Advance Discrete Planning (2)
Practical issues

Jane Li

Assistant Professor
Mechanical Engineering Department, Robotic Engineering Program
Worcester Polytechnic Institute

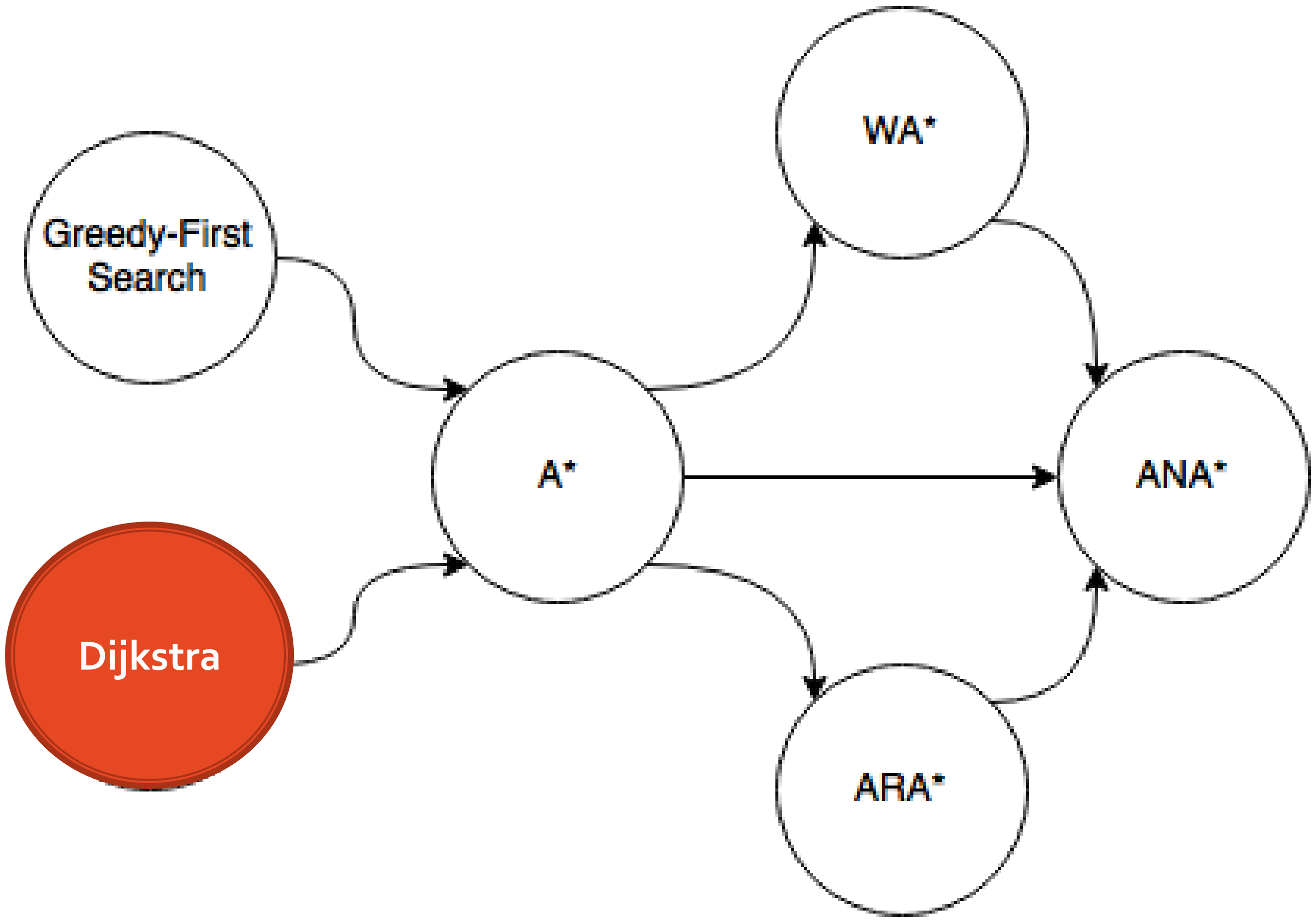
Quiz (10 pts)

- A search-algorithm prioritizes and expands the nodes in its open list items by their key values. How to compute the key values for
  - (2 pts) Dijkstra algorithm
  - (2 pts) A* algorithm
  - (3 pts) ARA*
  - (3 pts) ANA*

Dijkstra

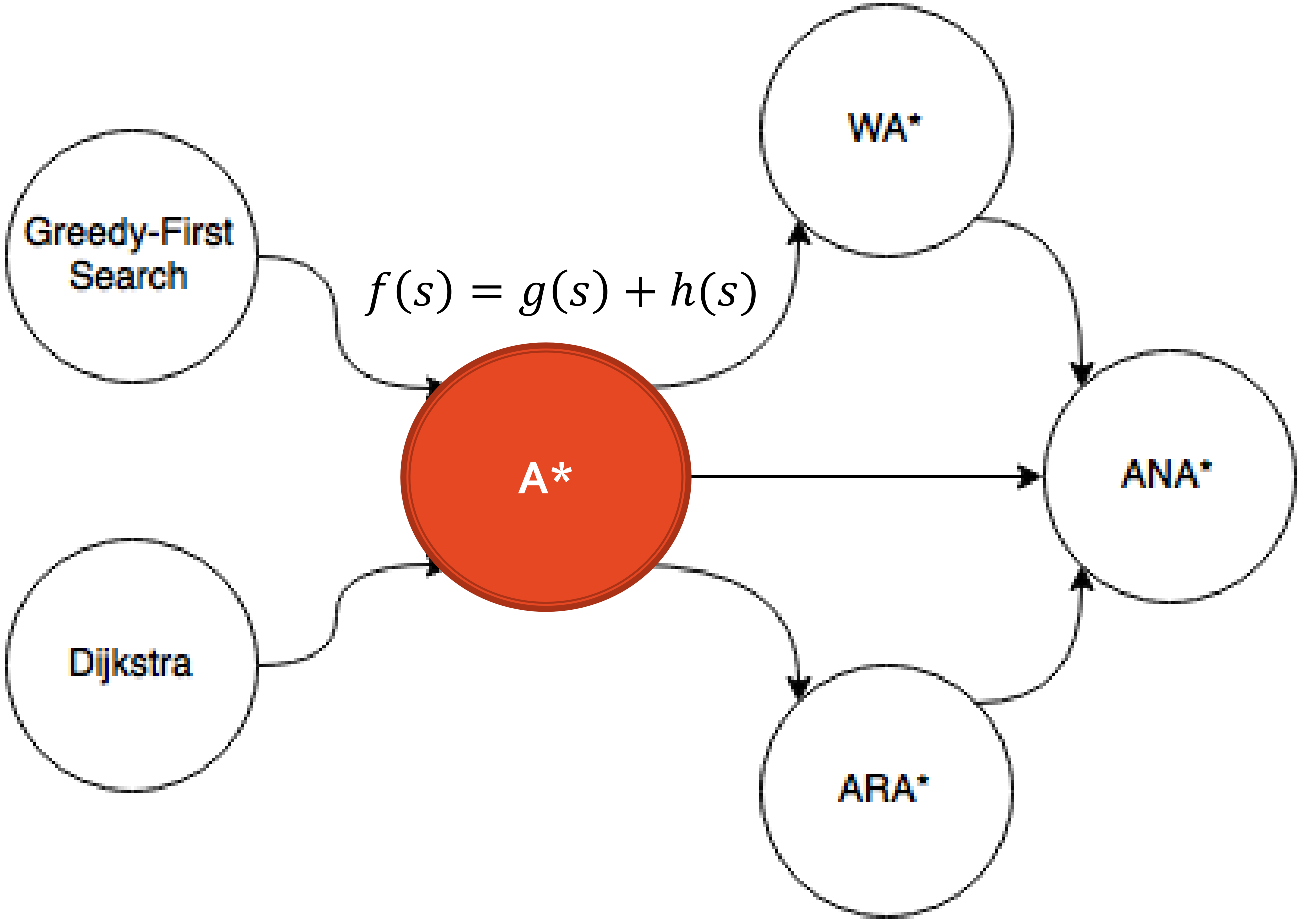
Optimality of Path to Goal

$f(s) = g(s)$

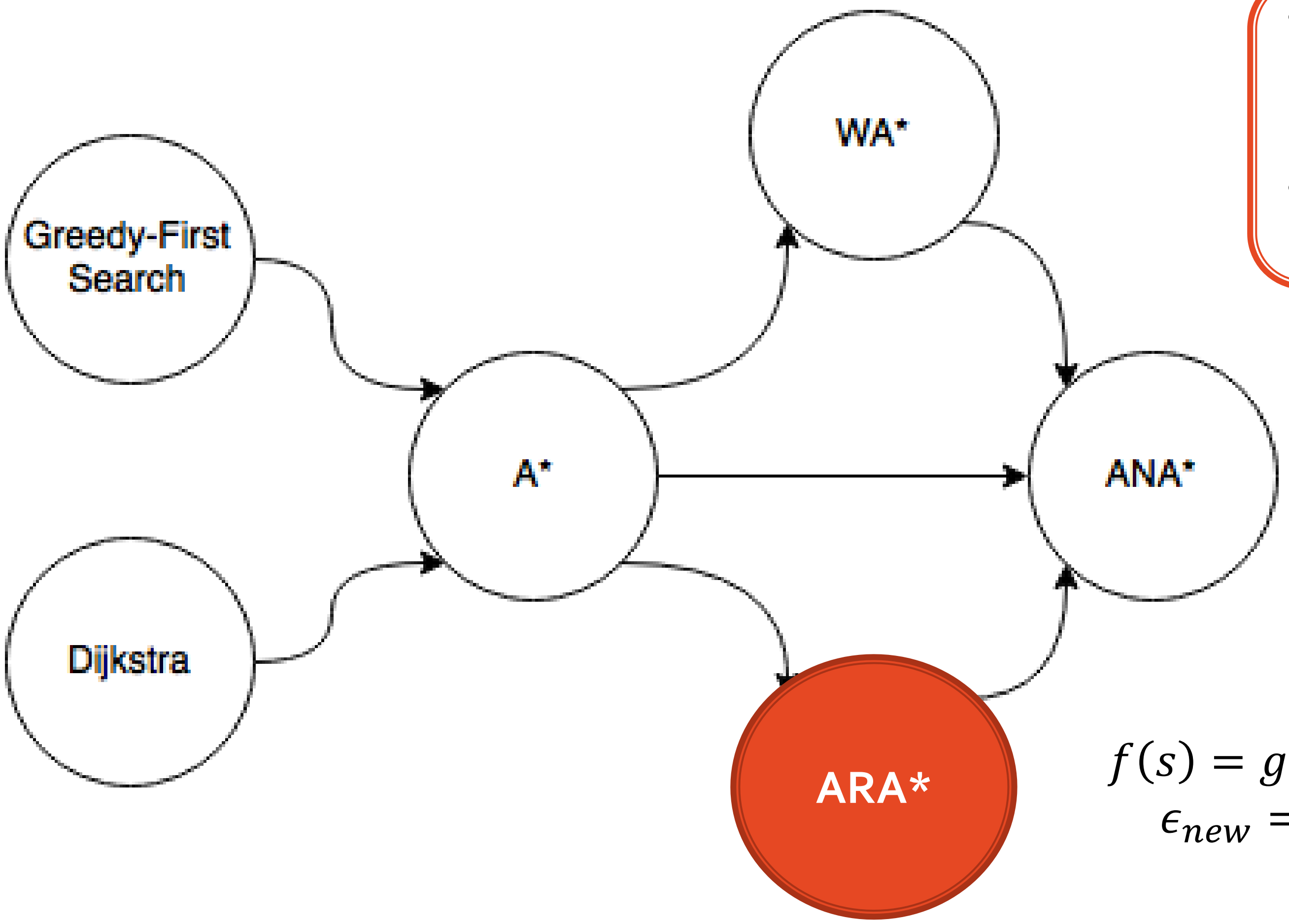


A*

- Converges Quickly to Optimal Path
- Heuristic Functions



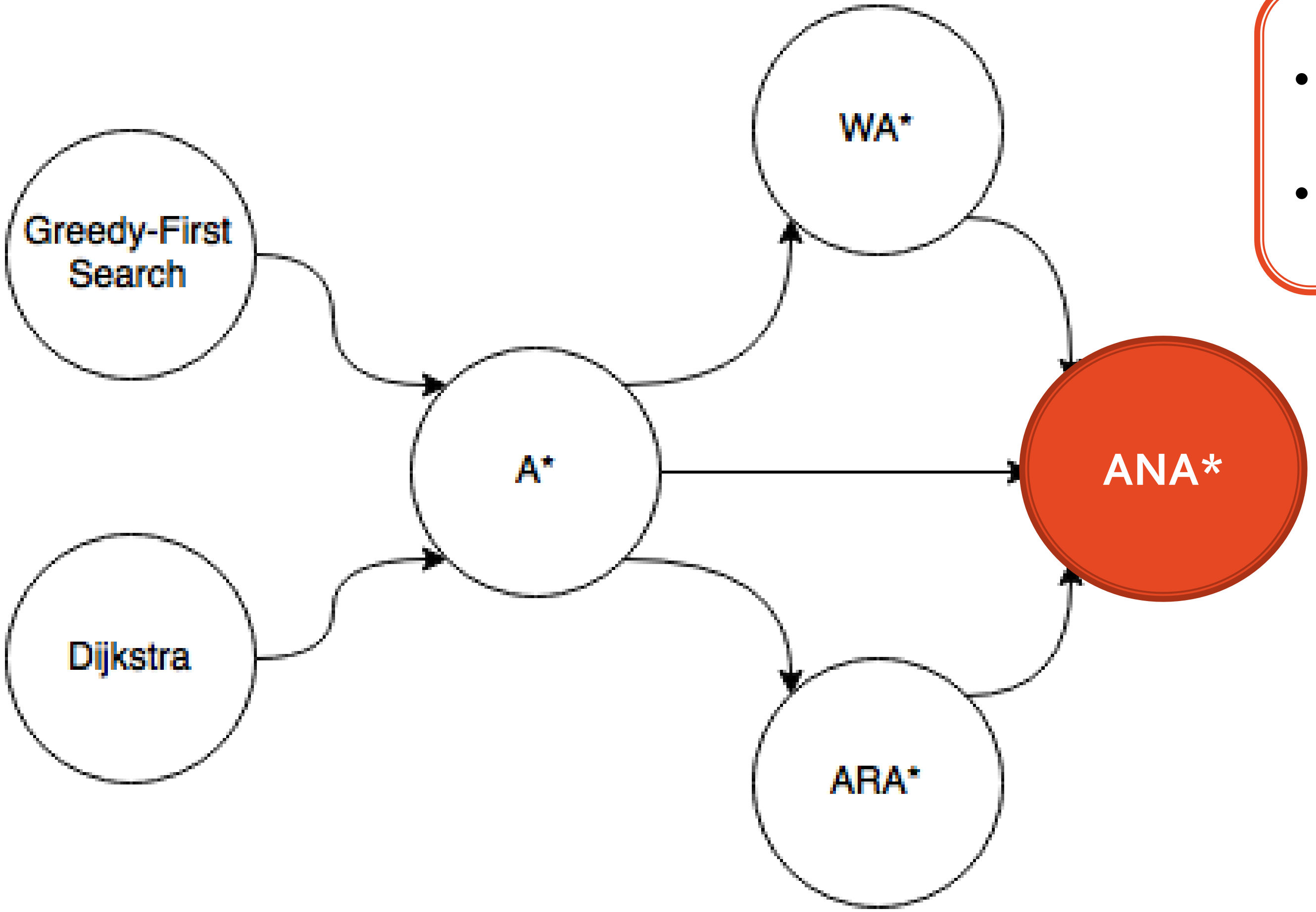
ARA*



- Issue WA*: Terminates with Sub-Optimal Solution
- ARA*: Iteratively reduces ϵ till Optimal Solution

$$f(s) = g(s) + \epsilon \cdot h(s)$$
$$\epsilon_{new} = \epsilon_{old} - \Delta\epsilon$$

ANA*



- **No Parameters**
- **Elegant choice of $\Delta\epsilon$ => Selected On-the-fly**

$$f(s) = g(s) + \epsilon \cdot h(s)$$

$$e(s) = \frac{G - g(s)}{h(s)}$$

$$\epsilon_{new} = \max_{\{s \in OPEN\}} e(s)$$

Advanced Discrete Motion Planning

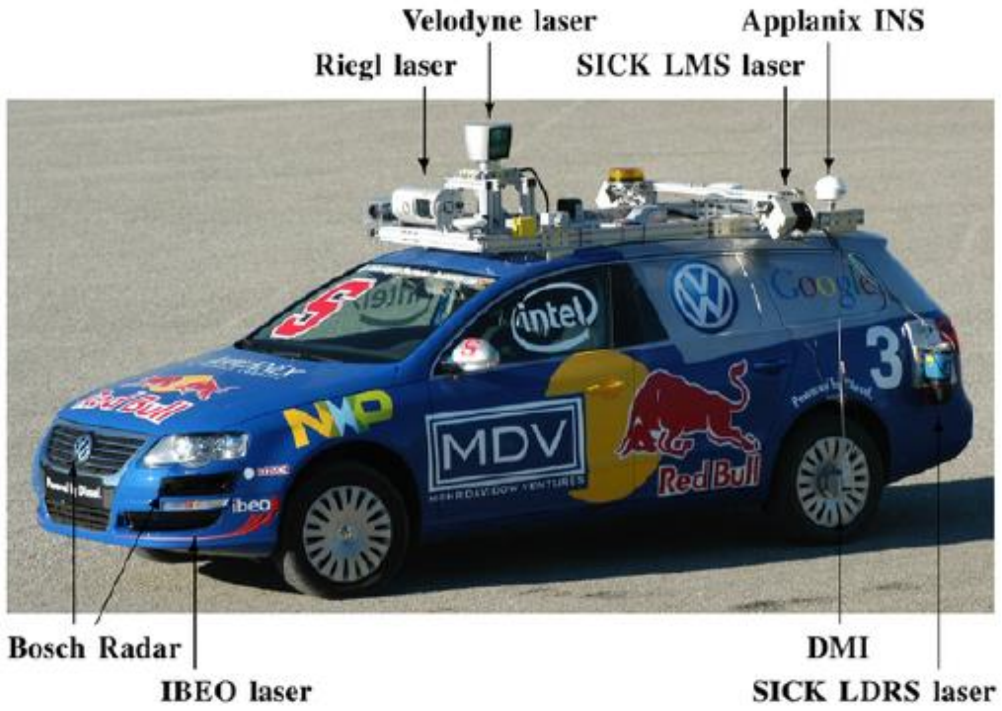
Overview of Advance Discrete Motion Planning

- More on search algorithms
  - Toward optimal and quicker solutions
  - Variants of A*
- Practical issues
  - Case study – Practical search techniques for autonomous driving
- Other advanced topics
  - Roadmaps for planning in dynamic environment
  - Learning search via imitation

Practical search techniques for autonomous driving

2007 DARPA Urban Challenge

- Task
  - Autonomous driving in unknown environment
  - Re-plan online while incrementally building an obstacle map



Stanford Racing Team
Junior in Urban Challenge (DARPA 2007)

2007 DARPA Urban Challenge

2007 DARPA Urban Challenge

- Capability – complex general path-planning tasks
  - Navigating parking lots
  - Executing U-turns
  - Dealing with blocked roads and intersections

- Performance
  - Typical full-cycle re-planning times of 50–300ms on a modern PC

Challenges

- Robot control and trajectories must be continuous
  - Continuous-variable optimization problem
- Existing search algorithms are fast in discrete state space
  - Produce non-smooth paths
  - Do not satisfy vehicle's non-holonomic constraints
- Other approaches?

Related work

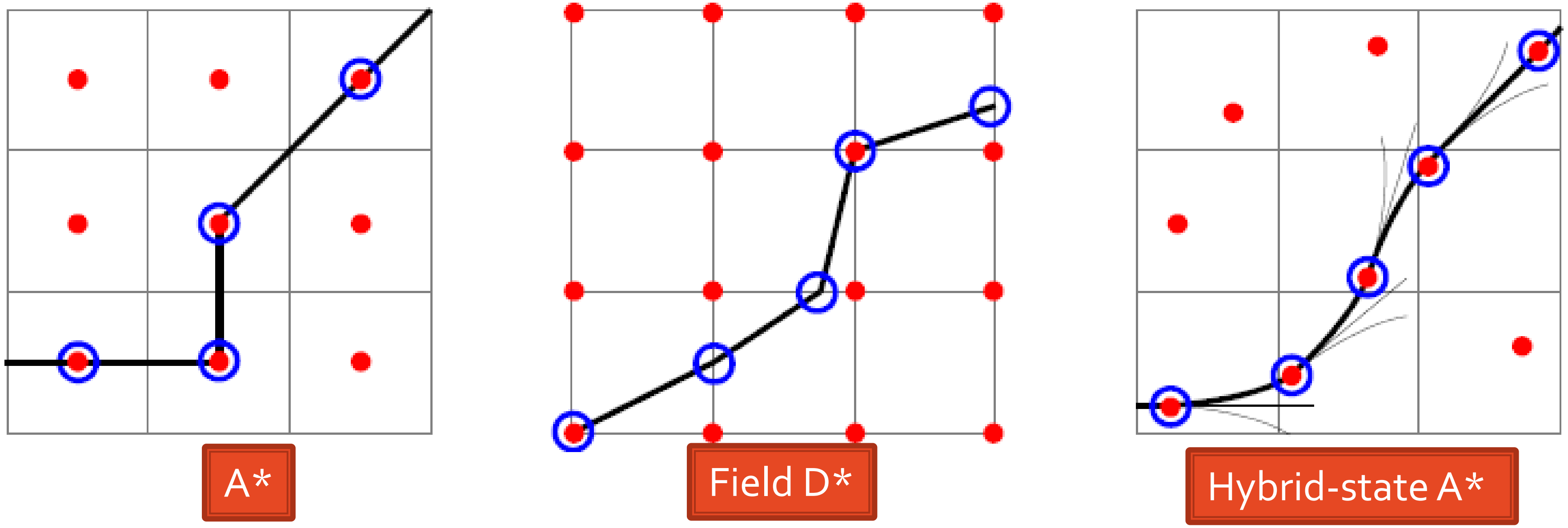
- Forward search in continuous coordinates
  - Guarantees kinematic feasibility
  - Need an efficient heuristic to guide expansion
- Non-linear optimization in control space
  - Fast convergence is difficult due to local minima

Solution – Two-phase motion planning

- Phase 1
  - Heuristic search in continuous coordinates to find a kinematically feasible solution → do not care optimality
- Phase 2
  - Use gradient descent to find at least the locally optimal solution
  - Global optimal is possible

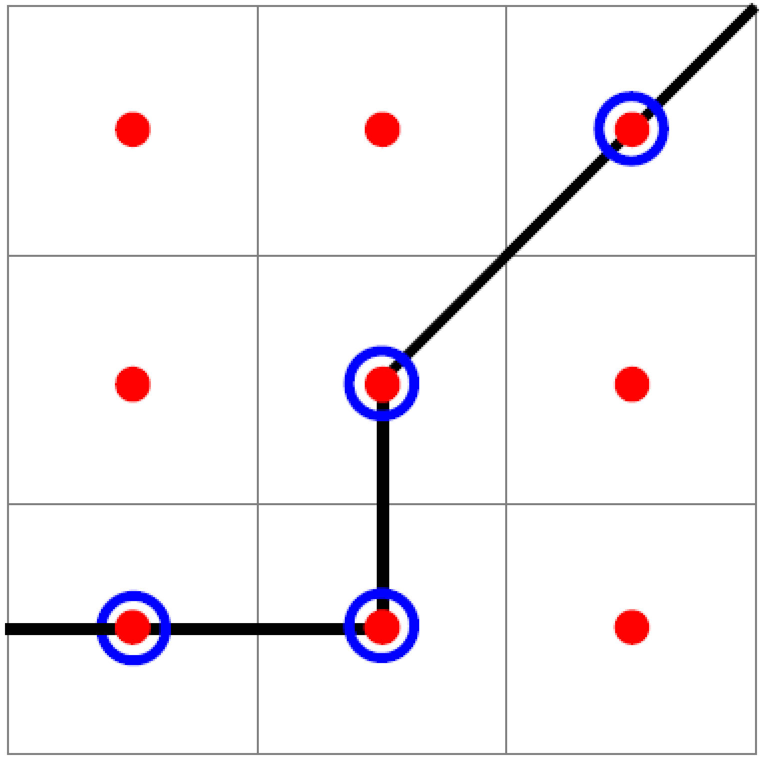
Hybrid-State A* Search

- Compared to A*
  - Search space (x, y, θ)
  - Associates with each grid cell a continuous 3D state of the vehicle



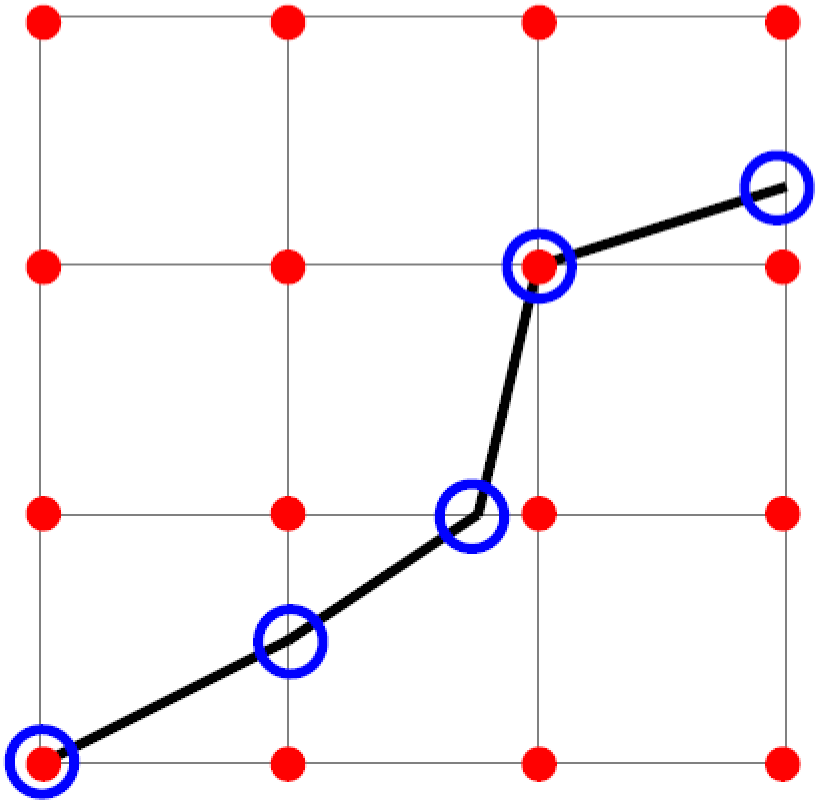
A*

- Features
  - Associates costs with cell centers
  - Only visit states at cell centers



Field D*

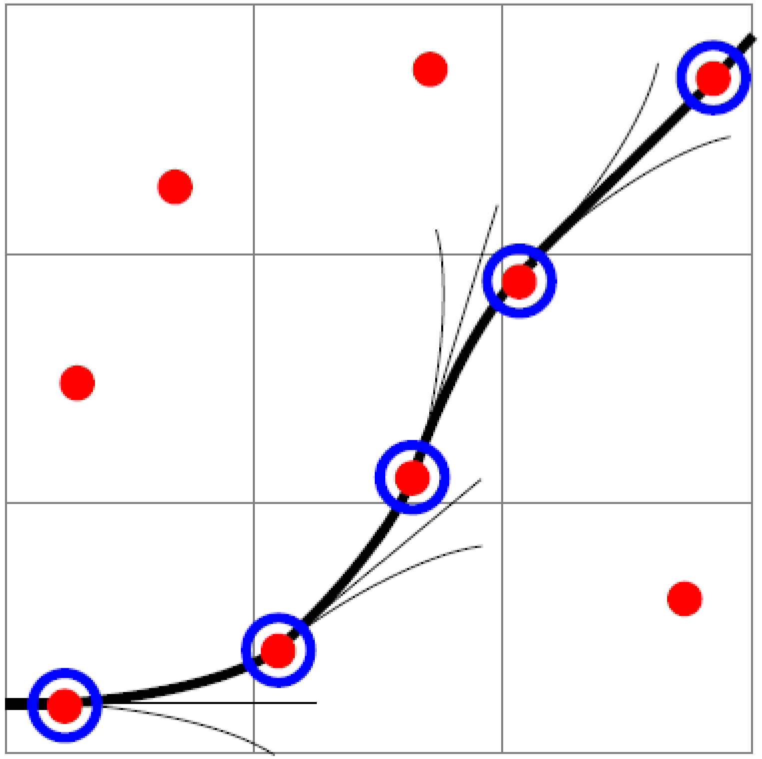
- Features
  - Associate costs with cell corners
  - Allow arbitrary linear paths from cell to cell



  - [Ferguson and Stentz 2005]

Hybrid-State A* Search

- Features
  - Associate a continuous state with each cell
  - Score of the cell = the cost of its associated continuous state



Performance

- Guaranteed to find the minimal-cost?
  - No
- Guarantee to be drivable?
  - Yes
- Sub-optimality w.r.t. global optimal solution?
  - In practice, hybrid-A* is close to global optimal solution → refine in Phase 2

Capability

- Can maneuver in tight spaces?
  - Like what?

- Plan forward and reverse motion?
  - Yes

Choice of Heuristic is Critical

- Which nodes to be expanded first
- Bad heuristics leads to wasted expansion

Heuristics

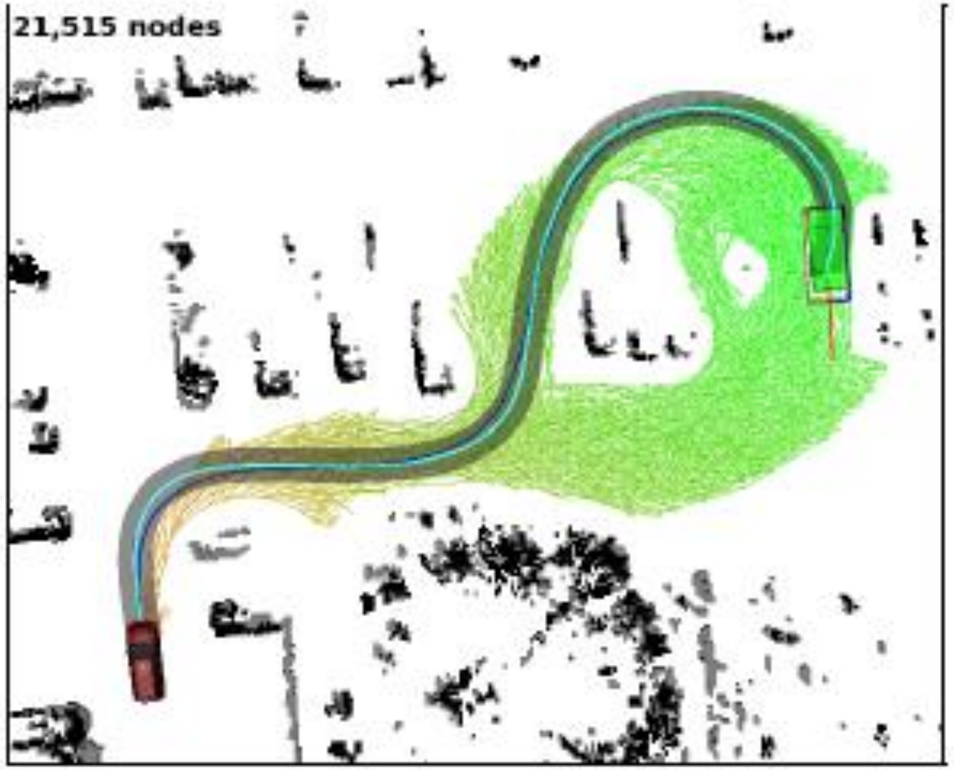
- h1(s) = Non-holonomic without obstacles
  - Ignores obstacles
  - Consider non-holonomic constraints
  - Admissible?

- Use the max of h1(s) and Euclidian distance – Benefits?
  - Prune search branches that approach the goal with the wrong headings
  - Computed offline

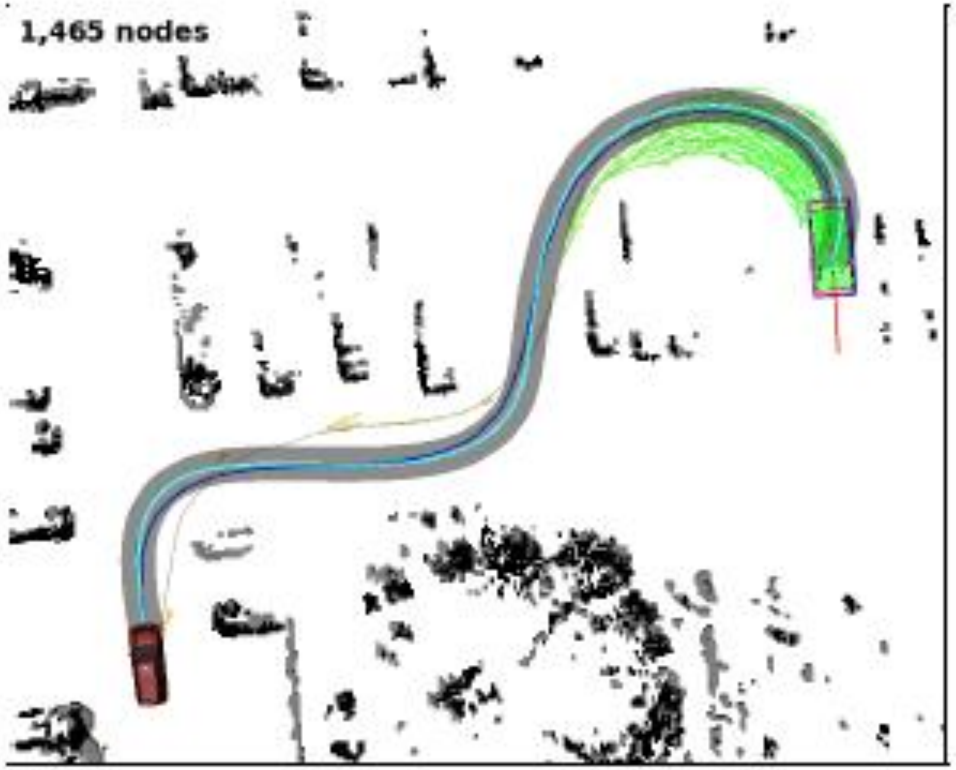
Heuristics

- h2(s) = dual of h1(s)
  - Consider obstacles
  - Ignore non-holonomic constraints
  - Admissible?
- Benefit?
  - Discover all U-shaped obstacles dead-ends in 2D
  - Guide the more expensive 3D search away from these areas

Heuristics

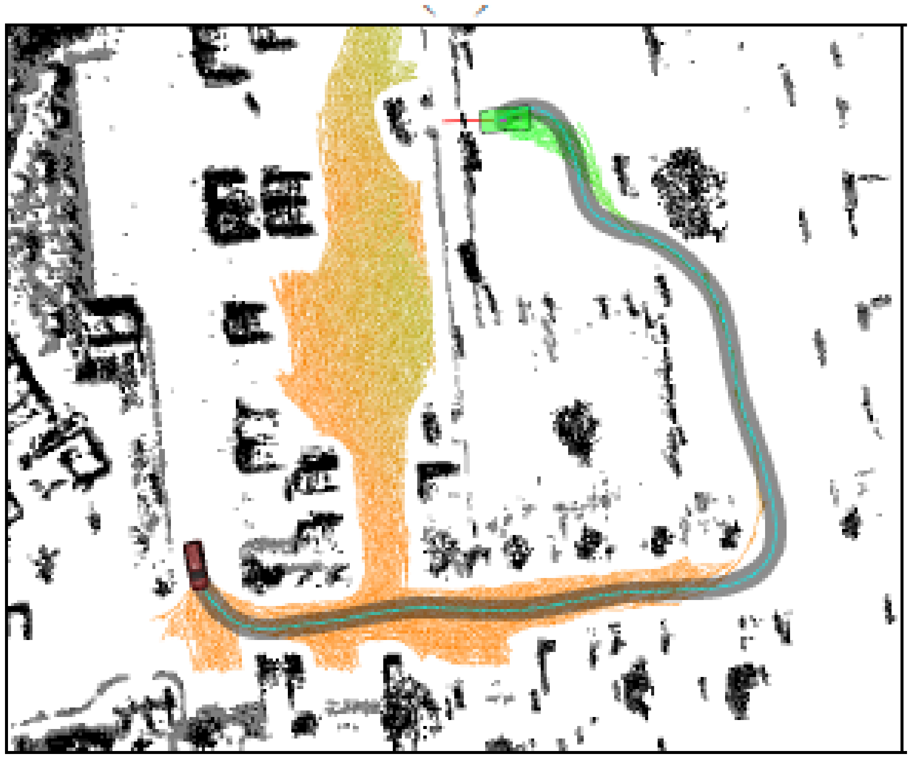


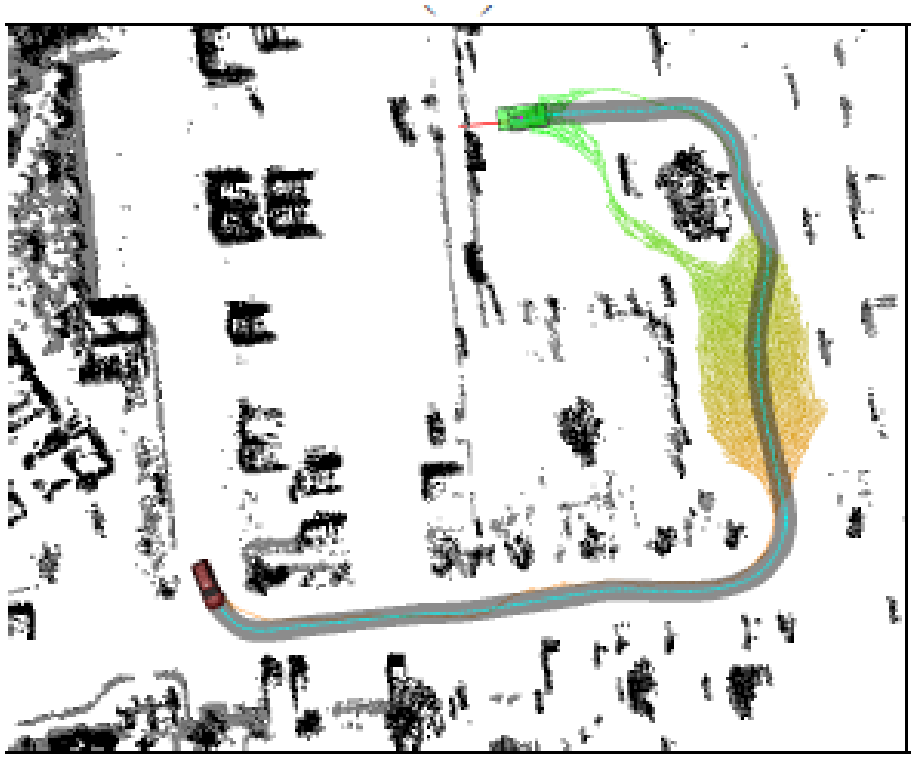
Euclidian Distance



h1(s)

Heuristics



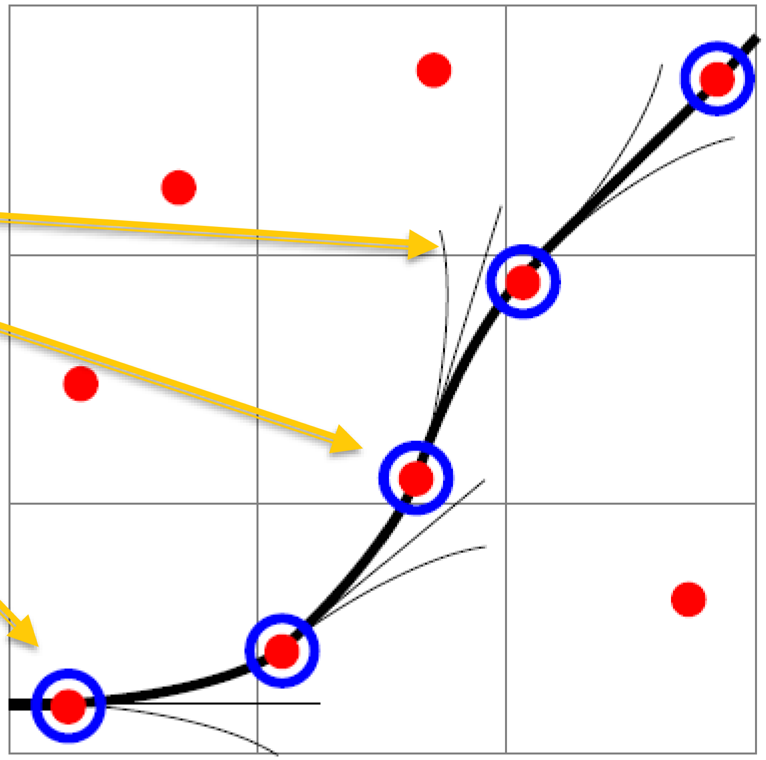


h1(s) only

h1(s) and h2(s)

Analytic Node Expansions

How to precisely determine where these points are?

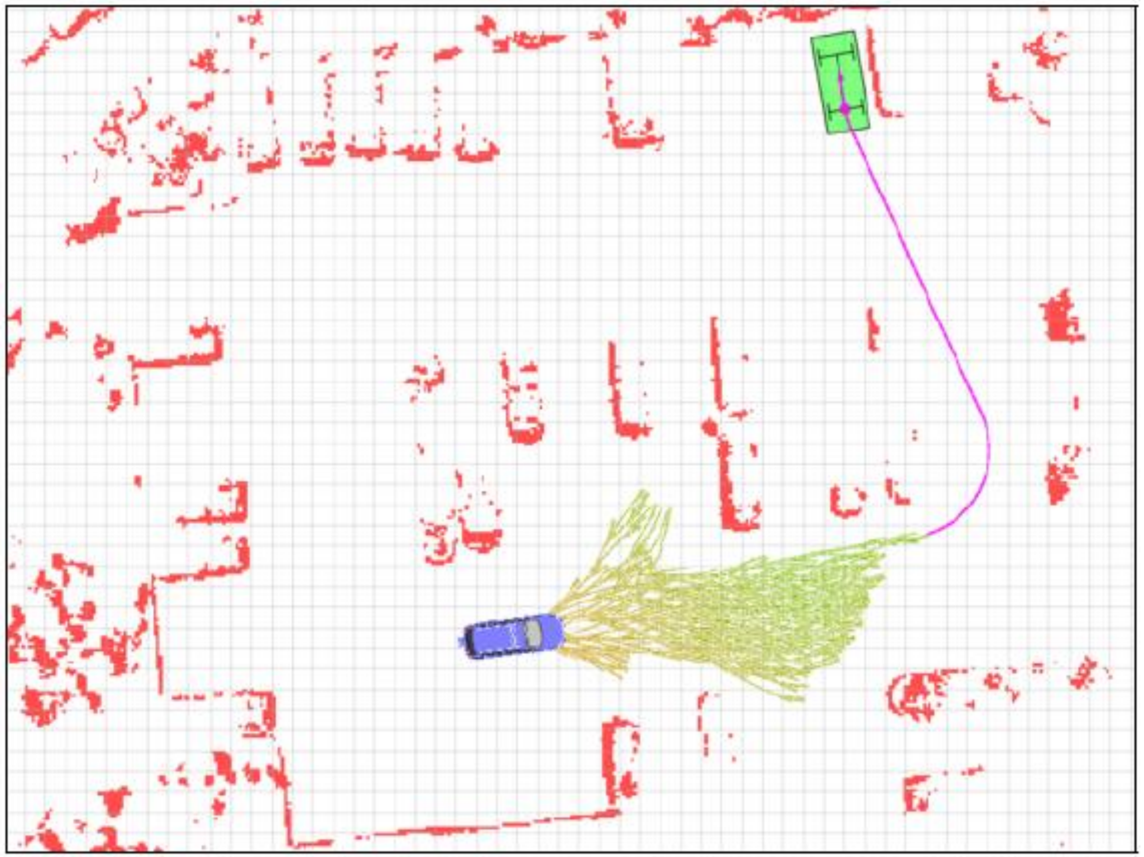


Analytic Node Expansions

- Driving motion primitives
  - A node in the tree is expanded by **simulating a kinematic model** of the car (using a particular control action) for a small period of time
  - The resulting Reed-Shepp path is then checked for collision against the current obstacle map

Analytic Node Expansions

- Yellow-green path
  - Search tree branches (i.e., short incremental expansions)
- Pink path
  - Reed-Shepp path leading towards the goal



Cost function is Critical

- Design a cost function that yields the desired driving behavior
- Criteria??
  - Penalizing proximity to obstacles → Potential field
  - Create high-potential areas in narrow passages → cannot pass

Voronoi field

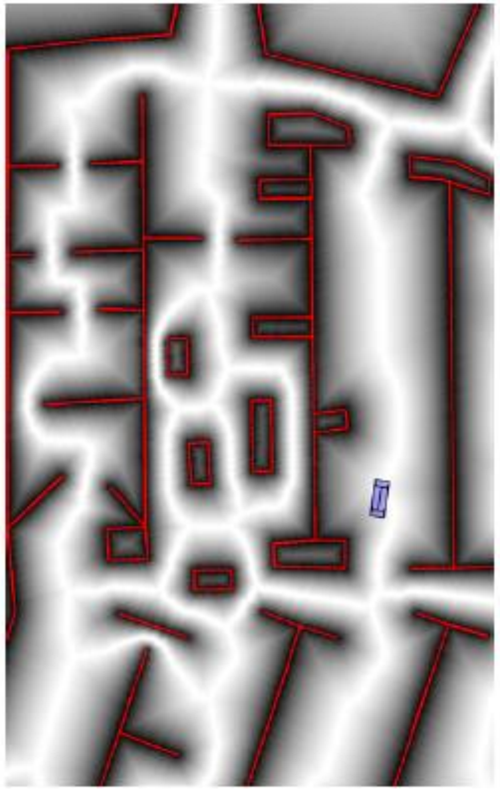
- Rescale the field based on the workspace geometry

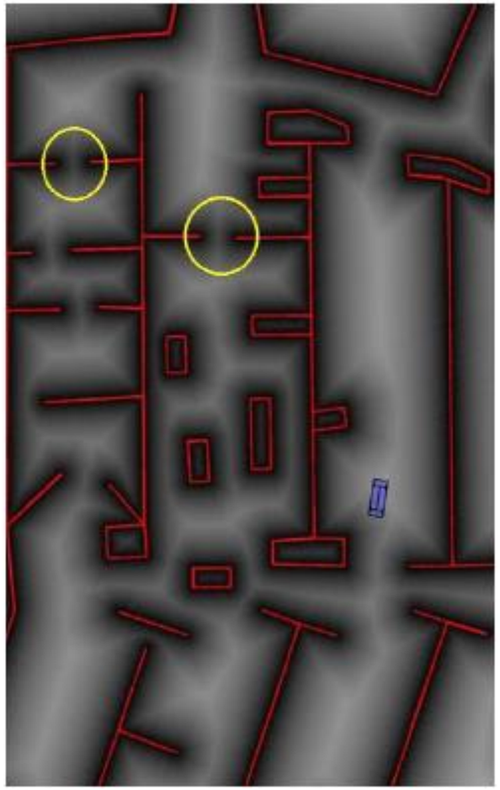
$$\rho_V(x,y) = \left(\frac{\alpha}{\alpha + d_{\mathcal{O}}(x,y)}\right)\left(\frac{d_{\mathcal{V}}(x,y)}{d_{\mathcal{O}}(x,y) + d_{\mathcal{V}}(x,y)}\right)\frac{(d_{\mathcal{O}} - d_{\mathcal{O}}^{max})^2}{(d_{\mathcal{O}}^{max})^2}, \qquad d_{\mathcal{O}} \leq d_{\mathcal{O}}^{max}$$

- Otherwise, $\rho_V(x,y) = 0$

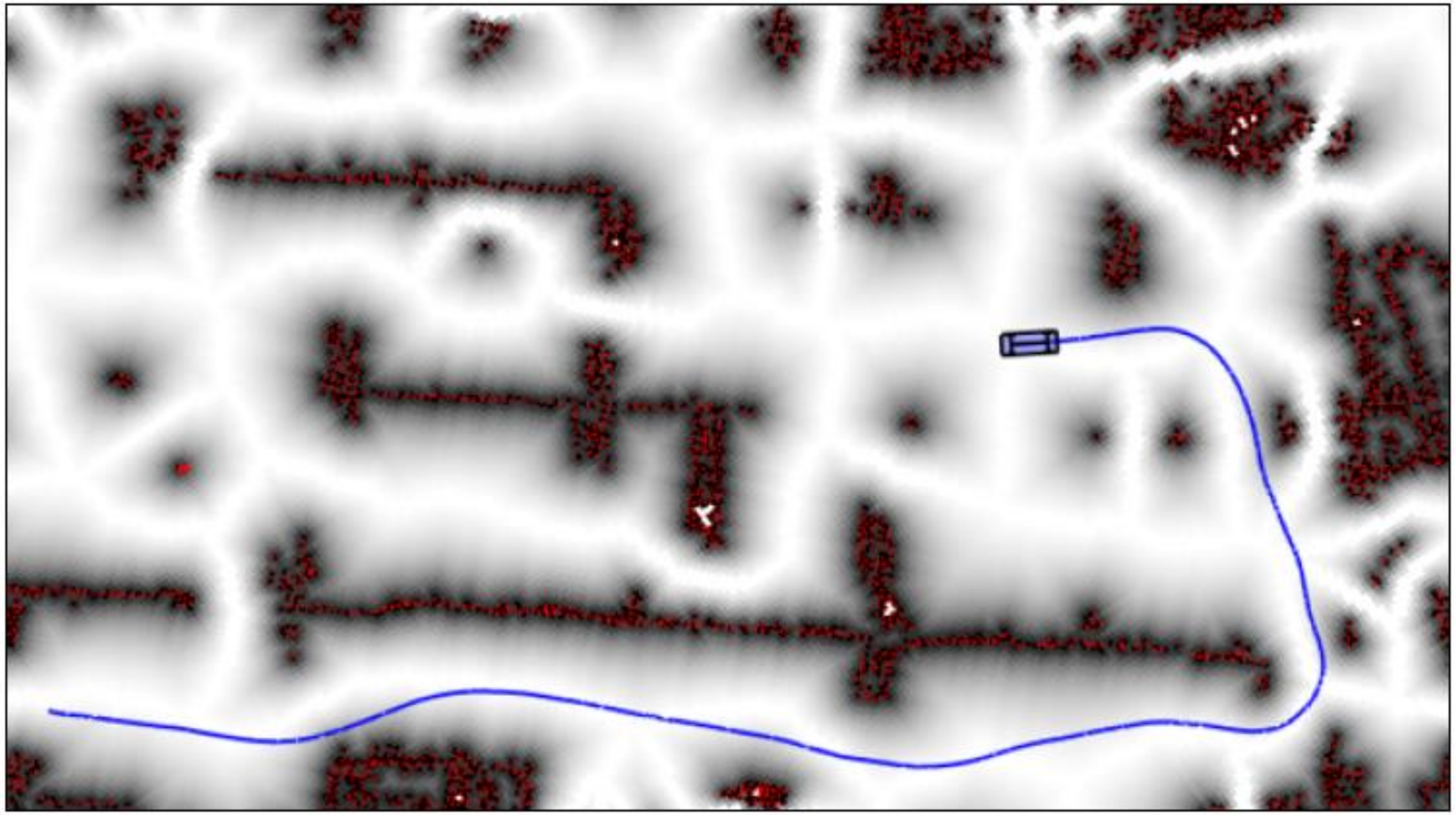
Voronoi field

- Rescale the field based on the workspace geometry





Voronoi Field



Phase 2 – Local Optimization and Smoothing

- Improve path length and solution smoothness

Prefer clearance (Voronoi Field)

Penalize collisions with obstacles

$$w_\rho \sum_{i=1}^{N} \rho_V(x_i, y_i) + w_o \sum_{i=1}^{N} \sigma_o\left(|\mathbf{x}_i - \mathbf{o}_i| - d_{\max}\right) +$$

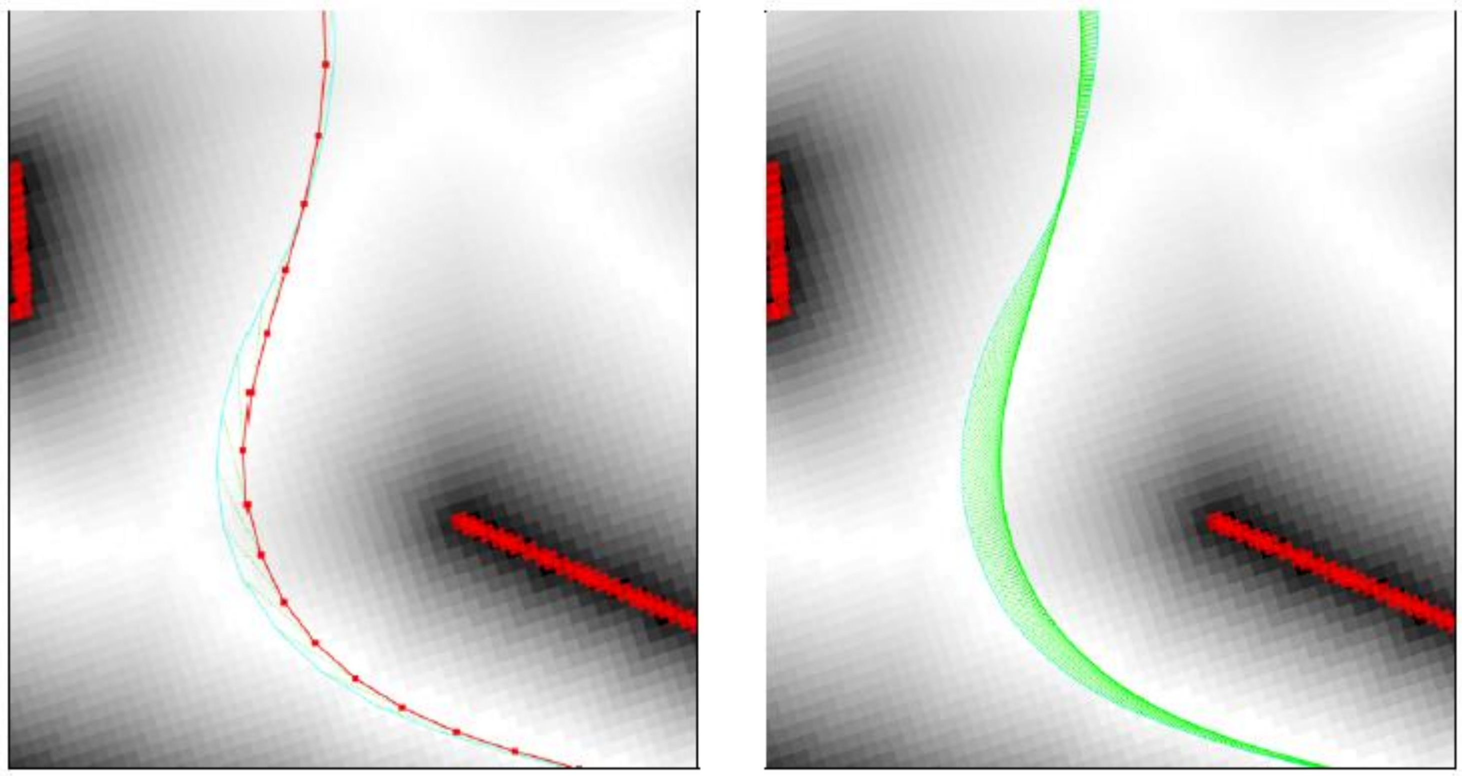
$$w_\kappa \sum_{i=1}^{N-1} \sigma_\kappa\left(\frac{\Delta\phi_i}{|\Delta\mathbf{x}_i|} - \kappa_{max}\right) + w_s \sum_{i=1}^{N-1} (\Delta\mathbf{x}_{i+1} - \Delta\mathbf{x}_i)^2,$$

Bound trajectory curvature

Measure of path smoothness

Phase 2 – Local Optimization and Smoothing

- Improve path resolution by interpolation



Reference

- [1] Dolgov, Dmitri, Sebastian Thrun, Michael Montemerlo, and James Diebel. "Practical search techniques in path planning for autonomous driving." Ann Arbor 1001, no. 48105 (2008): 18-80.

Student talk – Baskaran Prakash

End