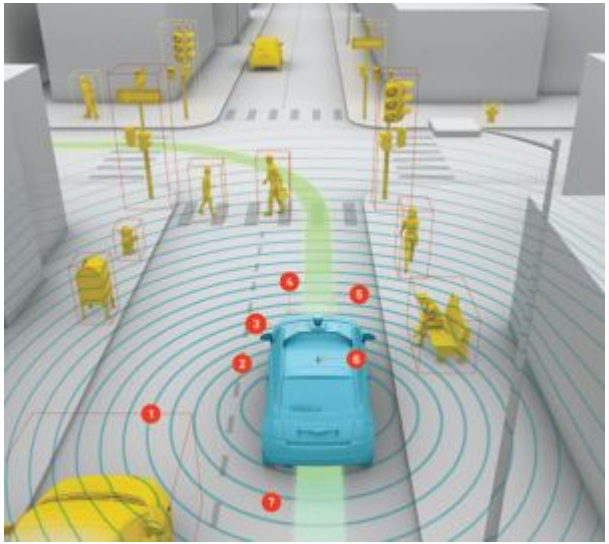
Online Anytime Planning For Autonomous Vehicles

Tianyi Gu

December 8, 2018

realtime robotics

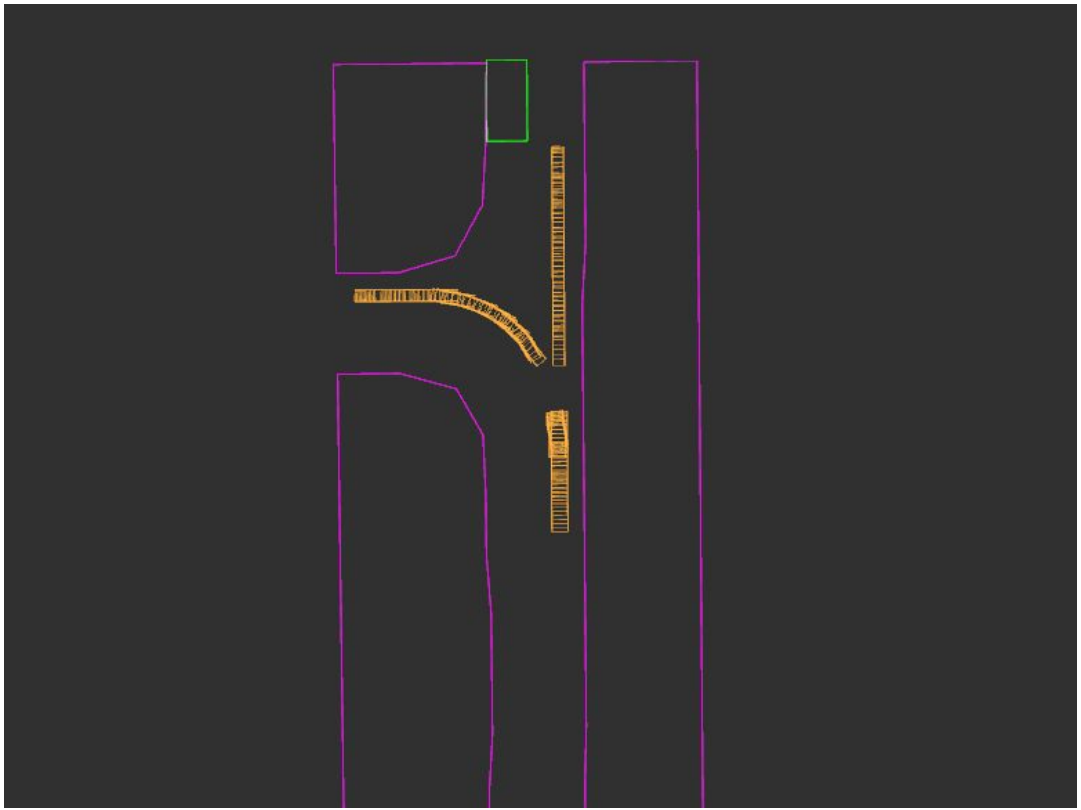


The Autonomous Vehicle Project

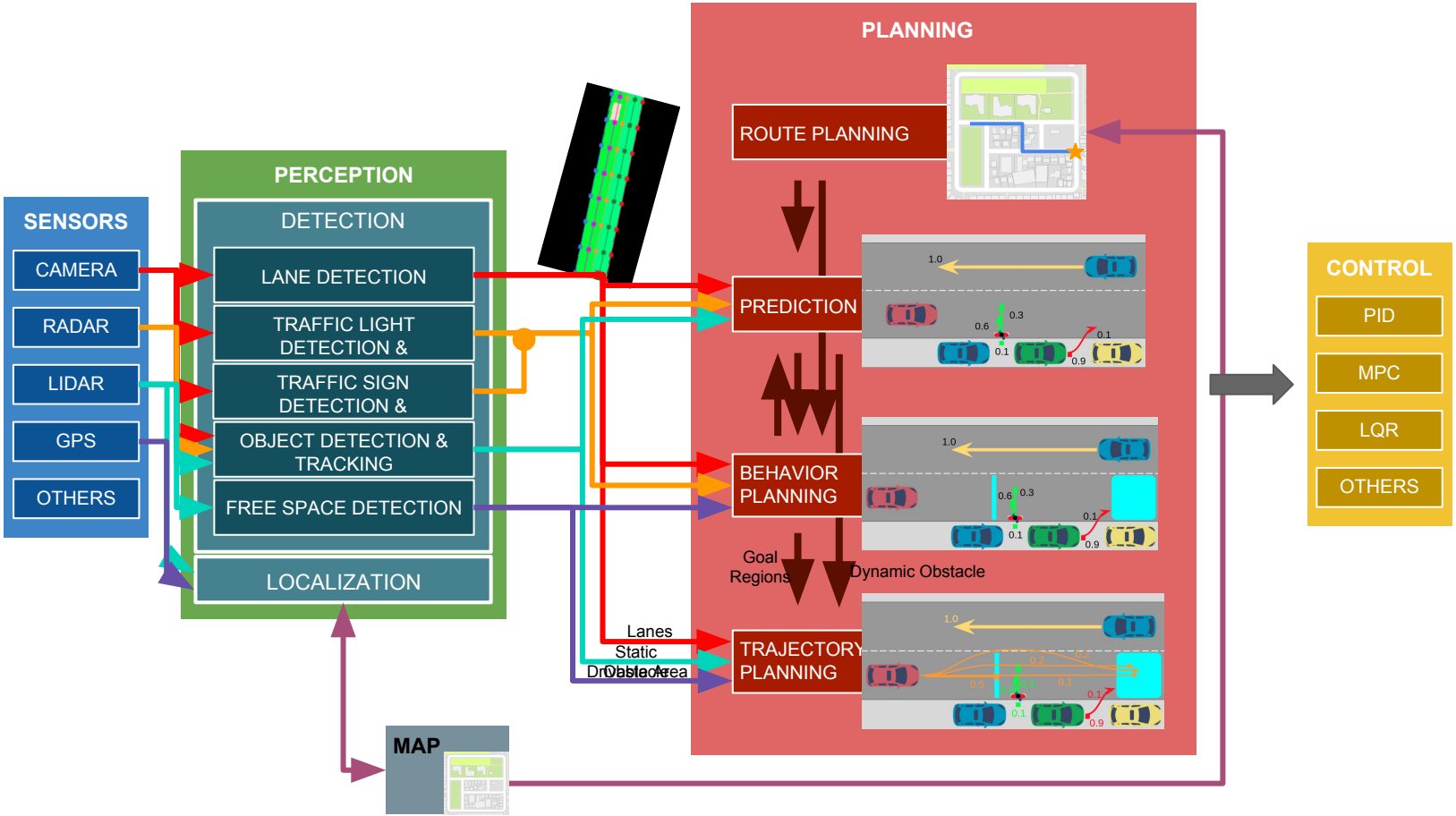
- Online Real-time decision-making framework
- Baseline planner

Background - The Problem

Background - The Problem

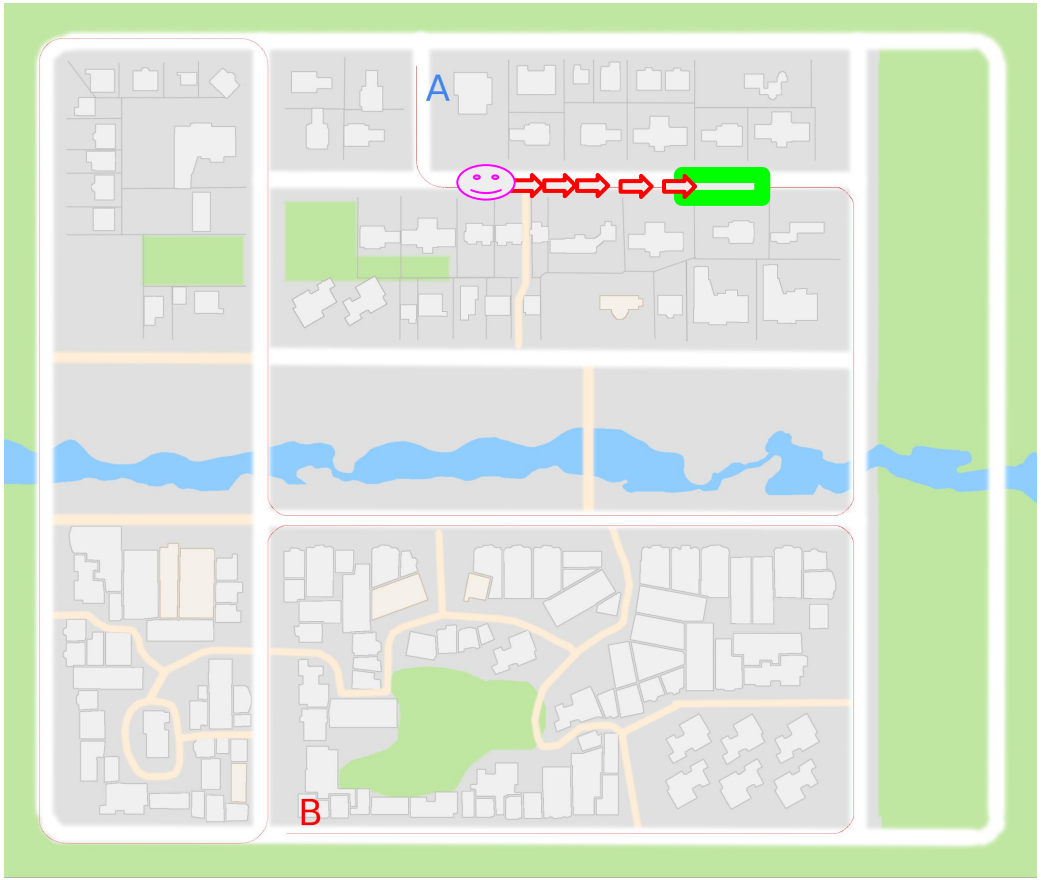


Background - The Big Picture of the Solution

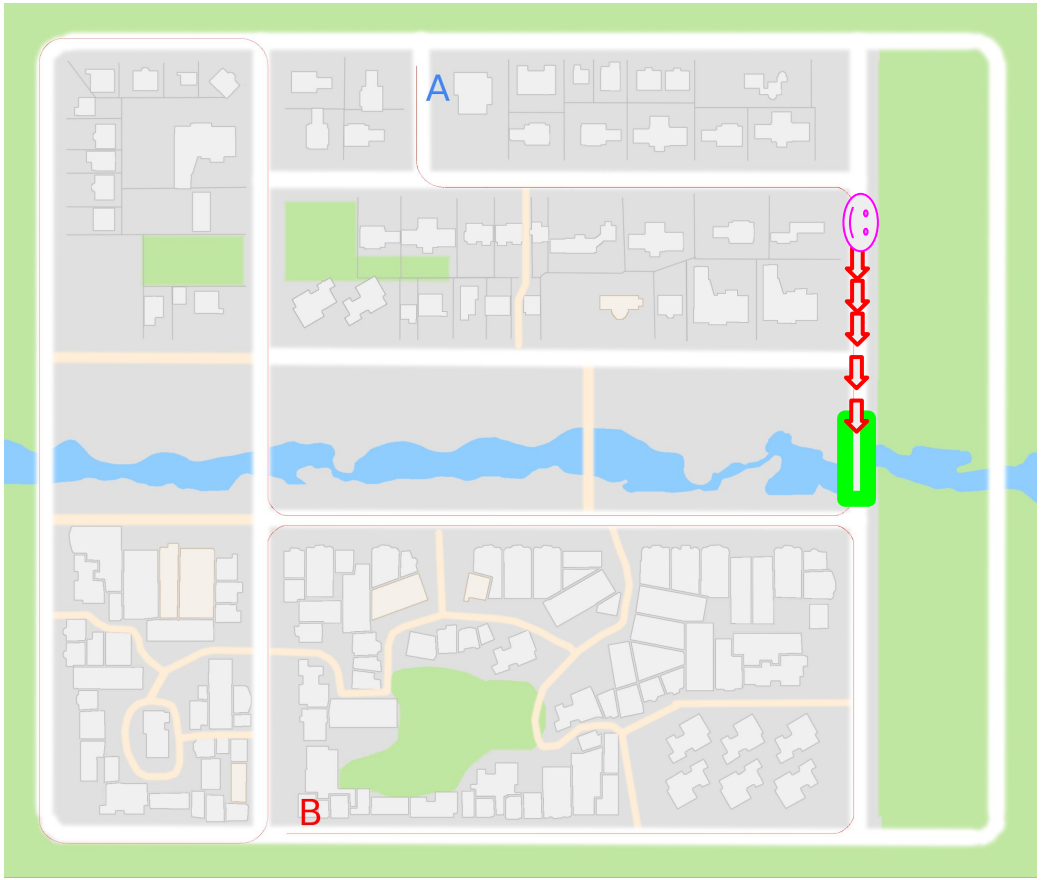


Background - Trajectory Planning

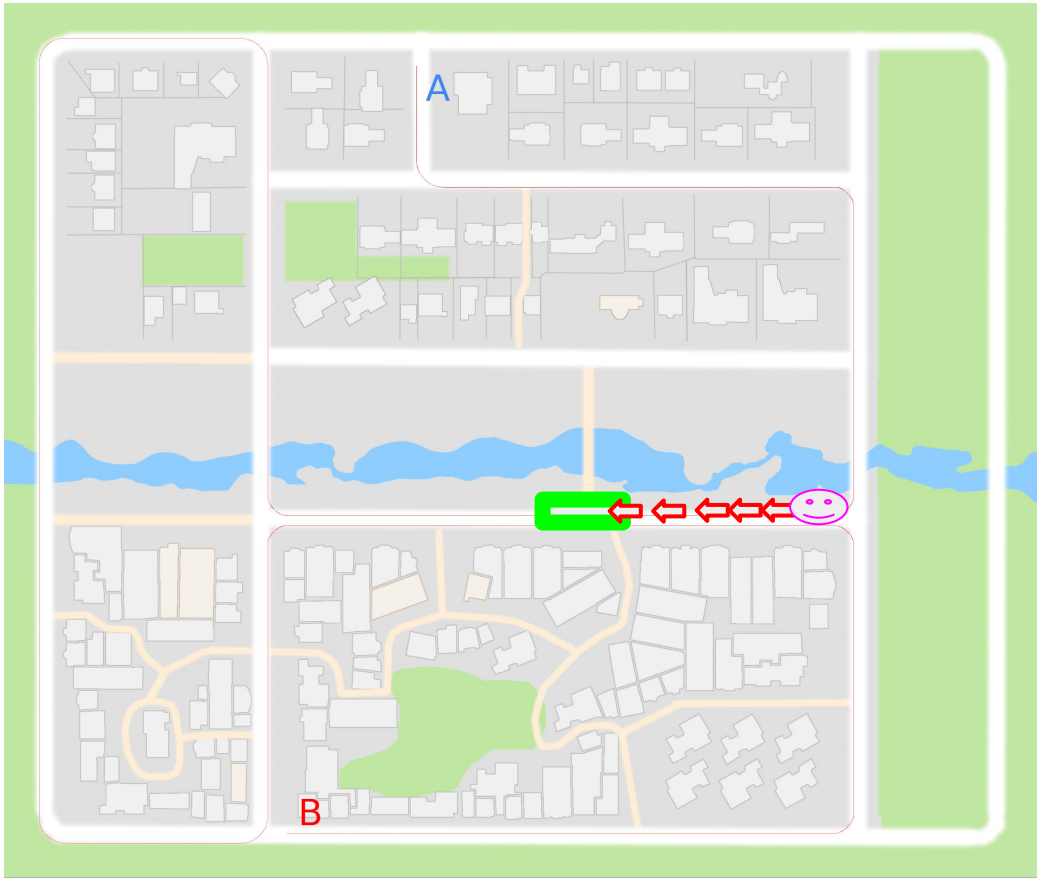
Background - Trajectory Planning



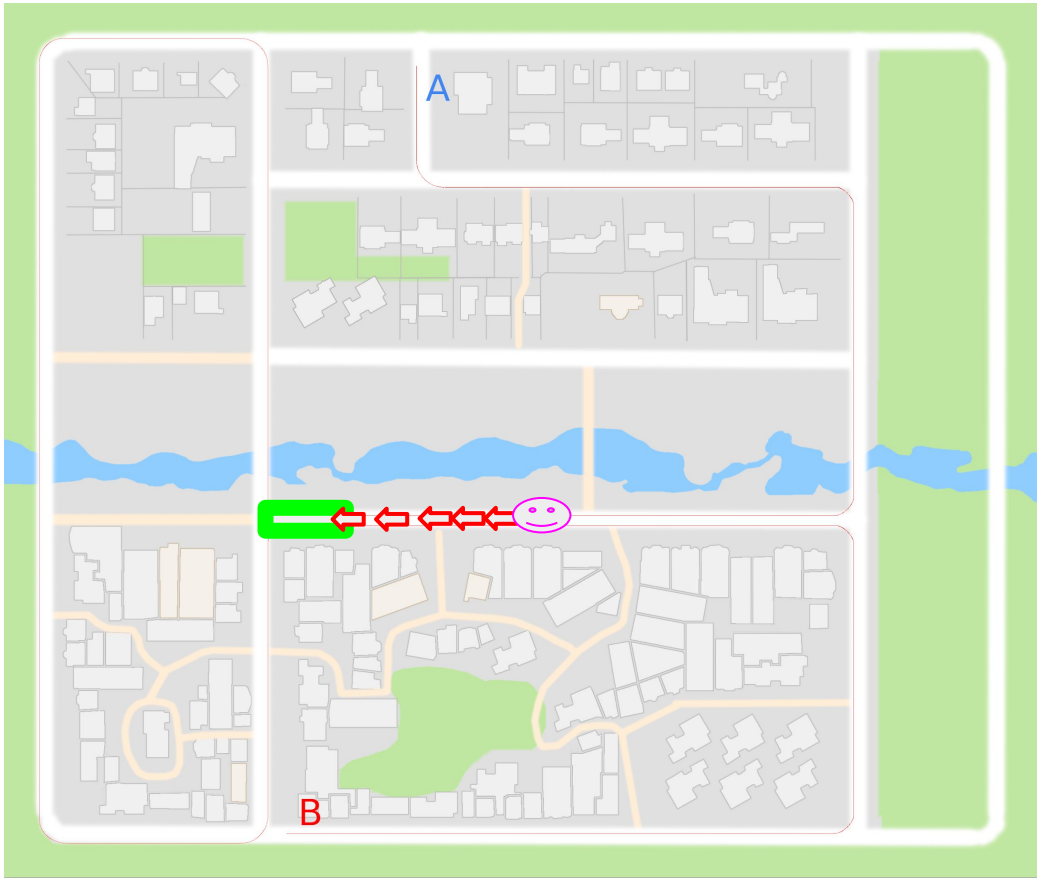
Background - Trajectory Planning



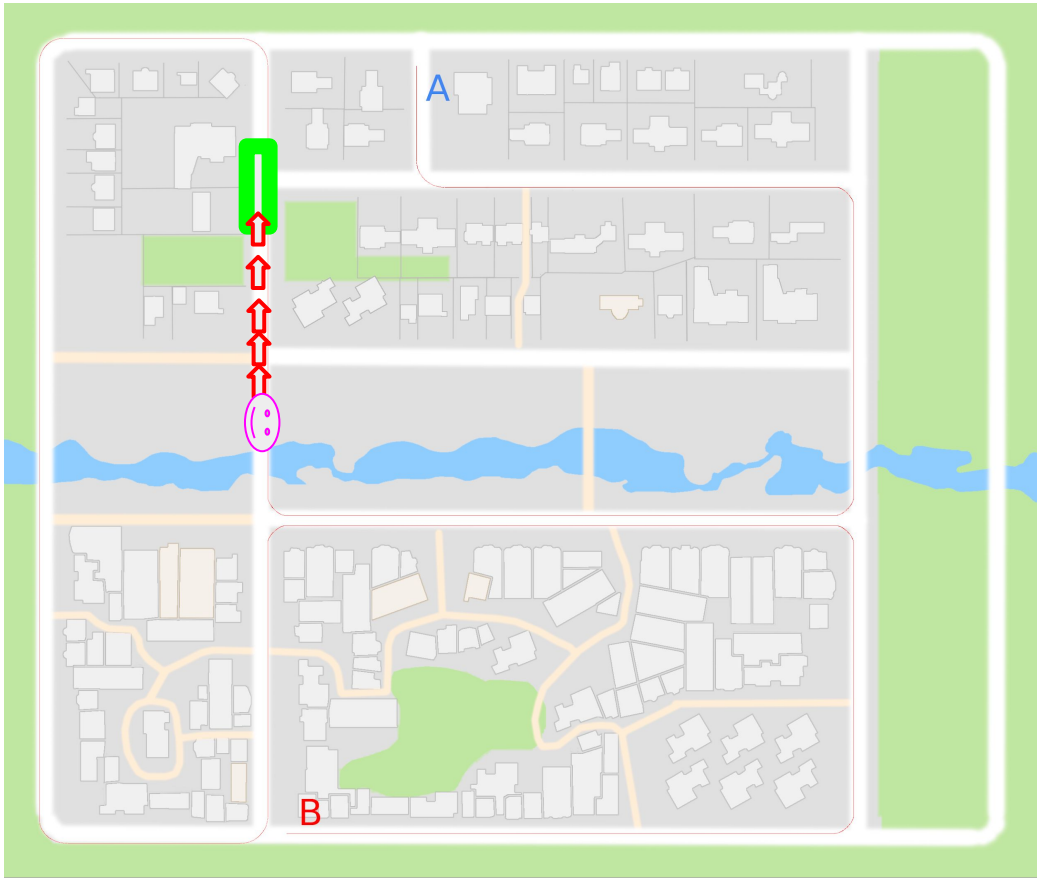
Background - Trajectory Planning



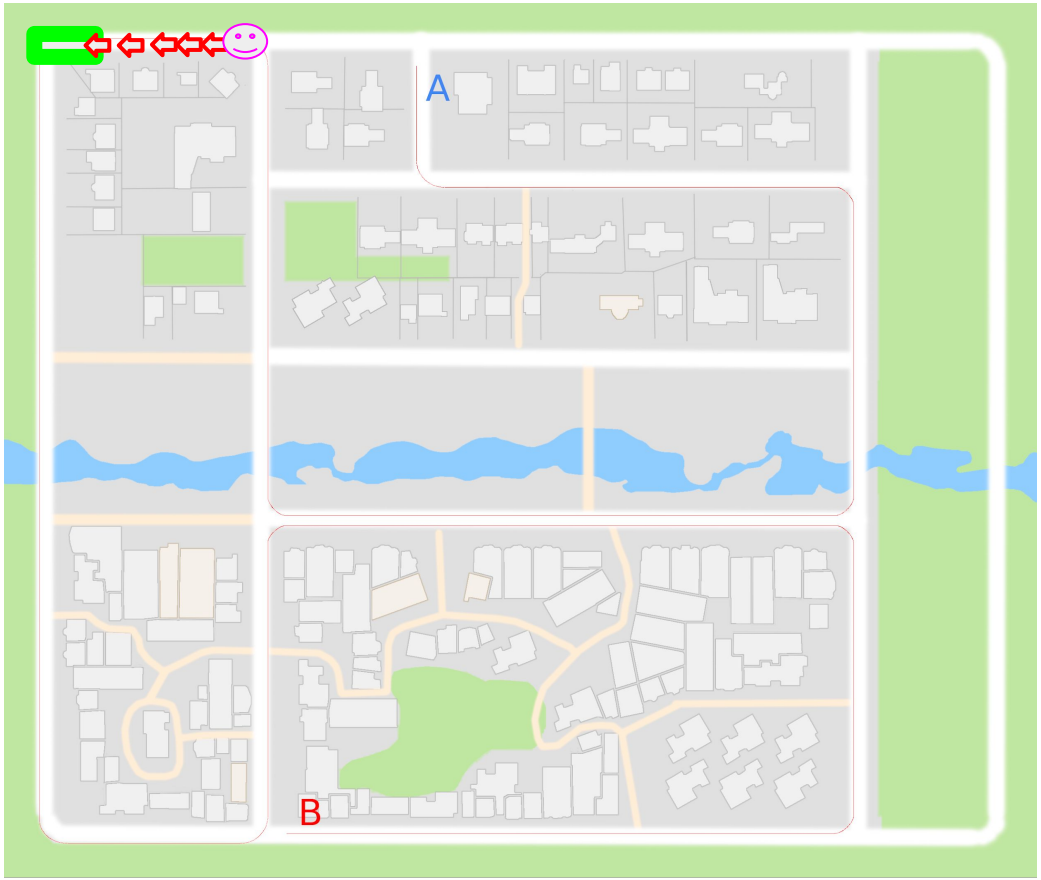
Background - Trajectory Planning



Background - Trajectory Planning



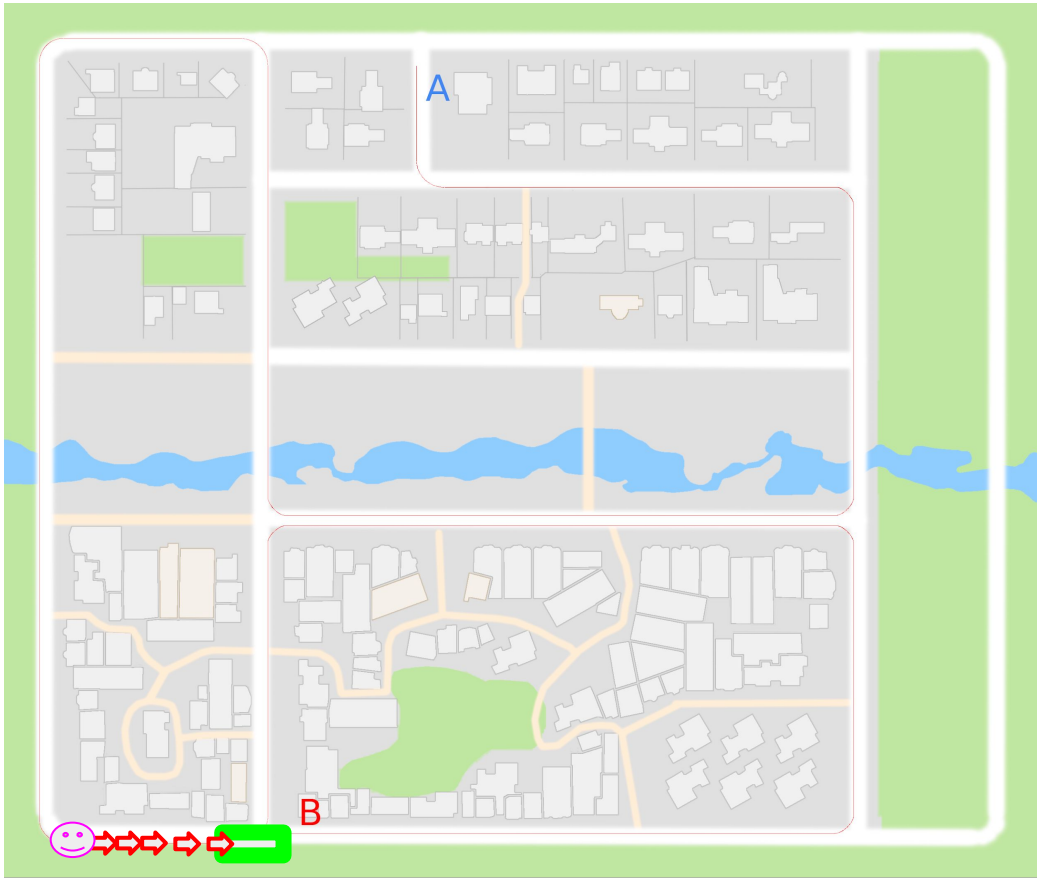
Background - Trajectory Planning



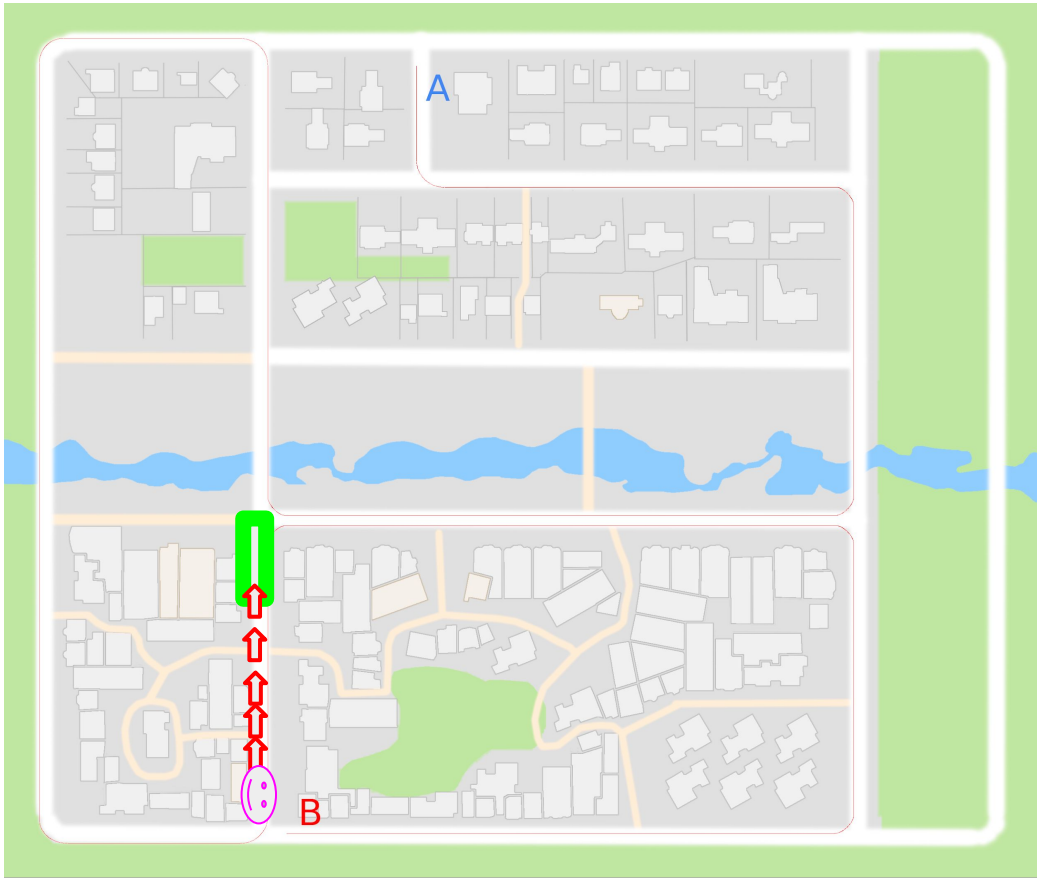
Background - Trajectory Planning

Background - Trajectory Planning

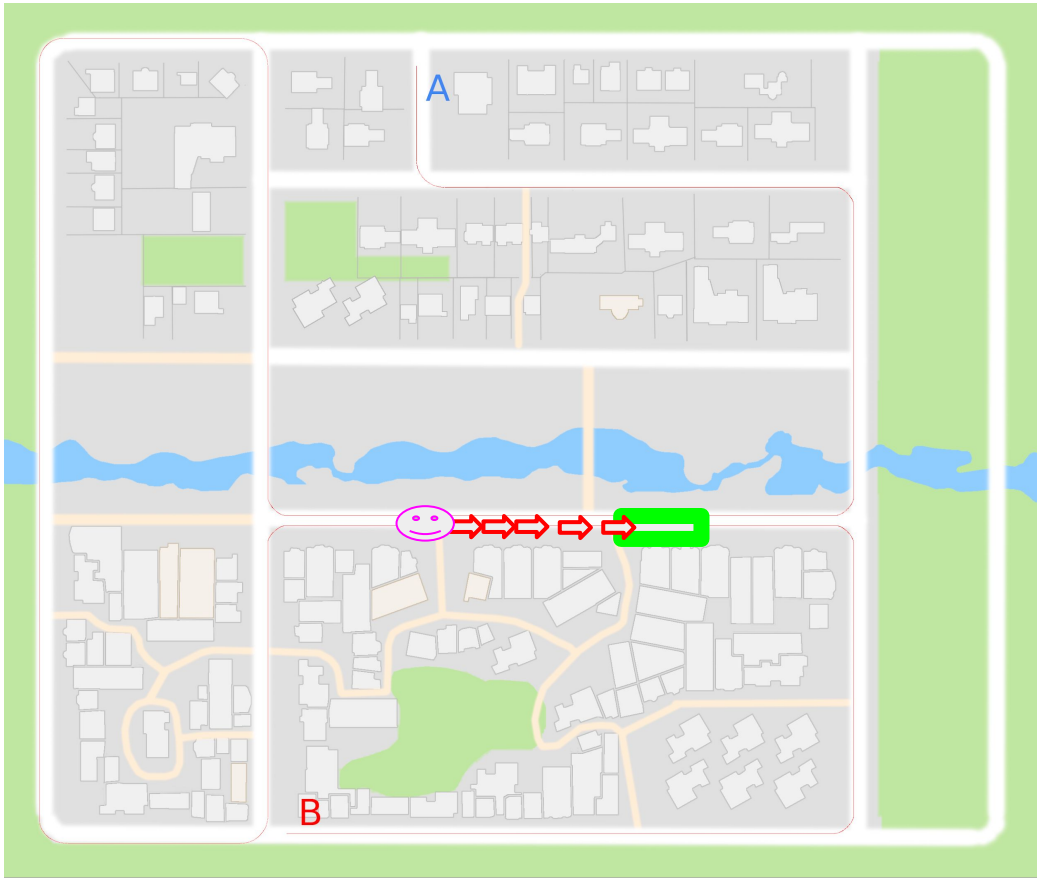
Background - Trajectory Planning



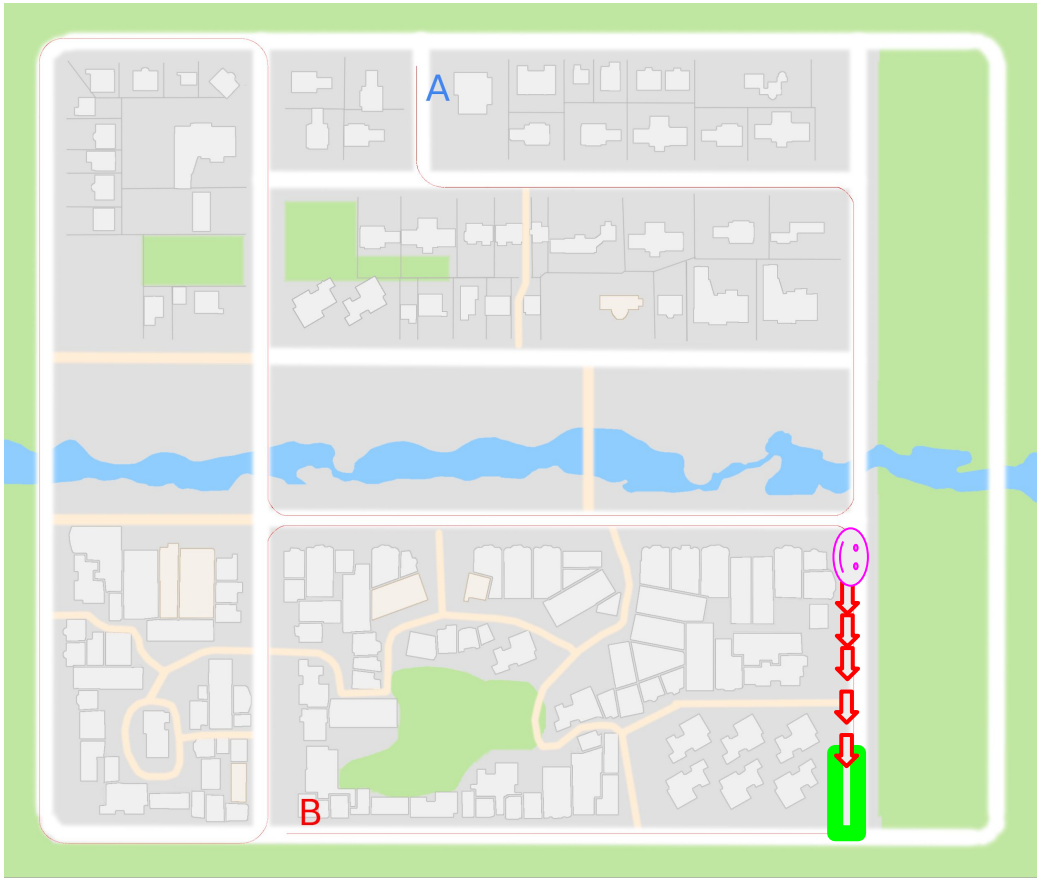
Background - Trajectory Planning



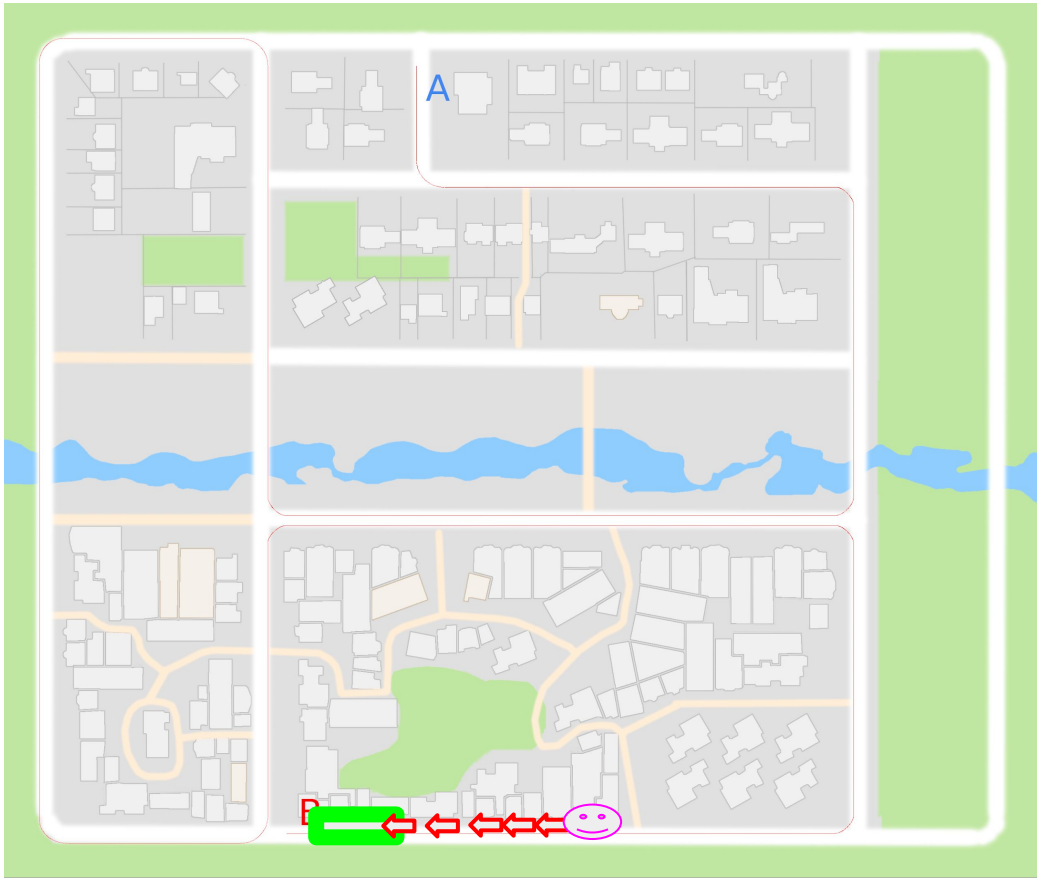
Background - Trajectory Planning



Background - Trajectory Planning



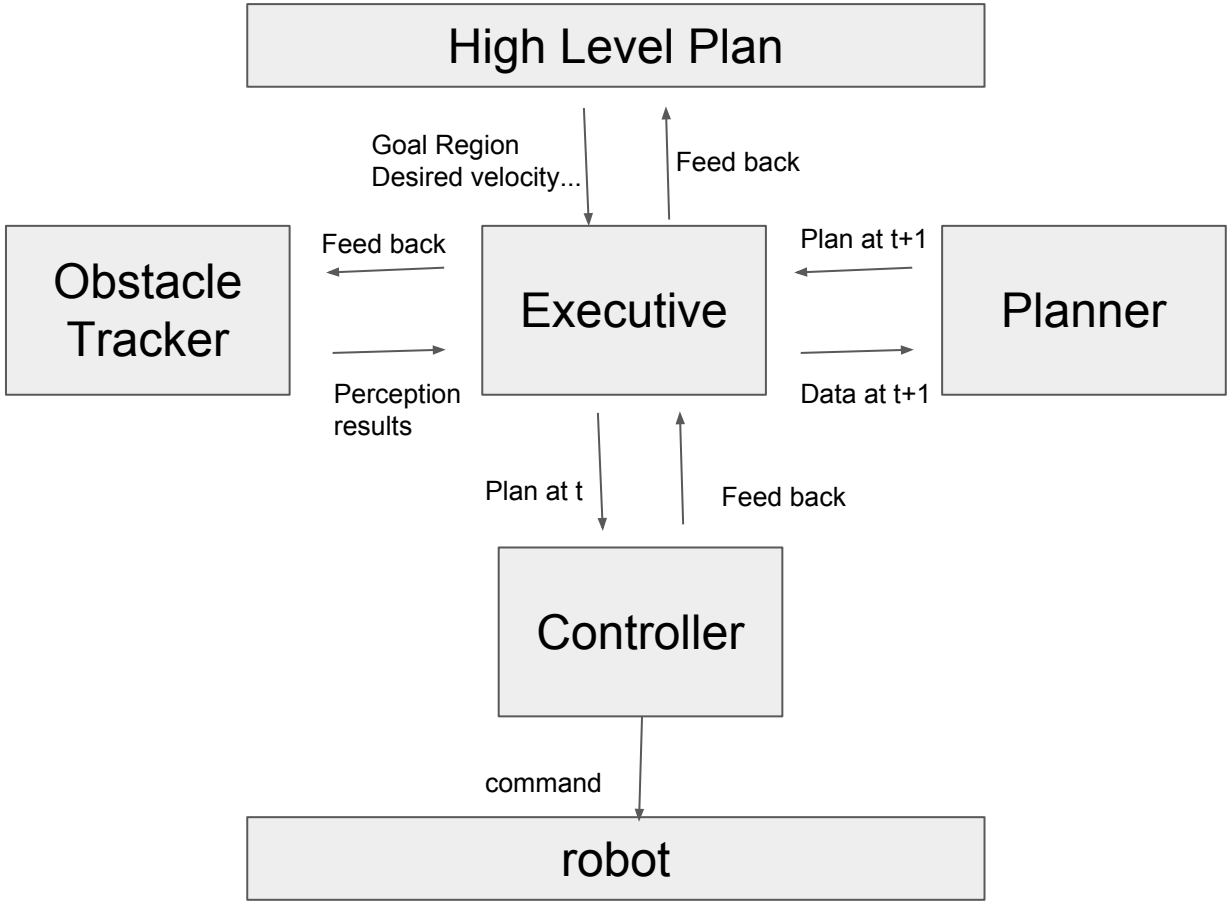
Background - Trajectory Planning



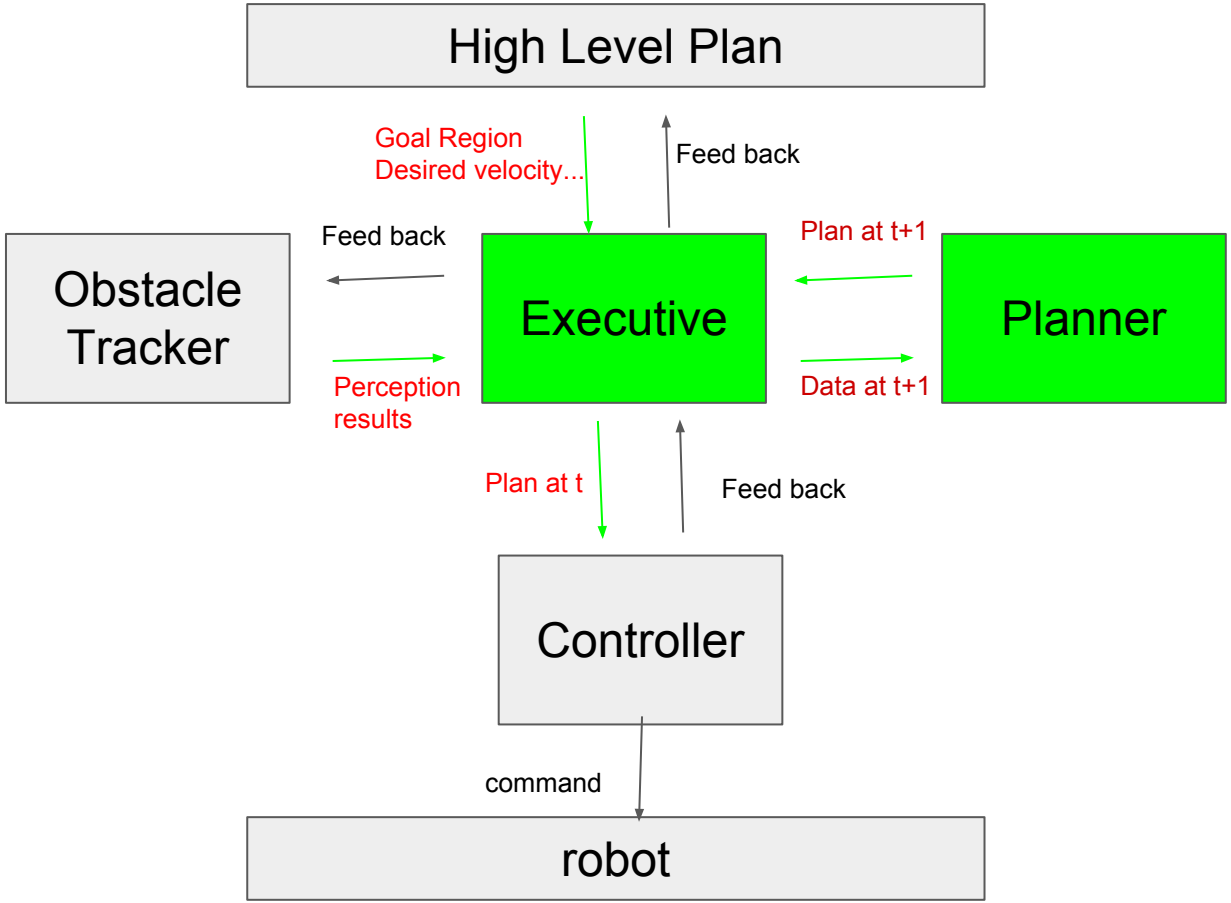
Online Real-time Framework

1. Executive
2. Obstacle tracker
3. Planner
4. Controller

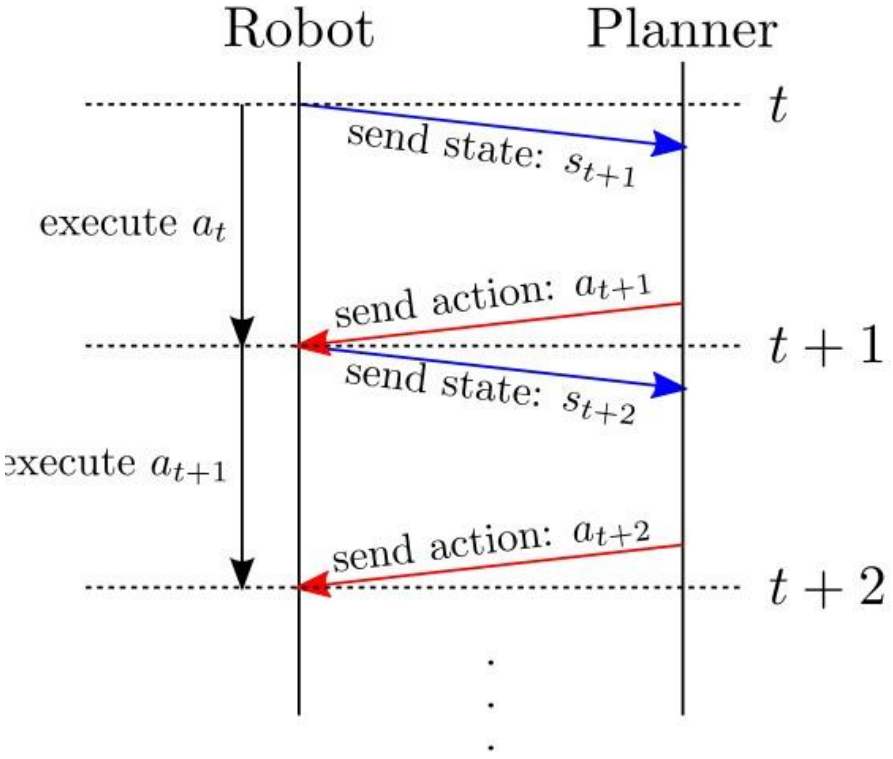
Online Real-time Framework



Online Real-time Framework



Online Real-time Framework



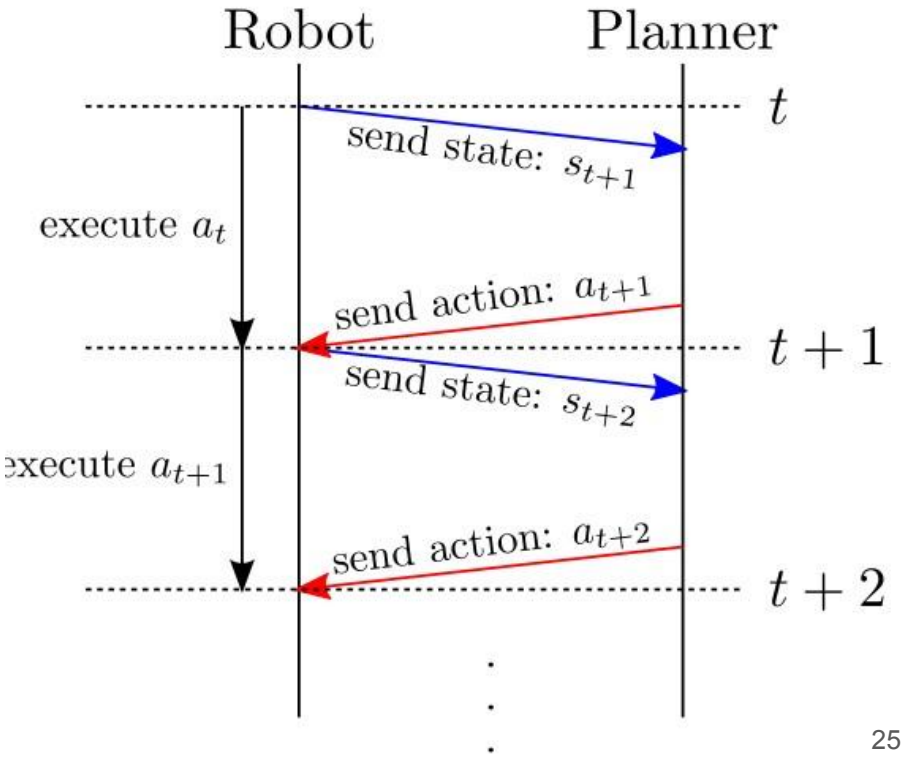
Executive

Get all the data

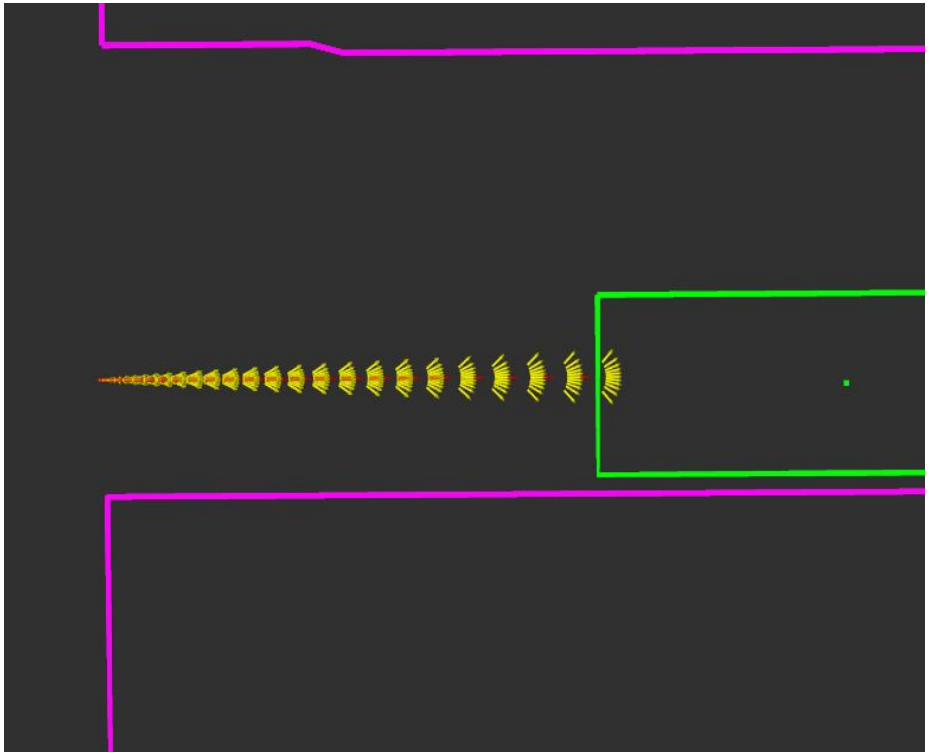
Validate the plan

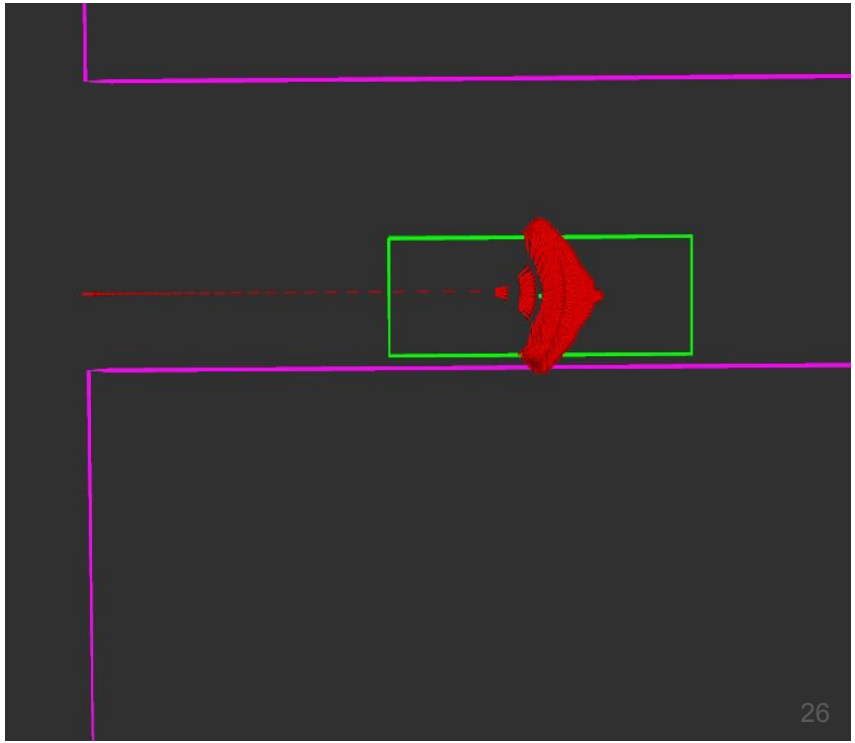
Publish plan

Send Plan request to Planner



Online Lattice Based Planner





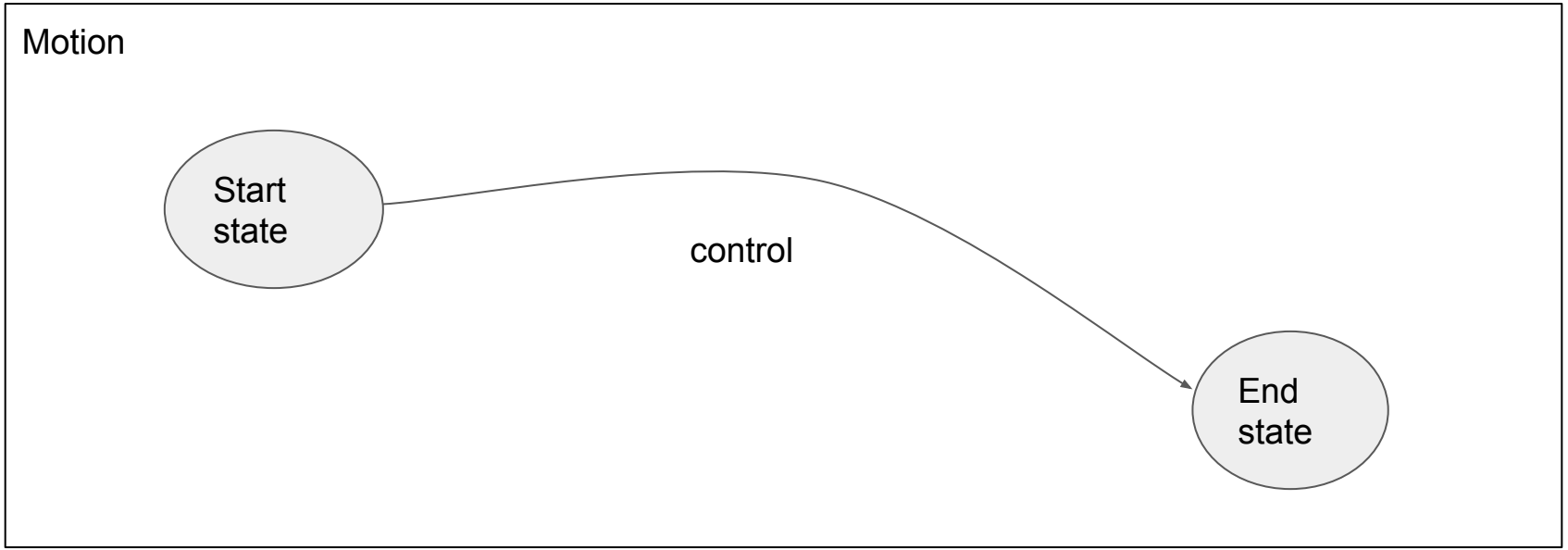
Online Lattice Based Planner

State (x, y, theta, speed, time)

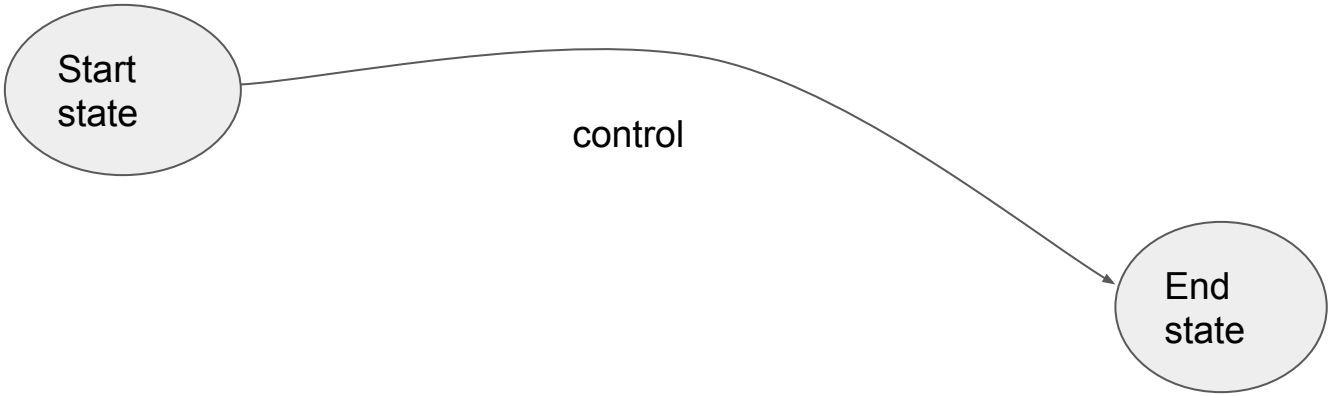
Control (acceleration, steering)

Motion (start state, end state, control)

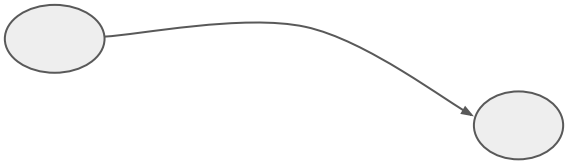
Online Lattice Based Planner



Online Lattice Based Planner

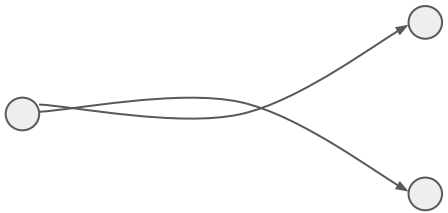


Online Lattice Based Planner

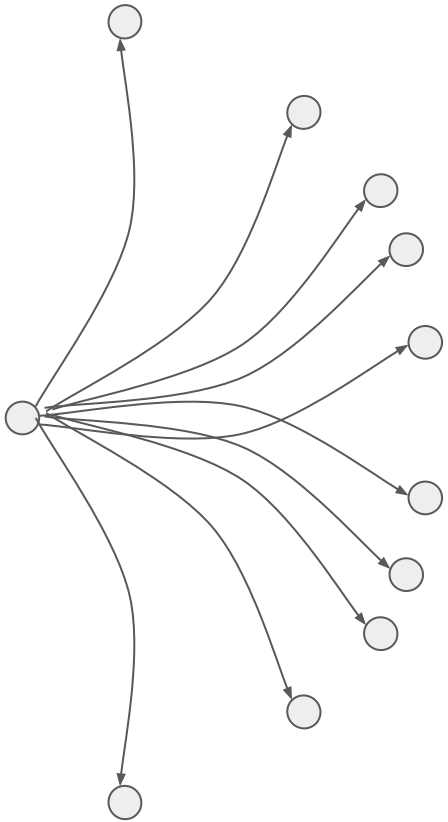


Online Lattice Based Planner

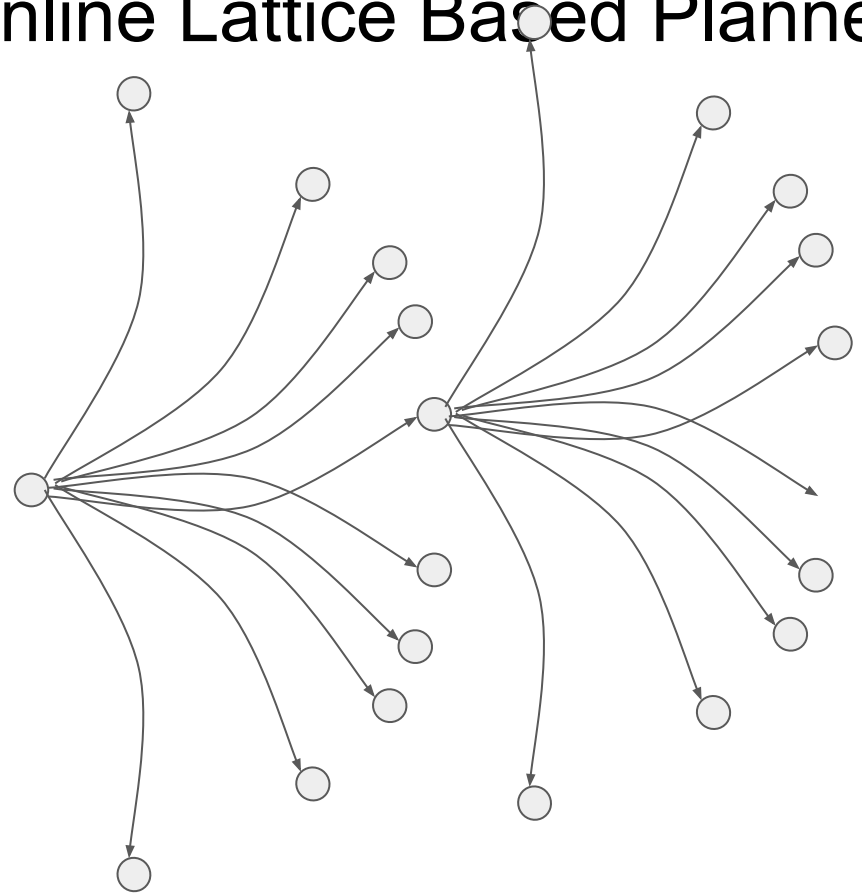
Online Lattice Based Planner



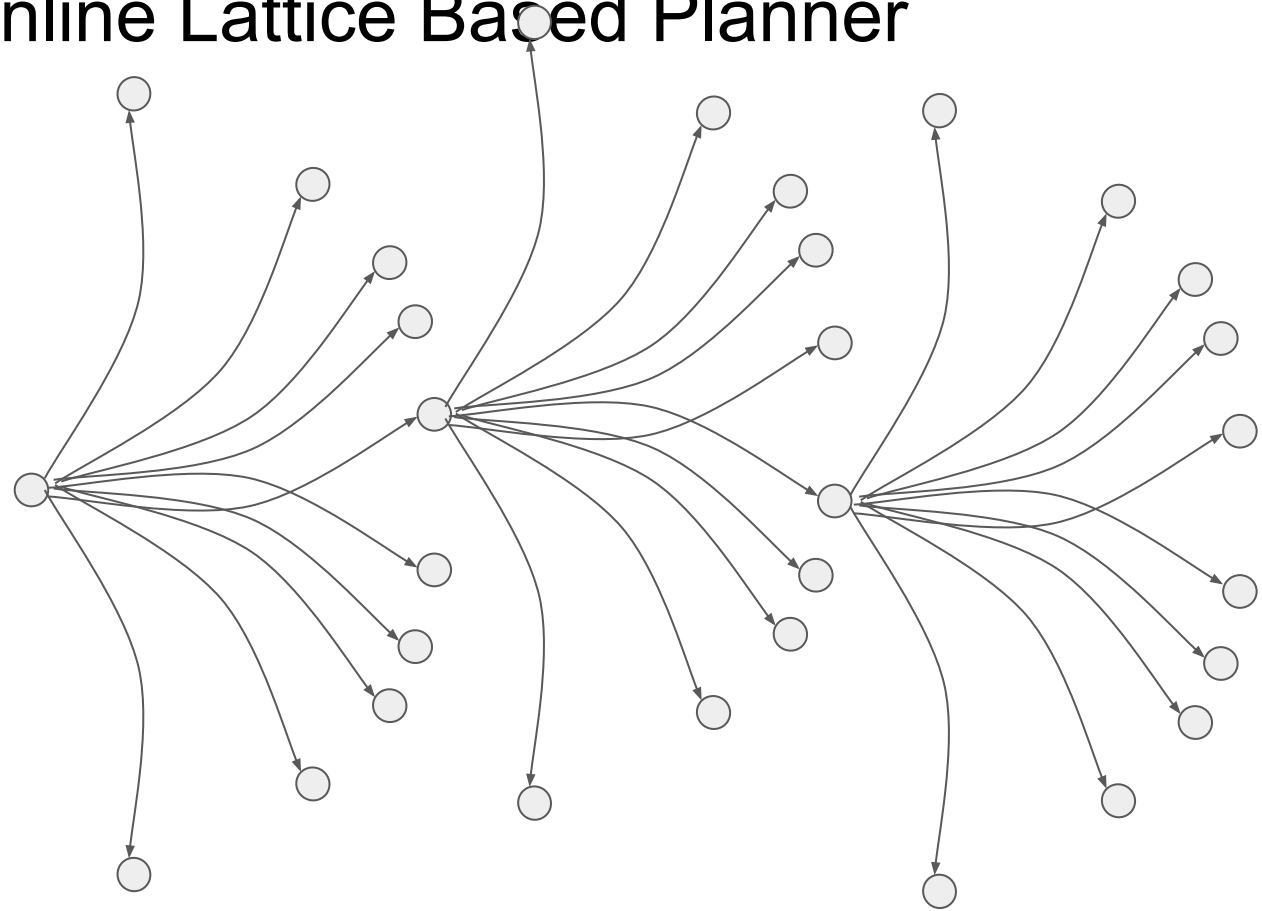
Online Lattice Based Planner



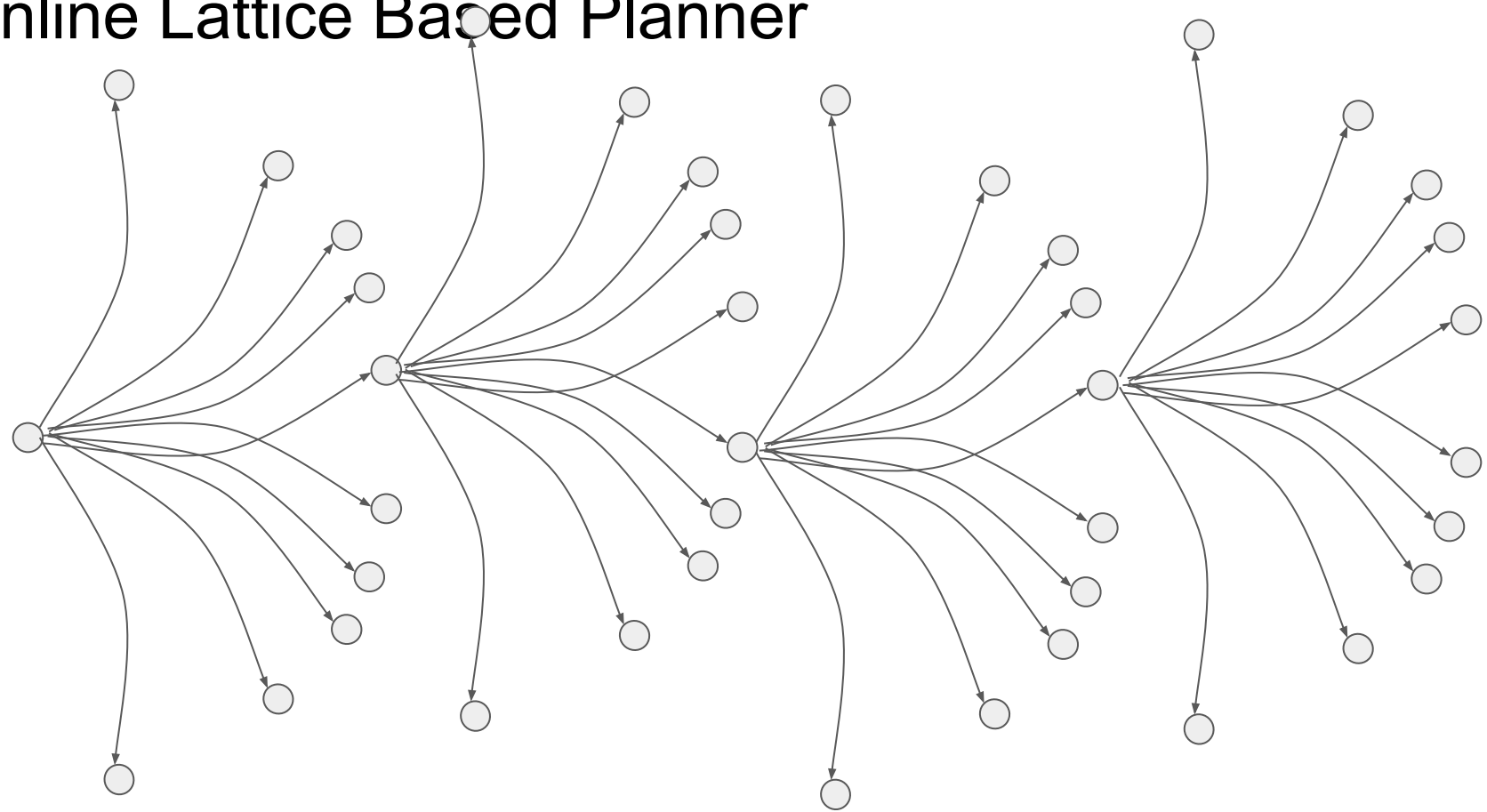
Online Lattice Based Planner



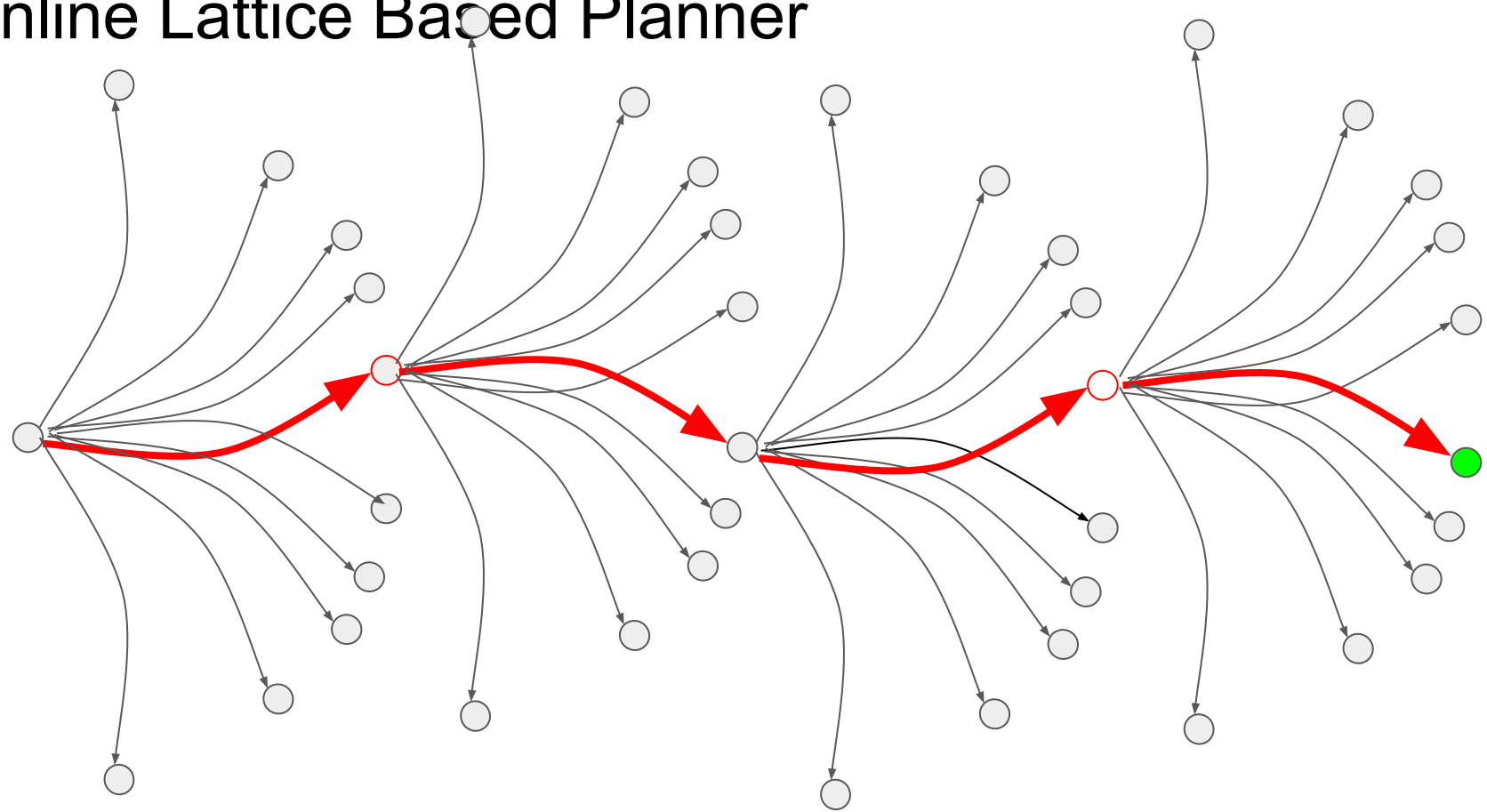
Online Lattice Based Planner



Online Lattice Based Planner

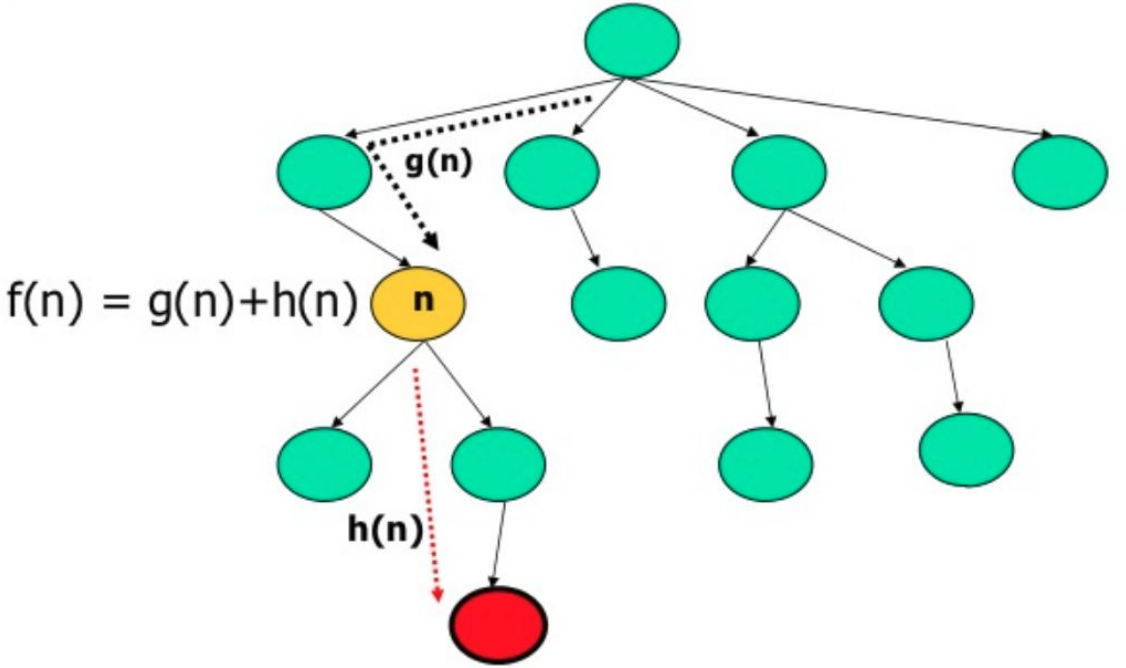


Online Lattice Based Planner



Online Lattice Based Planner

Astar search



Online Lattice Based Planner

Astar search

- f(n) = g(n) + h(n)
- g(n) = g(n-1) + Cost(n-1, n)
- Cost(n-1, n) = T(n-1,n) + w1 * CollisionCost
- CollisionCost = StaticCost + DynamicCost * CollideProbability
- h(n) = heuristic cost to goal

Online Lattice Based Planner

- Compound Heuristic
- Collision Checker
- Goal Checker

Online Lattice Based Planner

- Compound Heuristic
  - Straight path term
  - Velocity term
  - Orientation term
  - hc = max(hp, hv) + ho
- Collision Checker
- Goal Checker

Online Lattice Based Planner

- Compound Heuristic
- Collision Checker
- Goal Checker

Online Lattice Based Planner

- Compound Heuristic
- Collision Checker
  - Static obstacles
  - Dynamic obstacles
- Goal Checker

Online Lattice Based Planner

- Compound Heuristic
- Collision Checker
  - Static obstacles
    - Cost 1000
  - Dynamic obstacles
- Goal Checker

Online Lattice Based Planner

- Compound Heuristic
- Collision Checker
  - Static obstacles
    - Cost 1000
  - Dynamic obstacles
    - Cost 2000
    - Cost = Cost * P(col)
    - $$P(col) = 1 - P(\overline{col}) = 1 - \prod_{i=0}^{k} \prod_{j=0}^{n} (1 - P(col)_j^i)$$
- Goal Checker

Online Lattice Based Planner

- Compound Heuristic
- Collision Checker
- Goal Checker

Online Lattice Based Planner

- Compound Heuristic
- Collision Checker
- Goal Checker
  - Position is inside goal region
  - Velocity is equal desired velocity

Online Lattice Based Planner

Two Level Anytime Search with Fixed Horizon

- Goal Checker
- Time Heuristic

Online Lattice Based Planner

Two Level Anytime Search with Fixed Horizon

- Goal Checker
  - Reward 0, if not achieve time horizon
  - Reward 1, if achieve time horizon but does not satisfy goal condition
  - Reward 1000, if achieve time horizon and satisfy goal condition
- Time Heuristic

Online Lattice Based Planner

Two Level Anytime Search with Fixed Horizon

- Goal Checker
- Time Heuristic
  - ht(n) = time to goal（= hops to goal）
  - h = (1-w2)*ht + w2 * hc, 0<=w<=1

Demo Video

- Wait

  https://www.youtube.com/watch?v=3bs2jMOW628

- Slow down

  https://www.youtube.com/watch?v=C6MRnaQc3cE

- Get around

  https://www.youtube.com/watch?v=BVBUR7rDsoo

- Traffic light & Turn (0:41)

  https://www.youtube.com/watch?v=I3NCW1qjXck

- Dangerous solution

  https://www.youtube.com/watch?v=n7mC_P9b11w

Challenges

- Did not work at first place, hard to locate the issue
  - Build a visualizer
- Slow collision checker limit the number of look ahead of the planner (50)
  - Parallel the expansion (100)
  - Optimize code and reduce poly vertices (200)
  - Parallel intermediate state check (does not work well)
  - Approximate collision check (haven’t try)
  - Hierarchical collision check (haven’t try)
- Dangerous plan
  - Add a time window to each obstacle slide

…….

Precompute lattice (chip-based)

- 1000 HZ
- Multiple policy from dynamic obstacles could be take into account
- POMDP solution

Thank you!