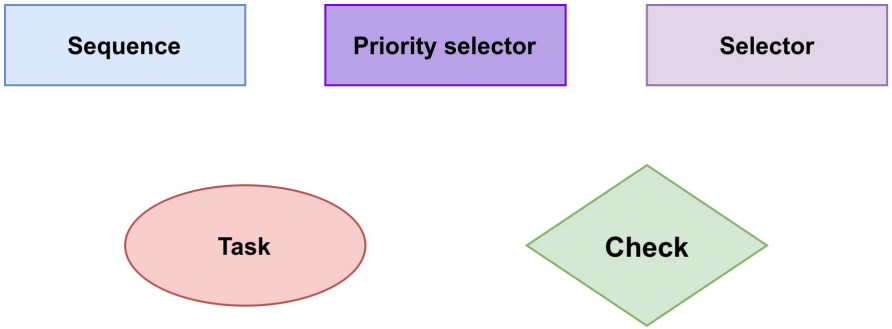


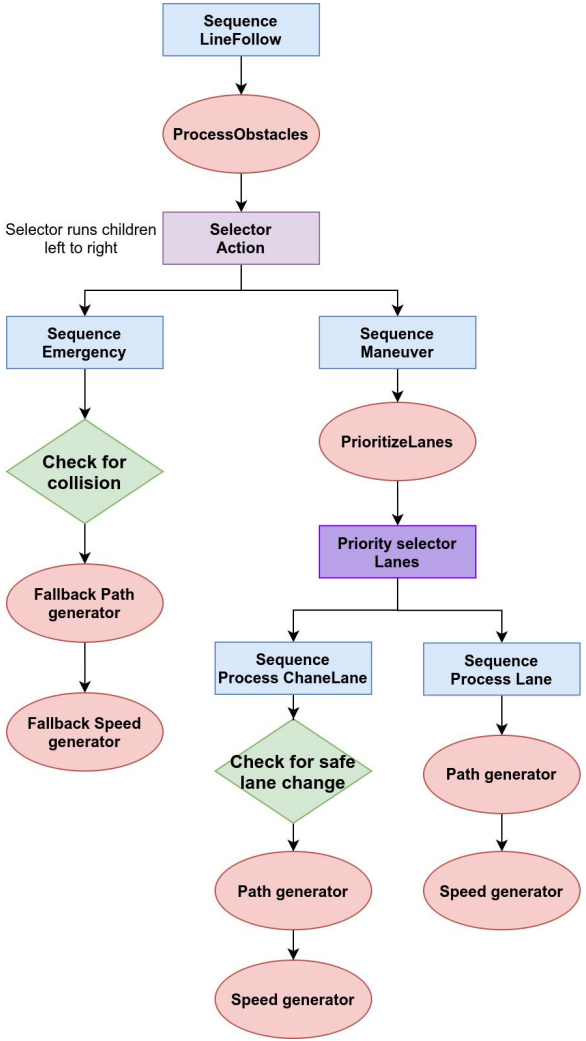
Behaviours: Behaviour tree

Implements Stages

Benefits:

1. Readability due to abstraction
2. Have the opportunity of quick response
3. Reusability of nodes in other behaviours
4. Can perform synthetic tests and formal proofs
5. Data-driven weight selection





Proposed solution

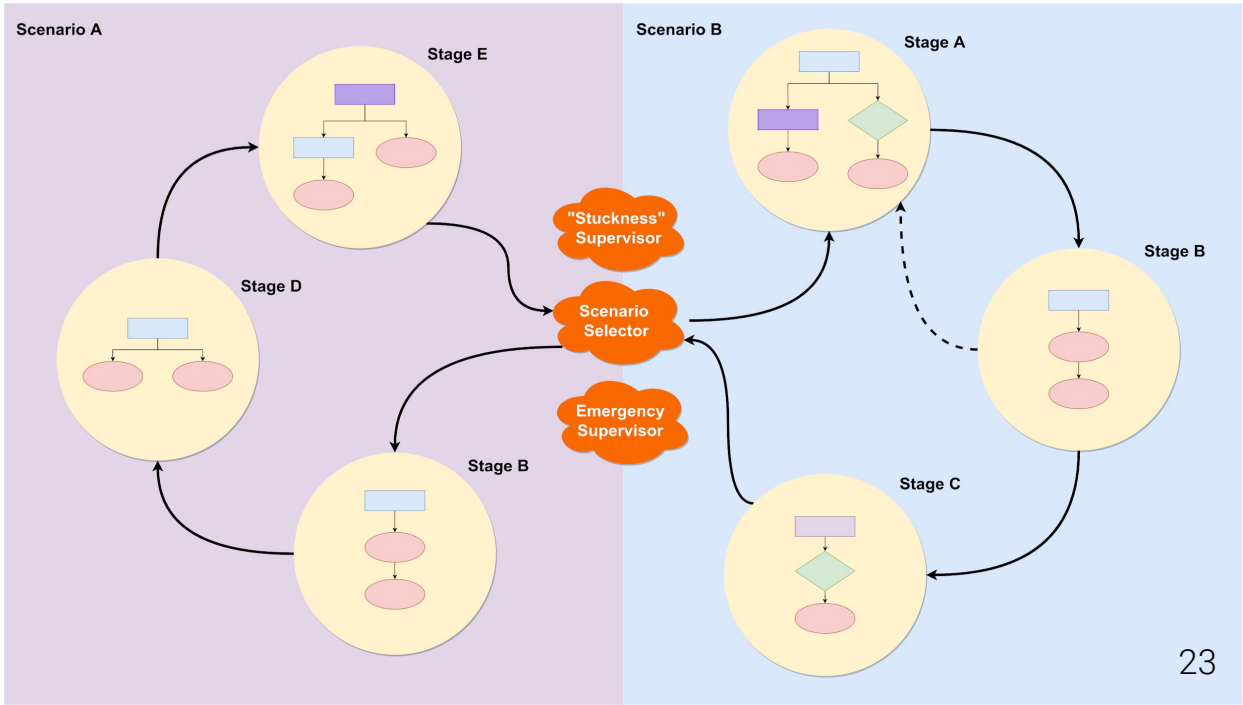
Scenarios (Contexts) - impose constraints, set desired state goals

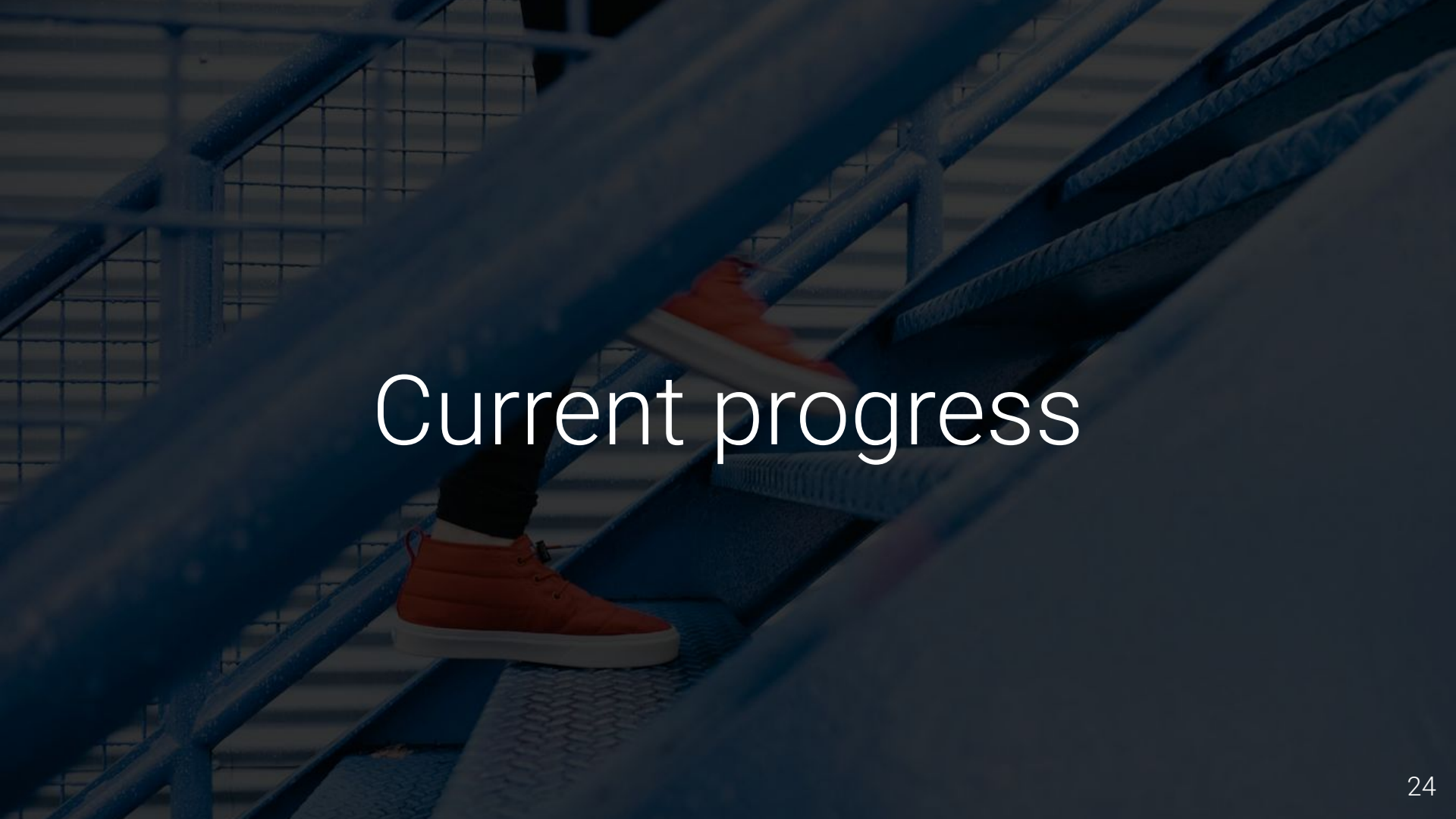
Stages (Hierarchies) - set local goals

Behaviours - implement stages

Data Frame - data structure for synchronization

Data Frame



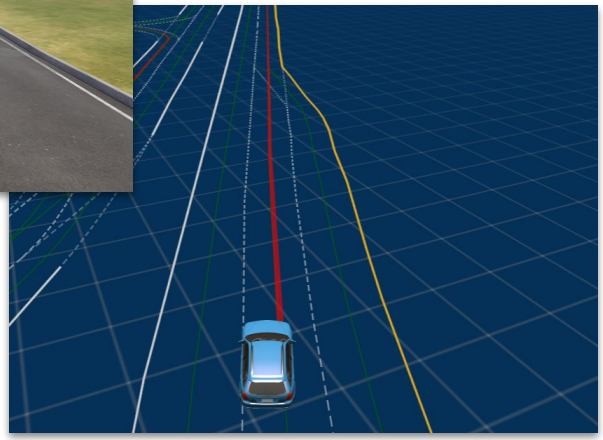


Current progress

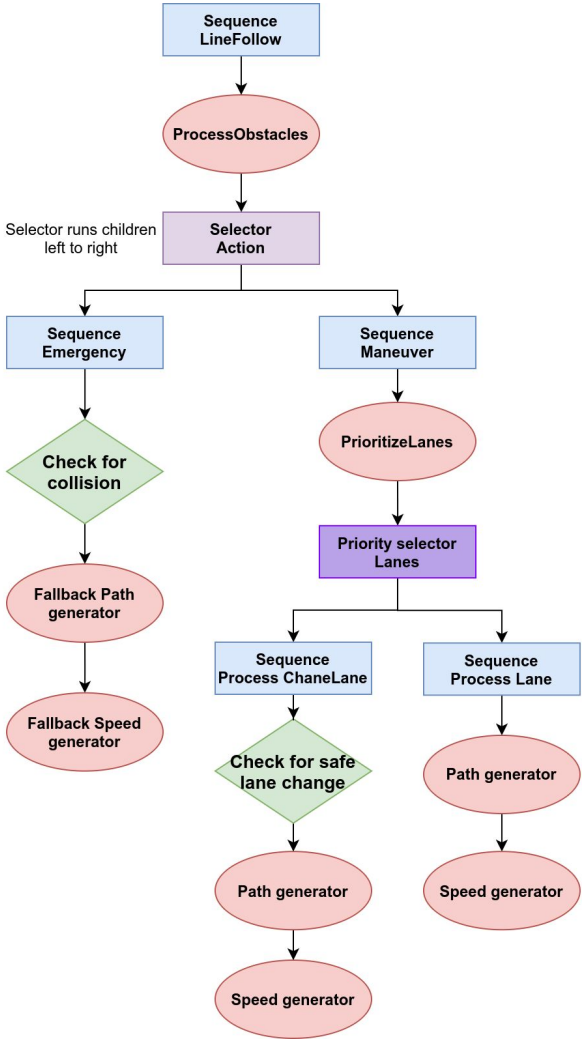
Current progress

1. Connection of Apollo v5.0 with Innopolis Simulator
2. Apollo planning module breakdown
3. Proof of concept implementation

Innopolis simulator



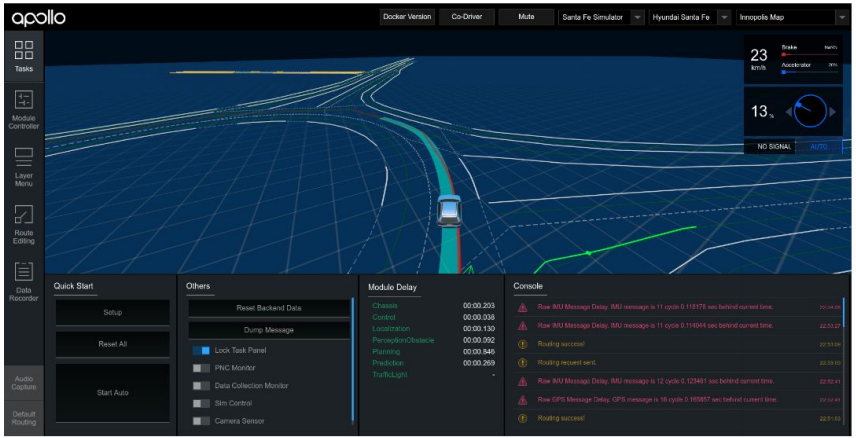
Apollo Dreamview



Apollo v5.0 and Innopolis Simulator

Innopolis Simulator Apollo integration

In this section we will see how to run the Apollo framework along with the Innopolis Simulator.

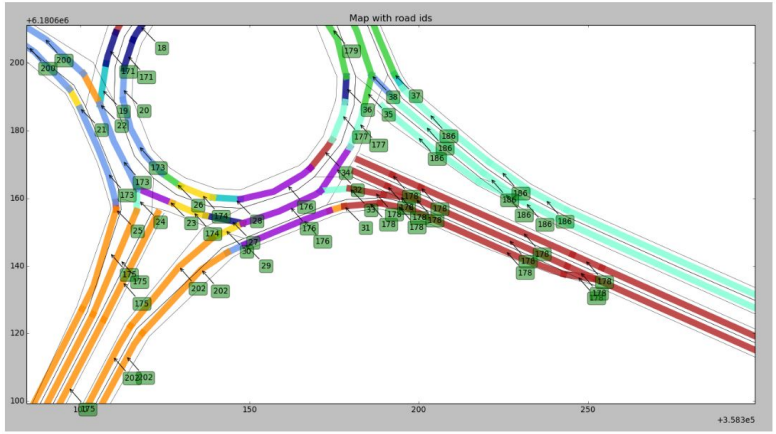


Required modifications

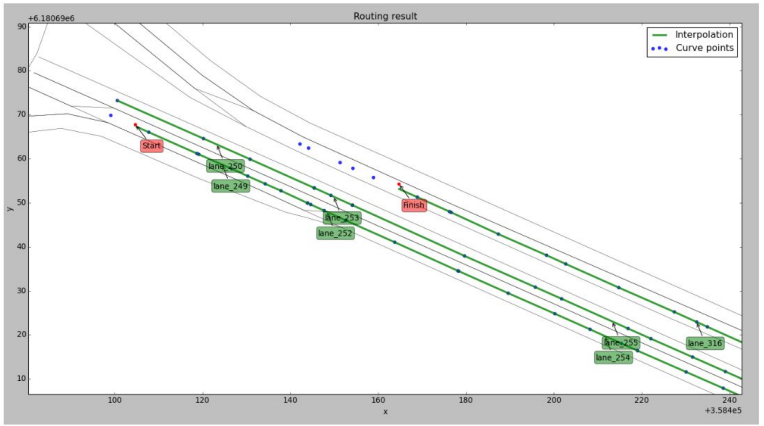
In order to have Innopolis Simulator running with Apollo 5.0 you will have to do the following steps:

1. Add Cyber Bridge
2. Update map protobuf files
3. Add a new vehicle to Dreamview
4. Add a new setup mode to Dreamview

Plot map

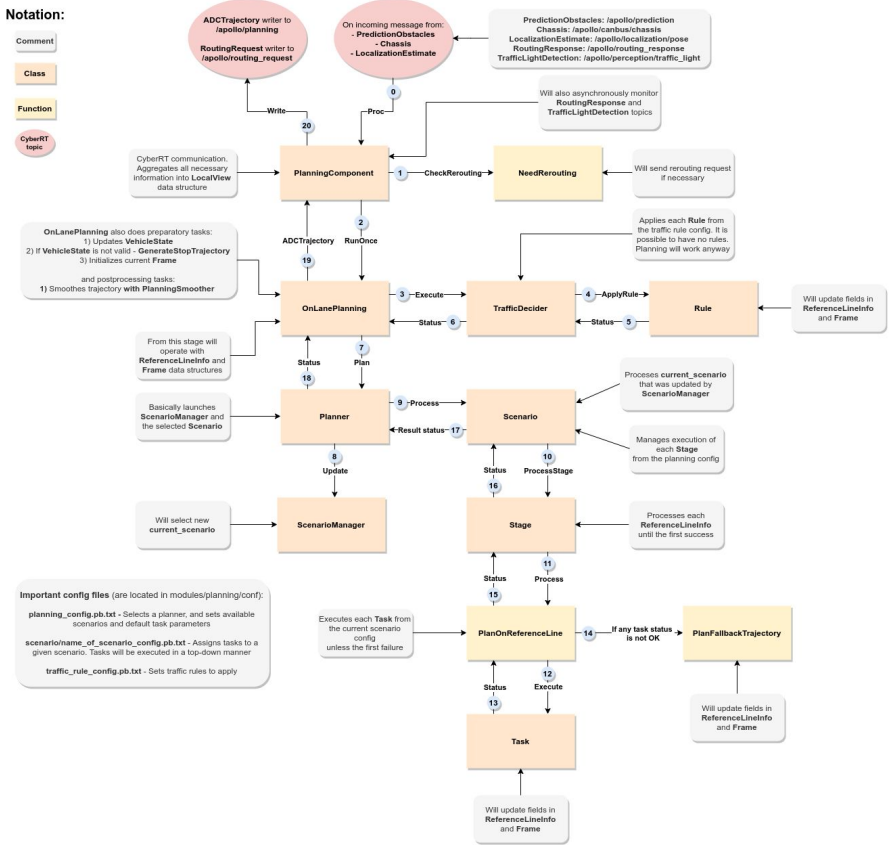


Plot route



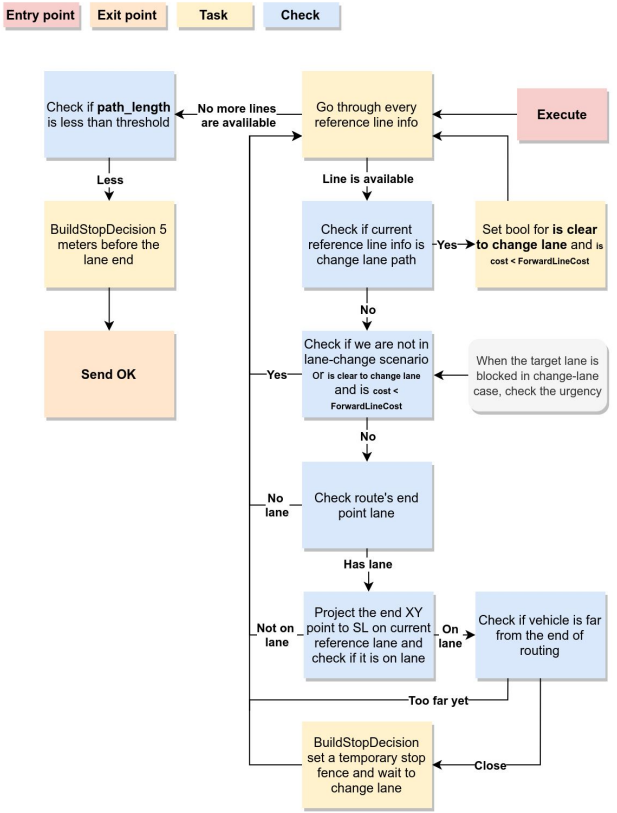
Apollo planning module breakdown

Apollo planning module



7

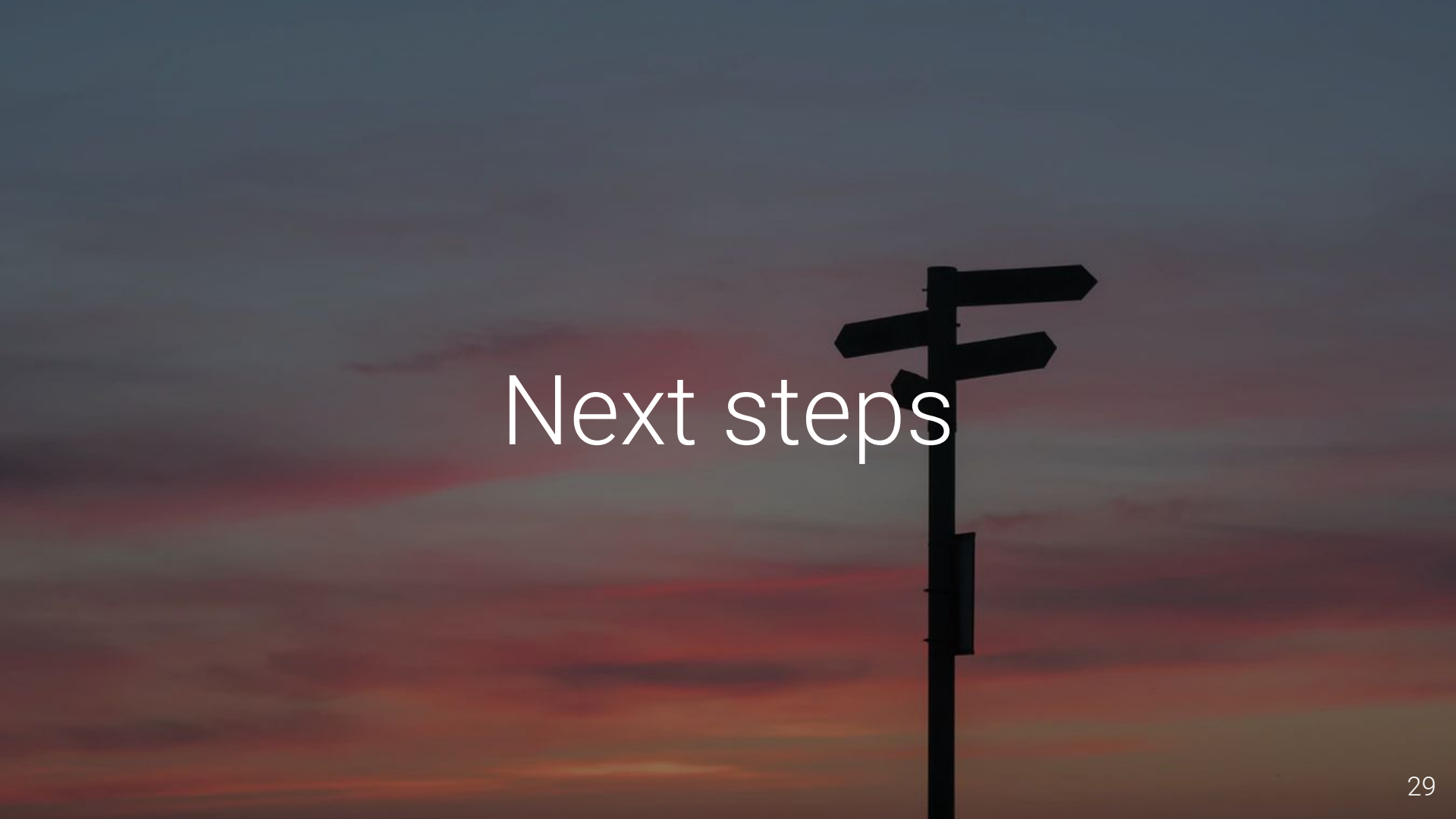
Rule-Based Stop Decider task



Proof of concept implementation

Lane following

Lane change due to obstacle


Next steps

Next steps

1. Modify Apollo Routing module
2. Implement Scenario Selector logic
3. Implement different scenario hierarchical behaviours
4. Visualize behaviour tree execution
5. Create performance measures for planners
6. Create simulation test scenarios
7. Compare the proposed planner with existing Apollo planners

References

- Tartan Racing: A Multi-Modal Approach to the DARPA Urban Challenge
- Junior: The Stanford Entry in the Urban Challenge
- Autonomous Driving in Urban Environments: Boss and the Urban Challenge
- Behavior Trees for decision-making in autonomous driving
- Baidu Apollo EM Motion Planner
- Design Space of Behaviour Planning for Autonomous Driving
- A Behavioral Planning Framework for Autonomous Driving
- Implementing Autonomous Driving Behaviors Using a Message Driven Petri Net Framework
- Path Planning for Autonomous Vehicles in Unknown Semi-structured Environments
- A Driving Behavior Planning and Trajectory Generation Method for Autonomous Electric Bus
- Behavior Modeling for Autonomous Driving
- Data Driven Prediction Architecture for Autonomous Driving and its Application on Apollo Platform
- Apollo Auto GitHub Repository
- Udacity Self-Driving Fundamentals: Featuring Apollo
- Udacity Self-Driving Car Nanodegree
- Self-Driving Cars Specialization by University of Toronto