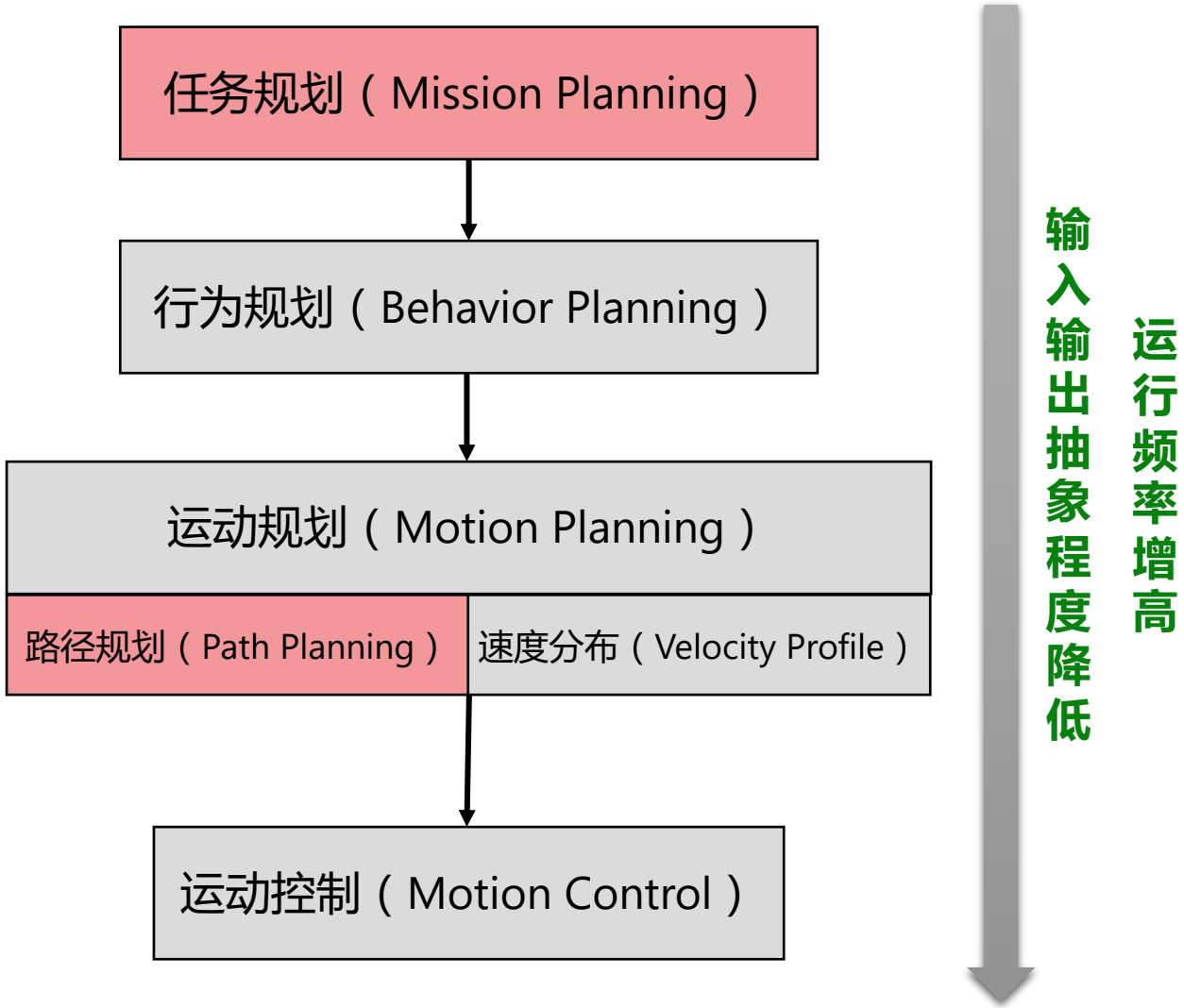
CS7355
自动驾驶前沿技术

2024年4月17日

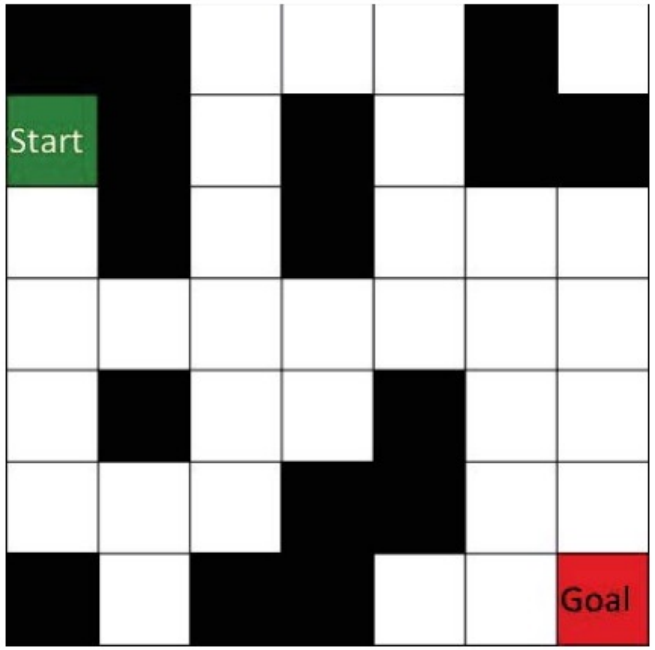
层级式自动驾驶规划架构



任务规划、2D路径规划 = 图搜索

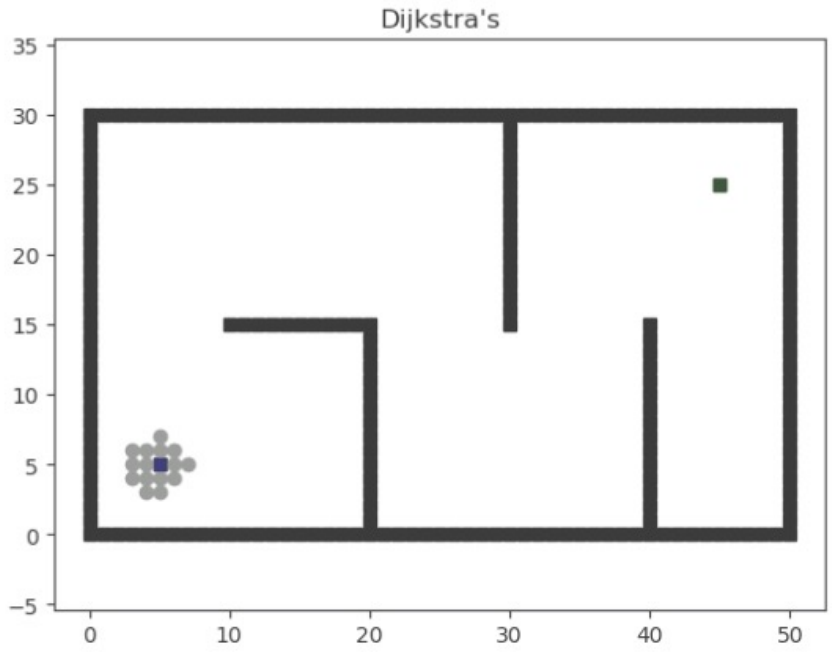
路网图

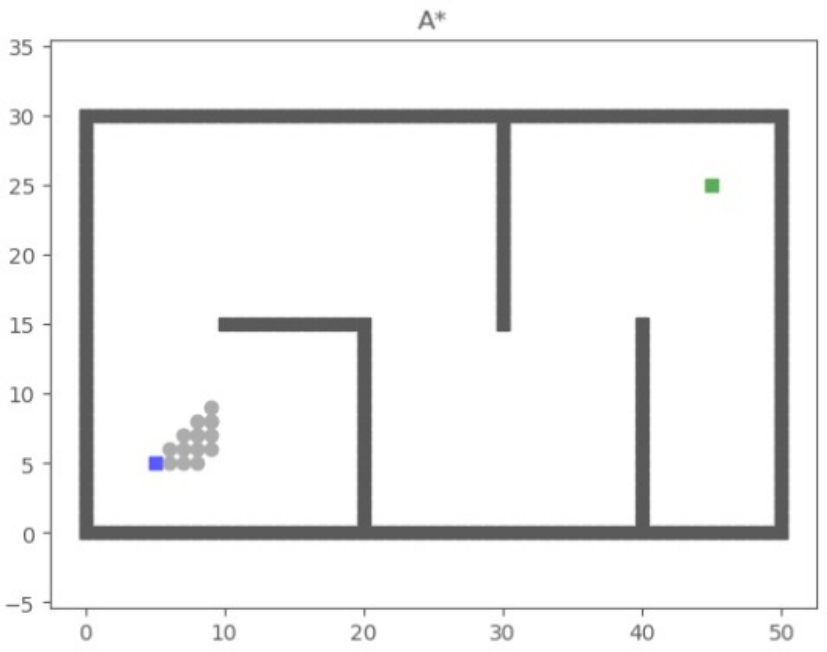
占用栅格

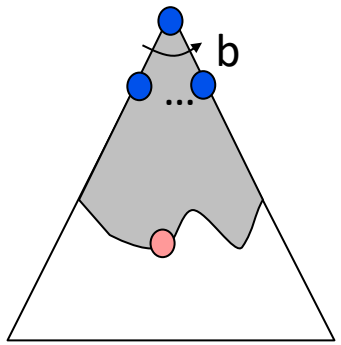


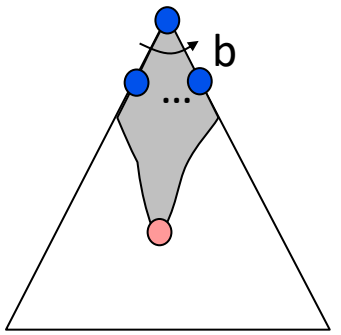
静态图中的路径规划：Dijkstra 与 A*

图片来自 zhm-real



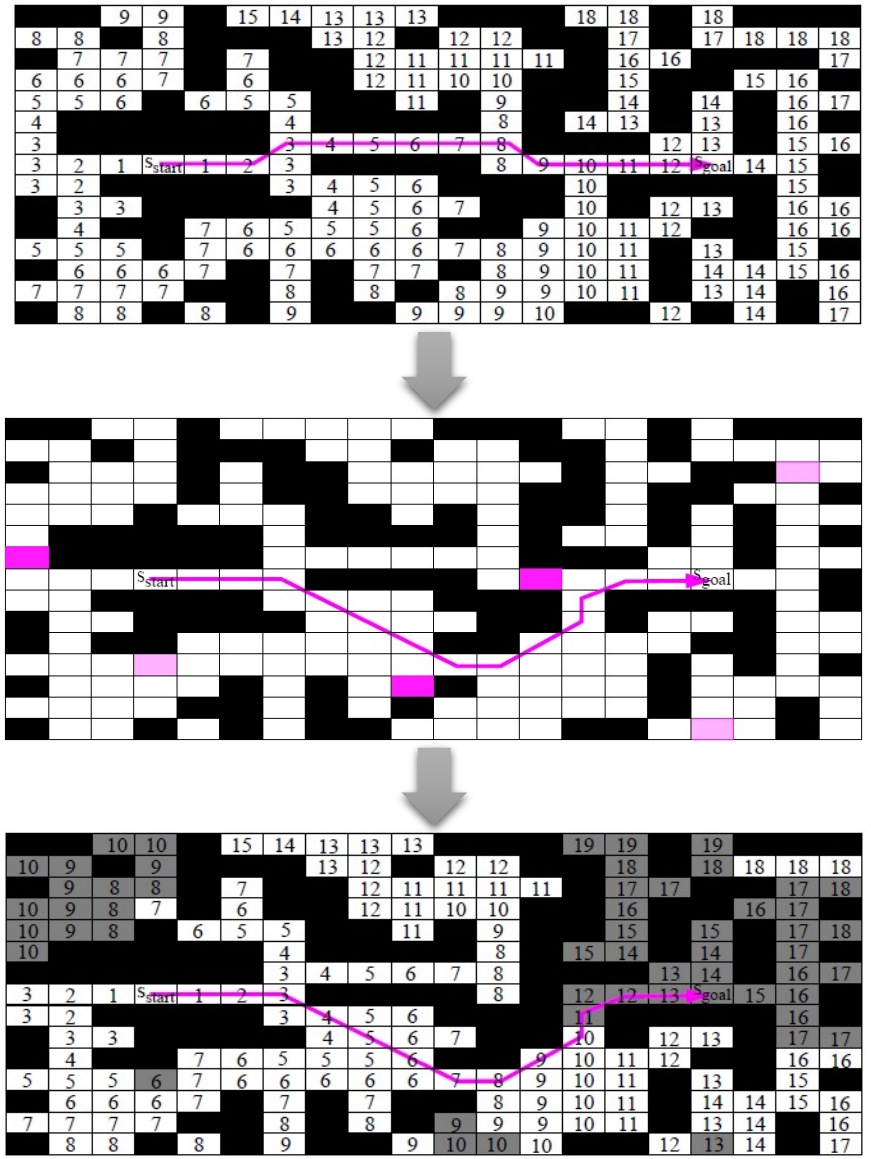






动态图中的路径规划：Incremental Search

- 进行A*搜索，得到初始路径
  - 保存搜索的结果
    - 最优路径
    - 所有已搜索节点的 Cost-to-come
    - 优先队列
- 重复：
  - 执行路径
  - 接收图的局部变化
  - 从最优路径附近出发，拓展和更新局部不一致的顶点



动态图中的路径规划： LPA*

procedure CalculateKey(s)
{01} return $[\min(g(s), rhs(s)) + h(s, s_{goal}); \min(g(s), rhs(s))]$;　　*优先值拆分为两部分: (1)类似 A*； (2)类似 Dijkstra*

procedure Initialize()　　***当第一次规划时：***
{02} $U = \emptyset$;
{03} for all $s \in S$ $rhs(s) = g(s) = \infty$;　　*所有估计成本皆为无穷大（无信息）*
{04} $rhs(s_{start}) = 0$;
{05} U.Insert(s_{start}, CalculateKey(s_{start}));　　*初始扩展队列只有起点*

procedure UpdateVertex(u)　　***当顶点加入扩展队列时：***
{06} if $(u \neq s_{start})$ $rhs(u) = \min_{s' \in Pred(u)}(g(s') + c(s', u))$;　　*计算 rhs-value*
{07} if $(u \in U)$ U.Remove(u);
{08} if $(g(u) \neq rhs(u))$ U.Insert$(u$, CalculateKey$(u))$;　　*检查局部一致性，若不一致，才加入队列并计算优先值*

procedure ComputeShortestPath()　　***每次渐进式搜索时：***
{09} while (U.TopKey() $<$ CalculateKey(s_{goal}) OR $rhs(s_{goal}) \neq g(s_{goal})$)　　*终止条件*
{10}　u = U.Pop();
{11}　if $(g(u) > rhs(u))$　　*估计的Cost-to-come过于悲观*
{12}　　$g(u) = rhs(u)$;
{13}　　for all $s \in Succ(u)$ UpdateVertex(s);　　*更新顶点、后继顶点加入扩展队列*
{14}　else　　*估计的Cost-to-come过于乐观*
{15}　　$g(u) = \infty$;
{16}　　for all $s \in Succ(u) \cup \{u\}$ UpdateVertex(s);　　*重置顶点、自身与后继顶点均加入扩展队列*

procedure Main()　　***主函数入口：***
{17} Initialize();
{18} forever
{19}　ComputeShortestPath();　　*LPA* 渐进式搜索*
{20}　Wait for changes in edge costs;
{21}　for all directed edges (u, v) with changed edge costs
{22}　　Update the edge cost $c(u, v)$;
{23}　　UpdateVertex(v);　　*更新图*

D* Lite

procedure CalculateKey(s)
{01'} return $[\min(g(s), rhs(s)) + h(s_{start}, s) + k_m; \min(g(s), rhs(s))]$;

优先值计算（注意 h 表示 heuristic cost-to-come）

procedure Initialize()
{02'} $U = \emptyset$;
{03'} $k_m = 0$;
{04'} for all $s \in S$ $rhs(s) = g(s) = \infty$;
{05'} $rhs(s_{goal}) = 0$;
{06'} U.Insert(s_{goal}, CalculateKey(s_{goal}));

当第一次规划时：

K_m 代表已经走过的路径长度（0）

所有估计成本皆为无穷大（无信息）

初始扩展队列只有终点

procedure UpdateVertex(u)
{07'} if $(u \neq s_{goal})$ $rhs(u) = \min_{s' \in Succ(u)}(c(u, s') + g(s'))$;
{08'} if $(u \in U)$ U.Remove(u);
{09'} if $(g(u) \neq rhs(u))$ U.Insert$(u$, CalculateKey$(u))$;

当顶点加入扩展队列时：

计算 rhs-value（注意 g 表示 cost-to-go）

检查局部一致性，若不一致，才加入队列并计算优先值

procedure ComputeShortestPath()
{10'} while (U.TopKey() $<$ CalculateKey(s_{start}) OR $rhs(s_{start}) \neq g(s_{start})$)
{11'} $k_{old} =$ U.TopKey();
{12'} $u =$ U.Pop();
{13'} if ($k_{old} <$ CalculateKey(u))
{14'} U.Insert$(u$, CalculateKey$(u))$;
{15'} else if $(g(u) > rhs(u))$
{16'} $g(u) = rhs(u)$;
{17'} for all $s \in Pred(u)$ UpdateVertex(s);
{18'} else
{19'} $g(u) = \infty$;
{20'} for all $s \in Pred(u) \cup \{u\}$ UpdateVertex(s);

每次渐进式搜索时：

终止条件

检测到路径成本升高，重新放回扩展队列

估计的 cost-to-go 过于悲观

更新顶点、前继顶点加入扩展队列

估计的 cost-to-go 过于乐观

重置顶点、自身与前继顶点均加入扩展队列

procedure Main()
{21'} $s_{last} = s_{start}$;
{22'} Initialize();
{23'} ComputeShortestPath();
{24'} while $(s_{start} \neq s_{goal})$
{25'} /* if $(g(s_{start}) = \infty)$ then there is no known path */
{26'} $s_{start} = \arg\min_{s' \in Succ(s_{start})}(c(s_{start}, s') + g(s'))$;
{27'} Move to s_{start};
{28'} Scan graph for changed edge costs;
{29'} if any edge costs changed
{30'} $k_m = k_m + h(s_{last}, s_{start})$;
{31'} $s_{last} = s_{start}$;
{32'} for all directed edges (u, v) with changed edge costs
{33'} Update the edge cost $c(u, v)$;
{34'} UpdateVertex(u);
{35'} ComputeShortestPath();

主函数入口：

第一次搜索（逆向 A）*

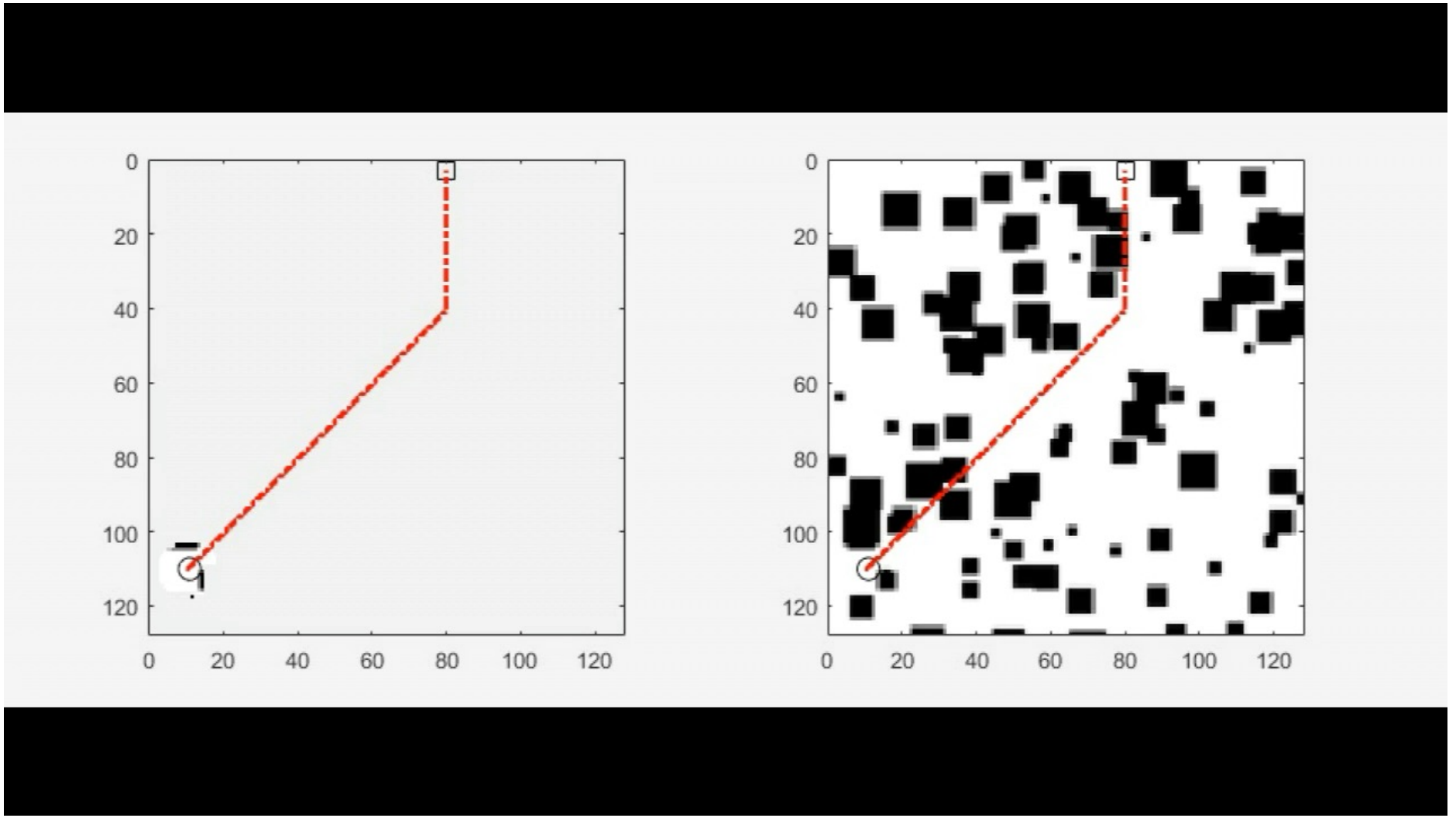
移动机器人、更新起点

更新已经走过的路径长度

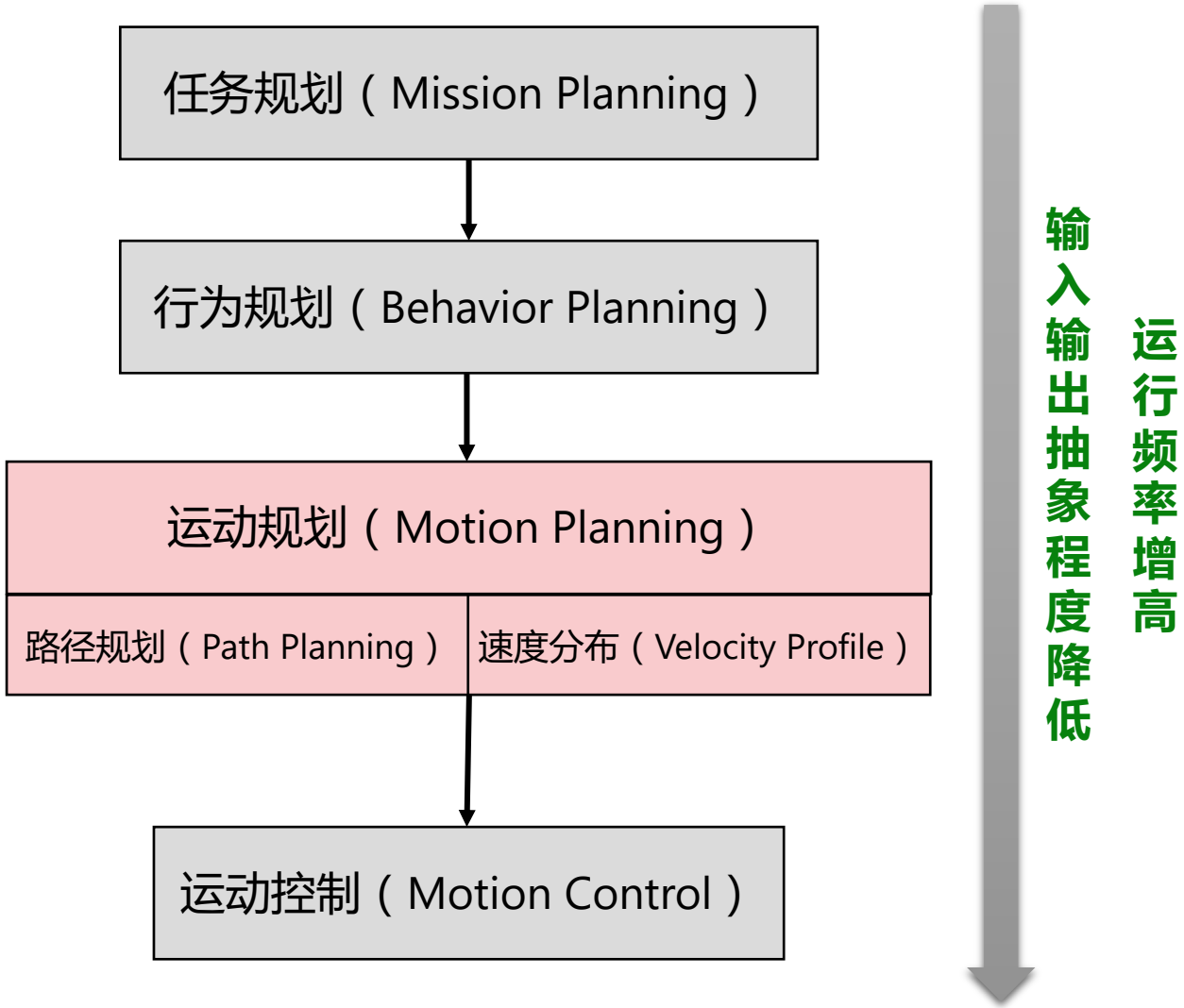
更新图

D 渐进式搜索（逆向 LPA*）*

动态图中的路径规划：D* Lite



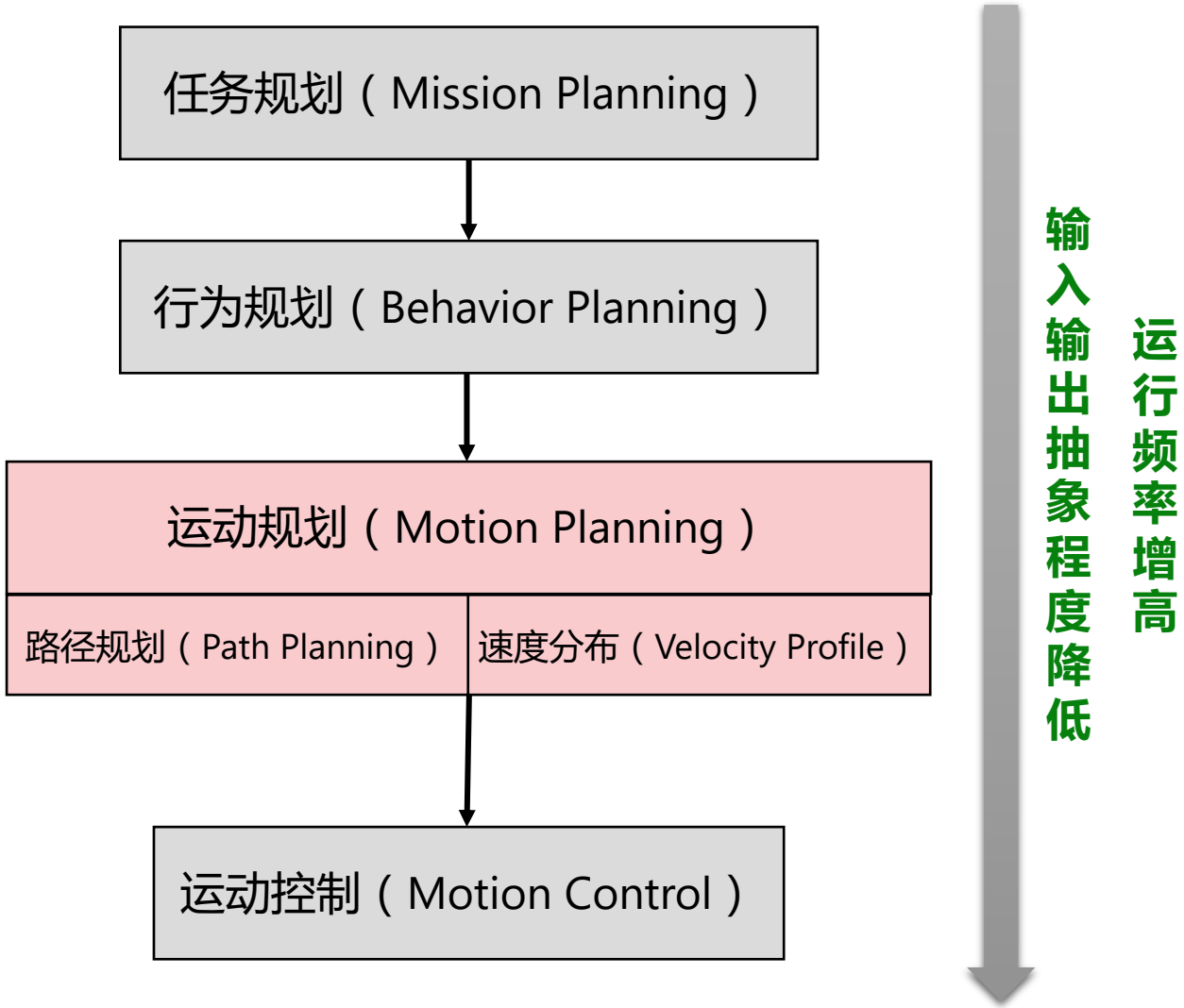
层级式自动驾驶规划架构



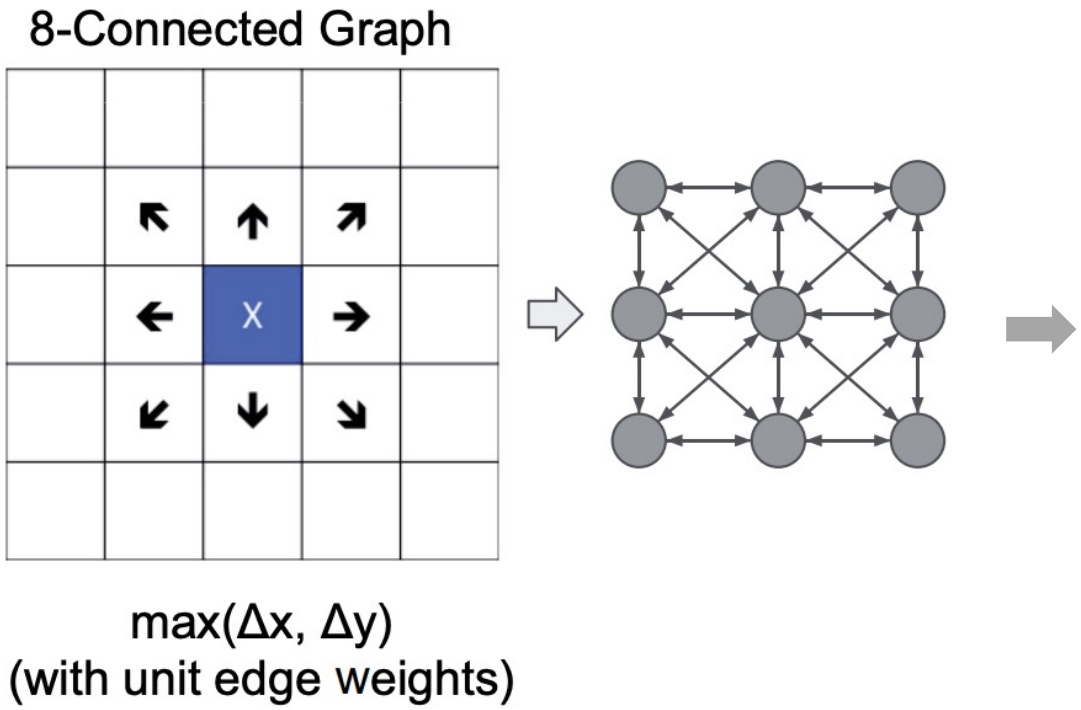
运动规划术语

- **轨迹**（trajectory）
  - $\tau(t)$：以时间 t 为参数的状态曲线
  - $\dot{\tau}(t)$：以时间 t 为参数的速度曲线
- **路径**（path）
  - $\tau(s)$：以弧长 s 为参数的状态曲线
  - 弧长（arc length）：已经走过的路径长度
    - 速度信息被抹去 $s = \int_0^{t'} |\dot{\tau}(t)|\, dt$
    - $\dot{\tau}(s)$ 是切线方向的单位向量
    - $\ddot{\tau}(s)$ 是长度为曲率的法向向量
- 规划问题分类：
  - **轨迹规划**（trajectory planning）需要输出轨迹 $\tau(t)$
  - **路径规划**（path planning）只需要输出路径 $\tau(s)$
  - **运动规划**（motion planning）可以指代轨迹规划、路径规划、生成沿着路径的速度曲线 $v(s)$

层级式自动驾驶规划架构

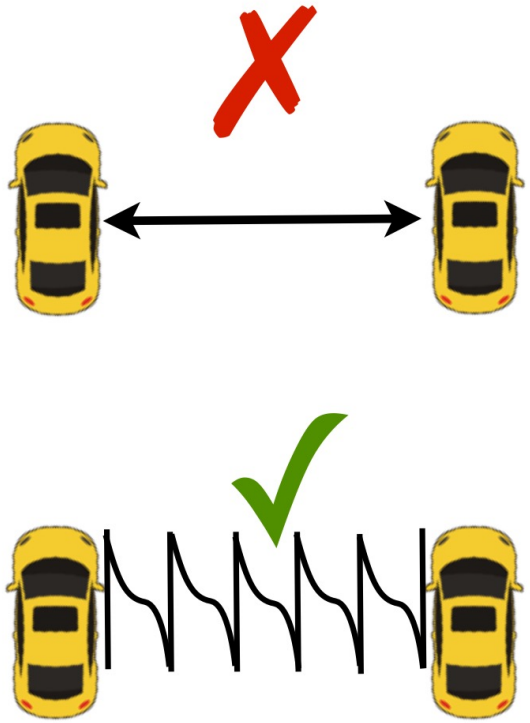


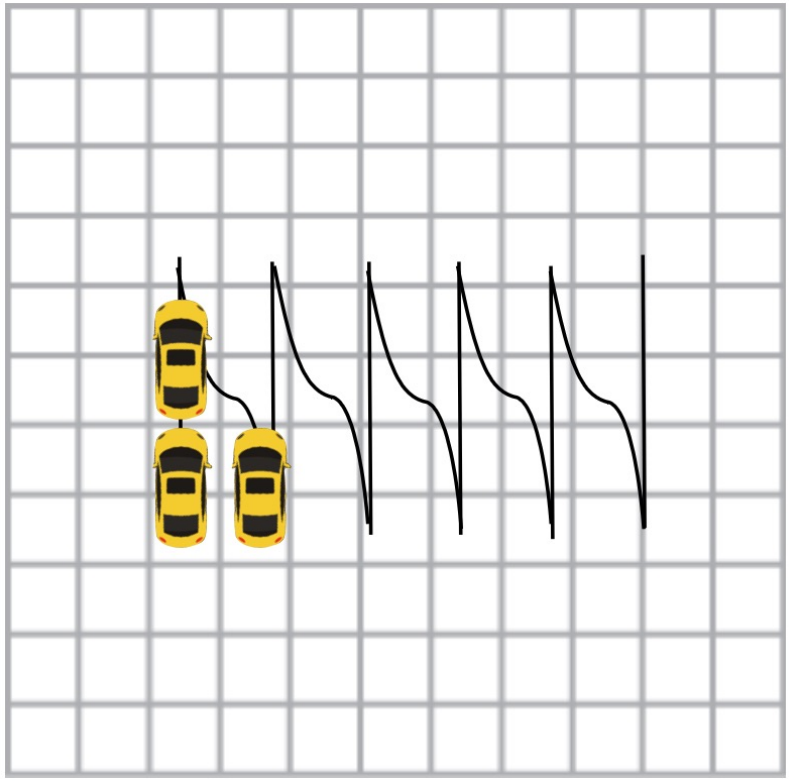
用 A* 或者 D* Lite 实现车辆路径规划？



车辆运动无法直接应用 A* 与 D* Lite！

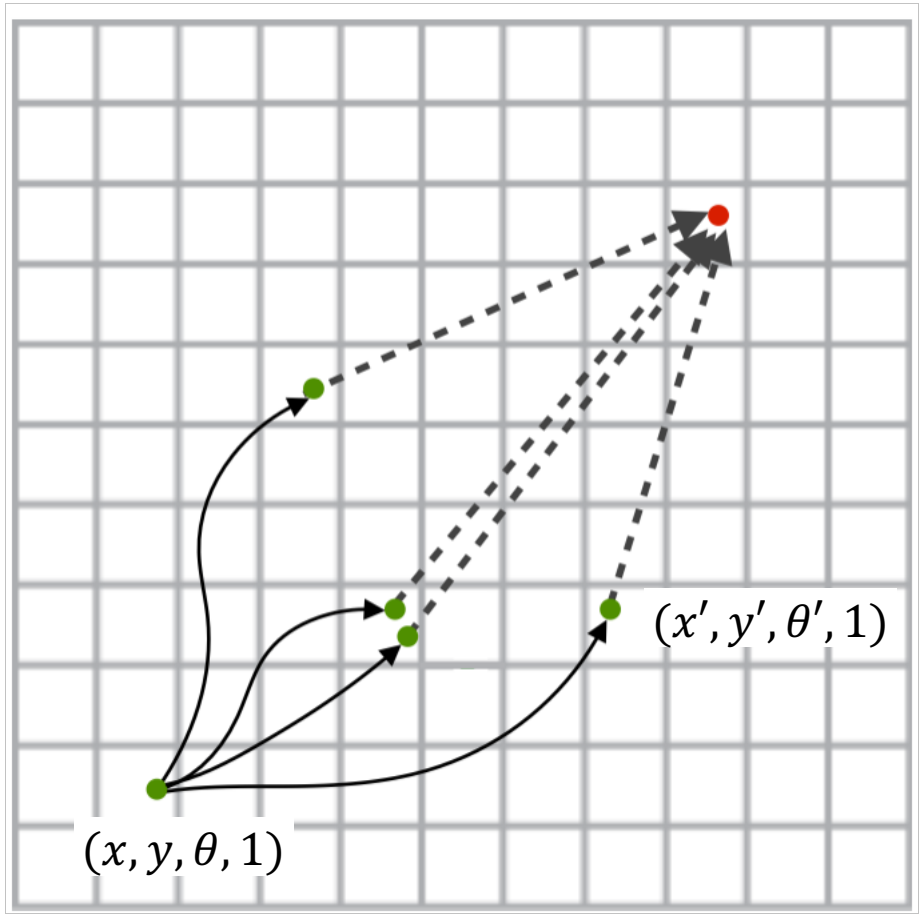
- 车辆运动学具有 nonholonomic constraint，无法按任意方向平移
- Occupancy grid 的连通性质非常复杂





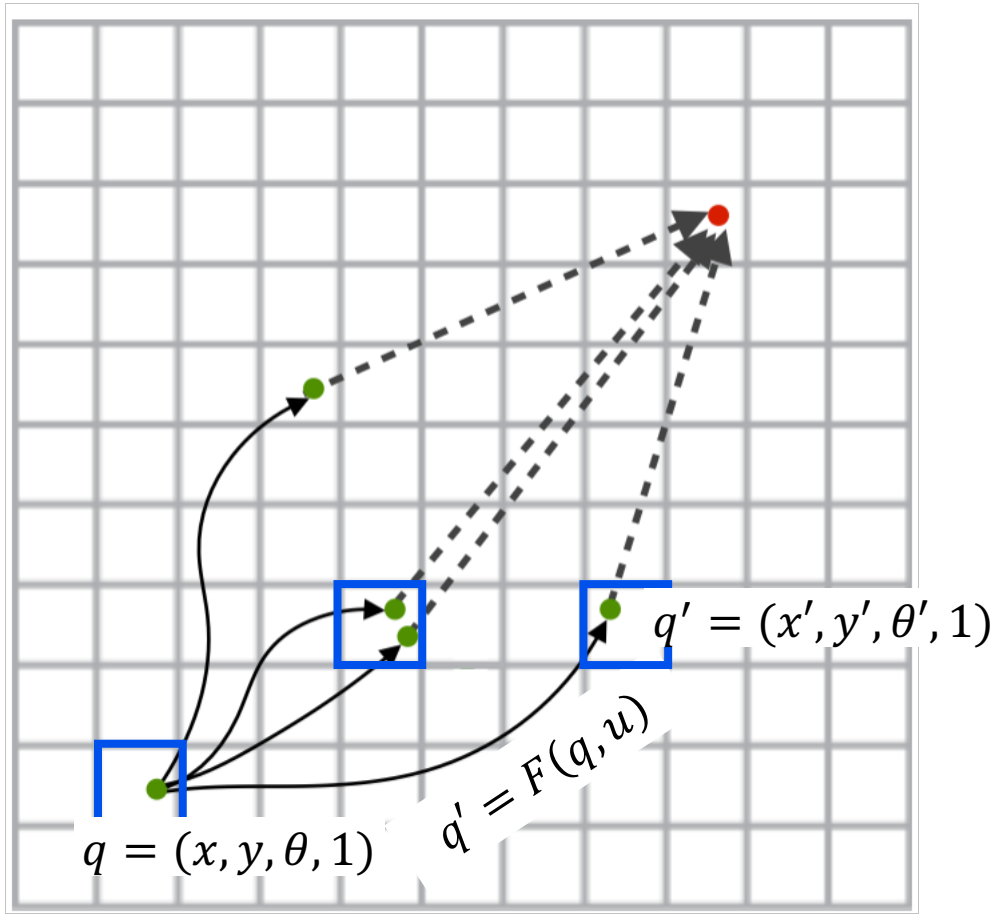
Hybrid A* 算法

- **栅格搜索**：状态(x, y, θ, dir) 构成的空间被离散化为 4D 栅格
- **Hybrid**: 每个**离散**的格子同时保存了一个**连续**的车辆状态
  - 格子中心很难被车辆到达
  - 格子中所储存连续状态是车辆可以到达的
  - "Hybrid" 的由来

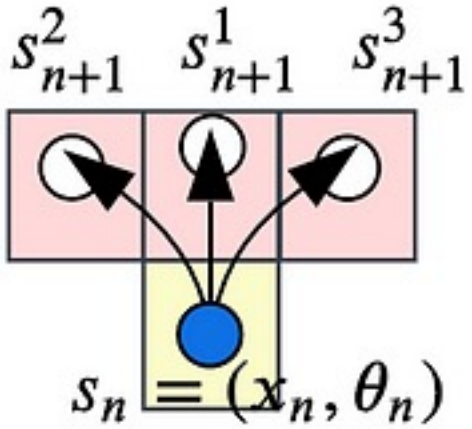


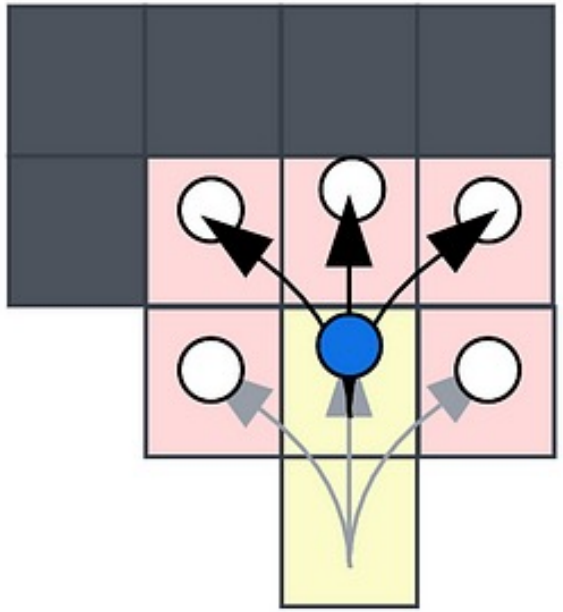
Hybrid A* 算法

- **节点扩展**：
  - 当前格子 c_i 包含连续状态 q
  - 运行控制 u 可得到后继状态 $q' = F(q,u)$
  - q' 落入另一个 c_j ，进行如下检测:
    - 若路径 (c_i, c_j) 检测到碰撞：
      - 丢弃 c_j
    - 若 c_j 未被访问：
      - 赋值 q'
      - 加入搜索树
    - 若 c_j 已经被访问 ：
      - c_j 中储存了另一个 $\widehat{q'}$
      - 比较 $f(\widehat{q'})$ 与 $f(q')$，取最优的状态

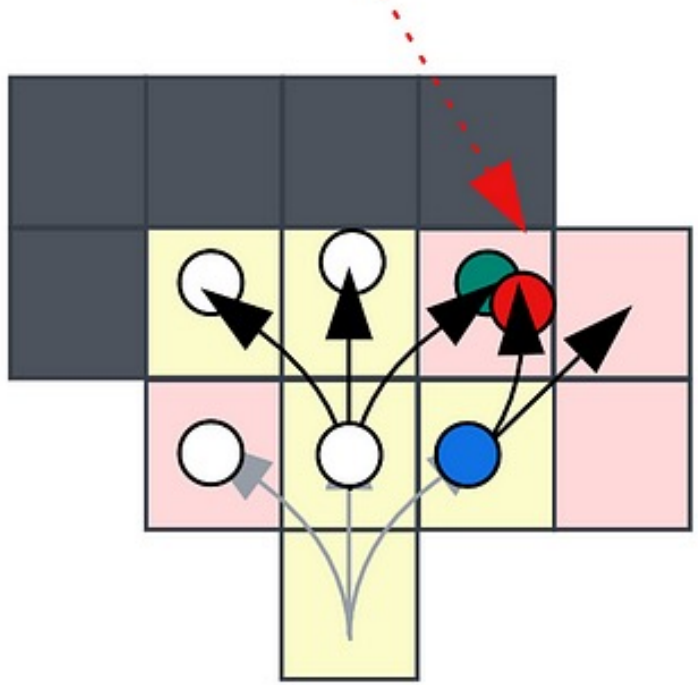


Hybrid A* 树扩展





比较 $f(q) = g(q) + h(q)$
只保留最优的 q



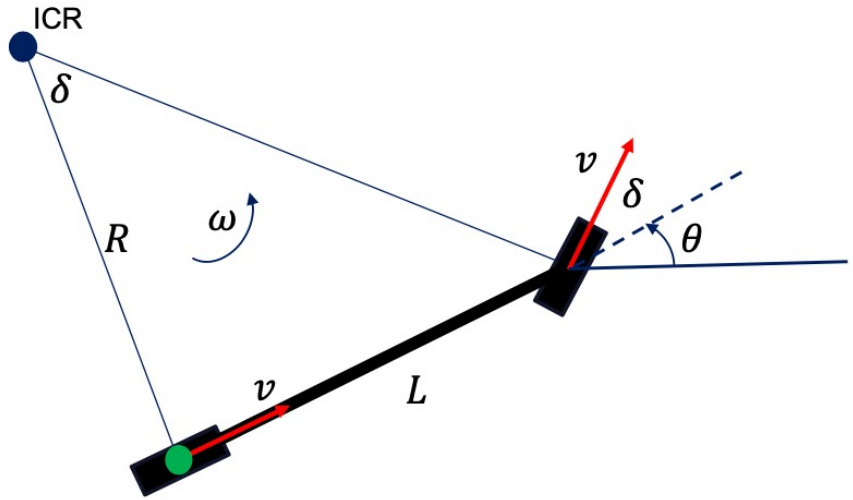
扩展轨迹 $q' = F(q, u)$

- 离散时间的自行车模型展开

$$x_n = x_{n-1} + v_{n-1}\cos(\theta_{n-1})\,\Delta t$$

$$y_n = x_{n-1} + v_{n-1}\sin(\theta_{n-1})\,\Delta t$$

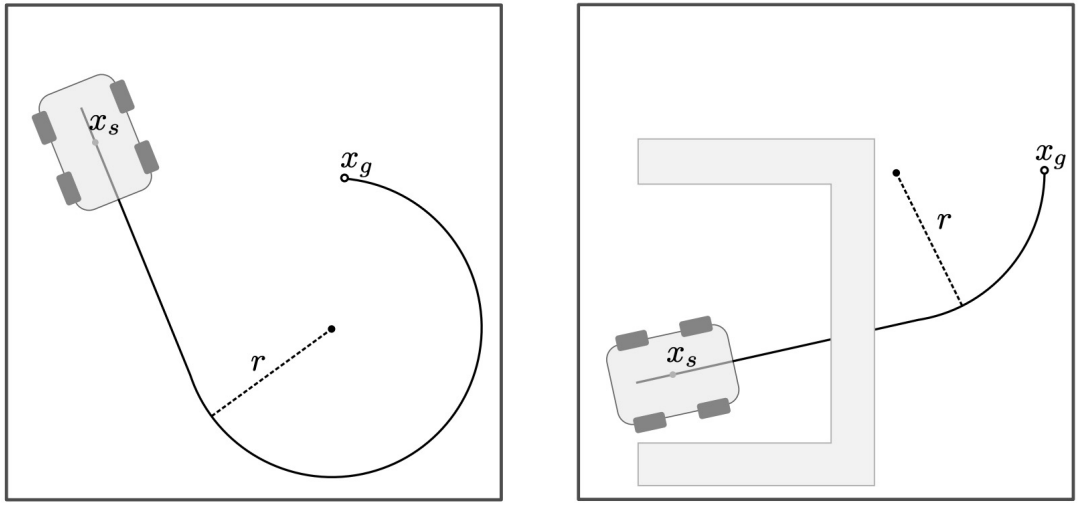
$$\theta_n = \theta_{n-1} + \frac{v_{n-1}\tan(\delta_{n-1})}{L}\Delta t$$



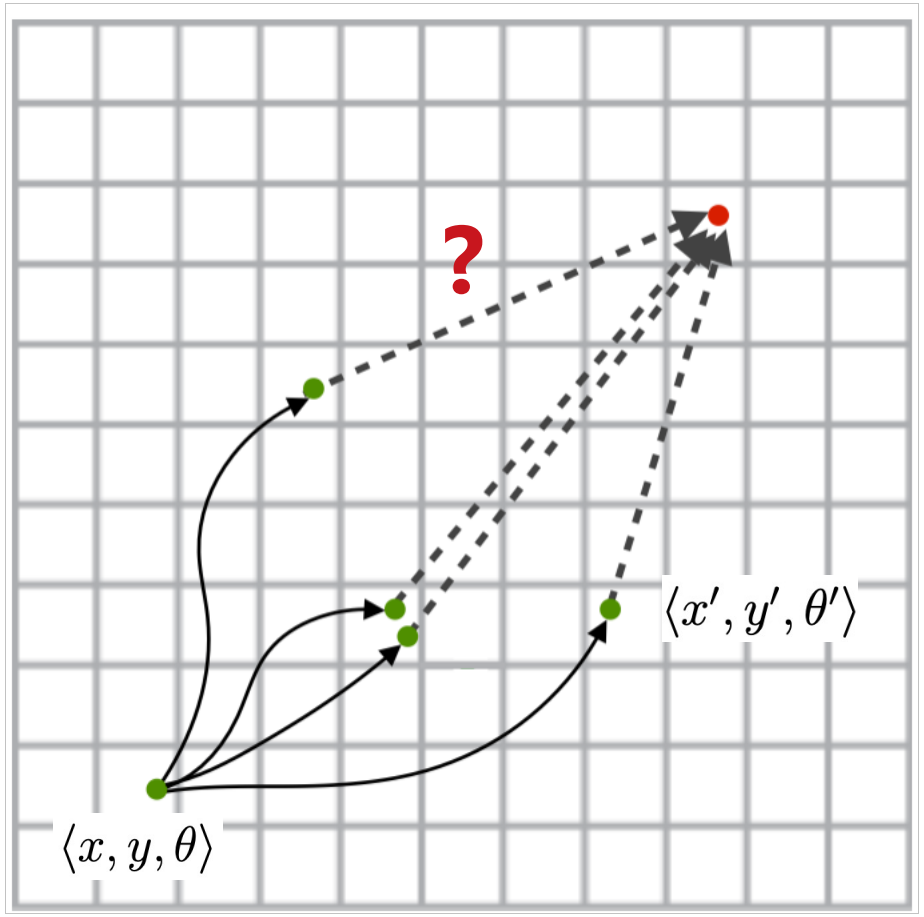
$$\dot{x} = v\cos(\theta)$$
$$\dot{y} = v\sin(\theta)$$
$$\dot{\theta} = \frac{v\tan(\delta)}{L}$$
$$\delta_{min} \le \delta \le \delta_{max}$$
$$v_{min} \le v \le v_{max}$$

Hybrid A* Heuristics

- 欧氏距离可用，但不好用
- 启发函数 <- relaxed problem
- **启发1**：Nonholonomic without obstacles h_1
  - 保持非完整约束
  - 去掉所有障碍物



图片来自 Karl's master thesis



Dubins Car 简化模型

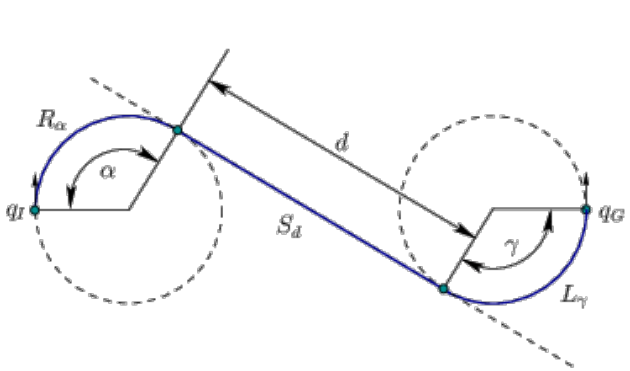
- 自行车模型：

$$\dot{x_r} = v\cos\theta$$
$$\dot{y_r} = v\sin\theta$$
$$\dot{\theta} = \frac{v\tan\phi}{L}$$

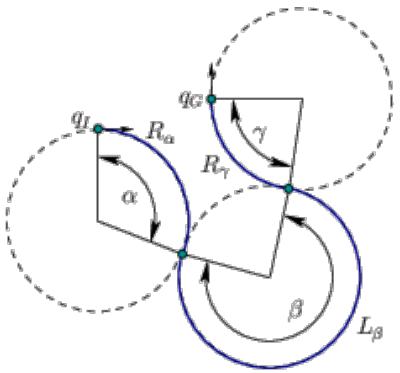
- **Dubins Car:** 当只采用恒定速度和最大转向角时，运动学模型可以简化为：

$$\dot{x_r} = v_0\cos\theta$$
$$\dot{y_r} = v_0\sin\theta$$
$$\dot{\theta} = \frac{u}{r_{min}}, u \in \{v_0, 0, -v_0\}$$

- **Dubins Curve:** Dubins car 从任意起点状态到任意终点状态的最短路径



$R_\alpha S_d L_\gamma$



$R_\alpha L_\beta R_\gamma$

Dubins Curves 最短路径

ON CURVES OF MINIMAL LENGTH WITH A CONSTRAINT ON AVERAGE CURVATURE, AND WITH PRESCRIBED INITIAL AND TERMINAL POSITIONS AND TANGENTS.*[1]

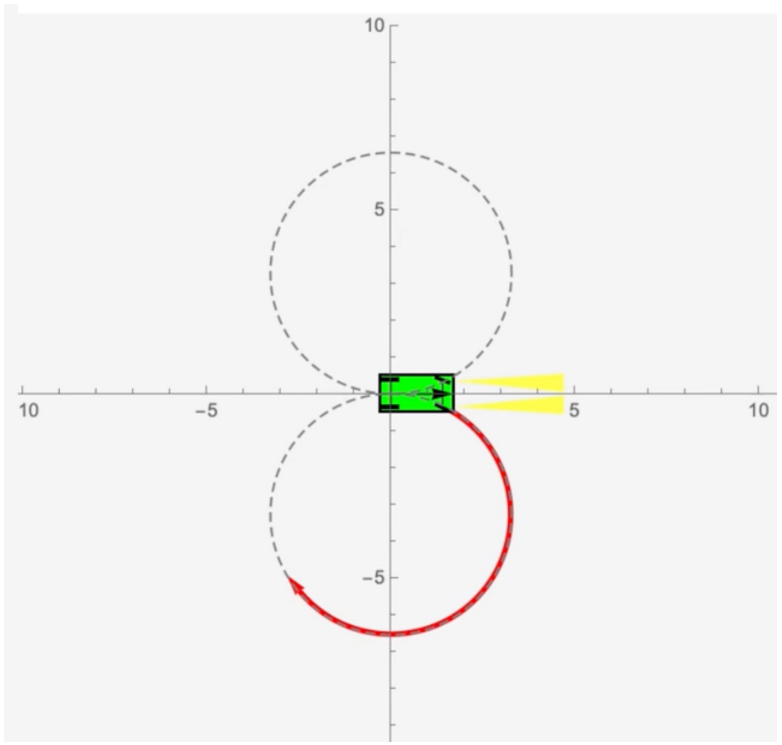
By L. E. Dubins.

6. Principal result. We wish to show that a planar R-geodesic is necessarily a very special curve, i.e. a continuously differentiable curve which consists of at most *three* pieces, each of which is either a straight line segment or an arc of a circle of radius R. Errett Bishop pointe
author that in view of Proposition 13 it is sufficient to show t
which consists of four such pieces can be an R-geodesic.

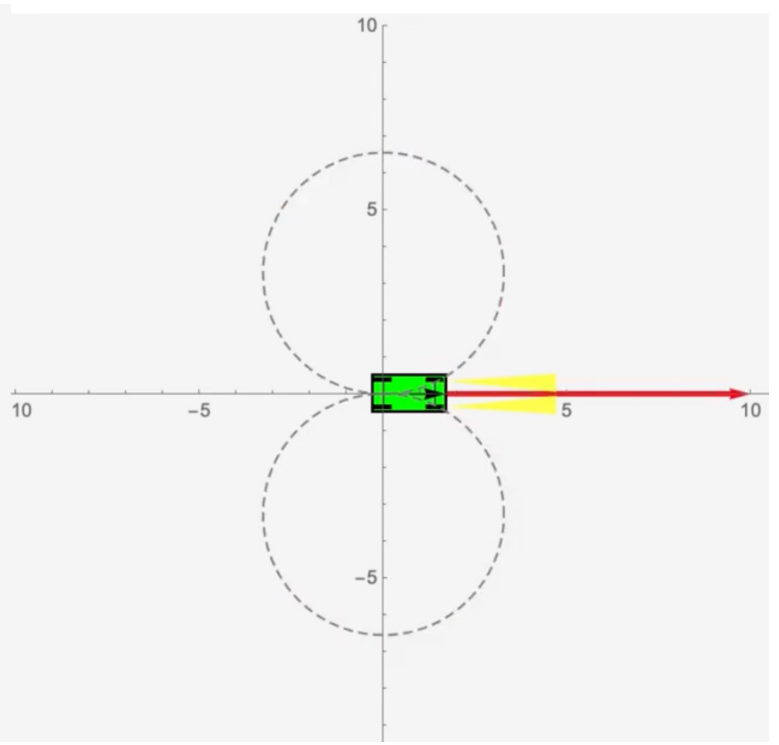
Let us designate a continuously differentiable curve by CC
that it consists of precisely four arcs of circles of radius R. Sin
designate by $CLCL$ a continuously differentiable curve which

Dubins Curves Motion Primitives

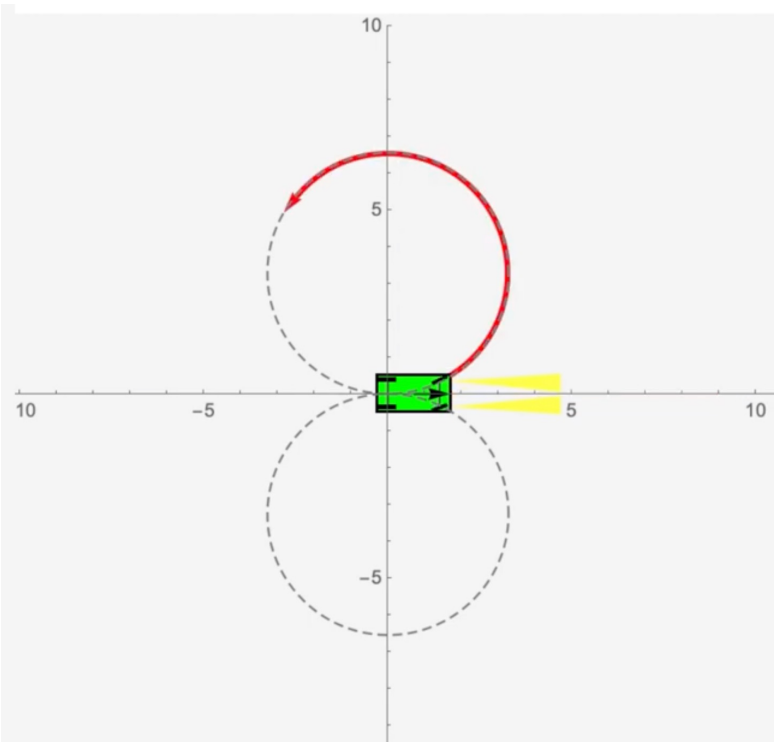
R



S



L



Dubins Curves 24 种可能性

{{S, S, S}, {S, S, L}, {S, S, R}, {S, L, S}, {S, L, L},

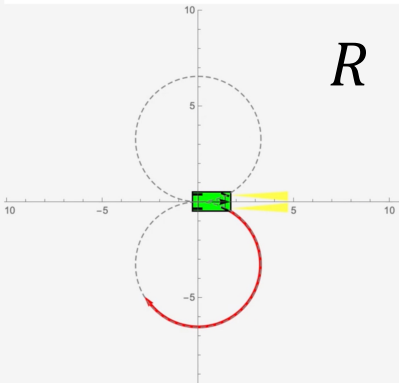
{S, L, R}, {S, R, S}, {S, R, L}, {S, R, R},{L, S, S},

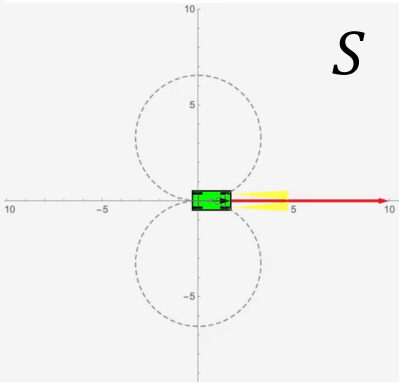
{L, S, L}, {L, S, R}, {L, L, S}, {L, L, L}, {L, L, R},

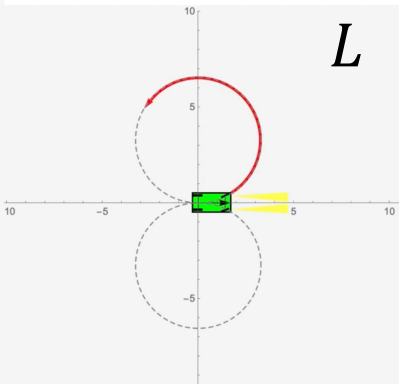
{L, R, S}, {L, R, L}, {L, R, R}, {R, S, S}, {R, S, L},

{R, S, R}, {R, L, S}, {R, L, L}, {R, L, R}, {R, R, S},

{R, R, L}, {R, R, R} }

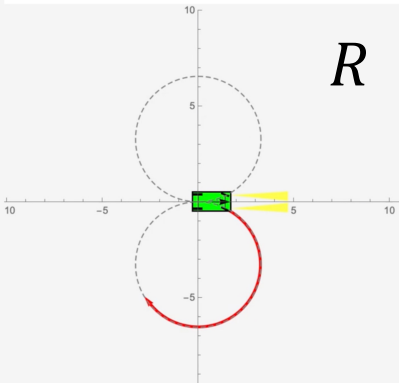


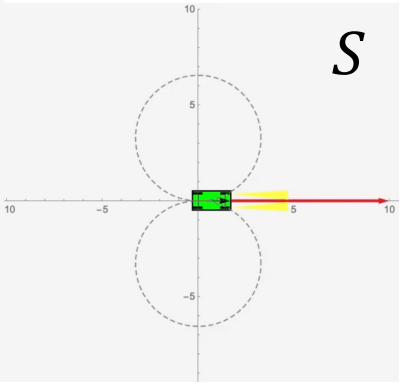


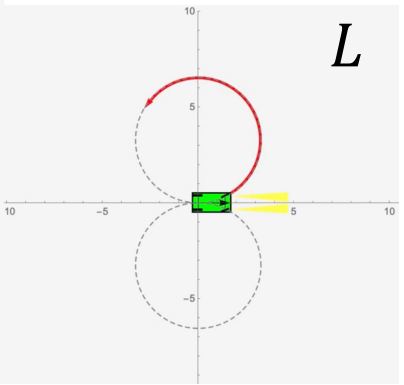


Dubins Curves 12 种可能性

{{S, S, S}, {S, S, L}, {S, S, R}, {S, L, S}, {S, L, L},
{S, L, R}, {S, R, S}, {S, R, L}, {S, R, R}, {L, S, S},
{L, S, L}, {L, S, R}, {L, L, S}, {L, L, L}, {L, L, R},
{L, R, S}, {L, R, L}, {L, R, R}, {R, S, S}, {R, S, L},
{R, S, R}, {R, L, S}, {R, L, L}, {R, L, R}, {R, R, S},
{R, R, L}, {R, R, R} }







Dubins Curves 12 种可能性

{{S, S, S},

{S, L, R},

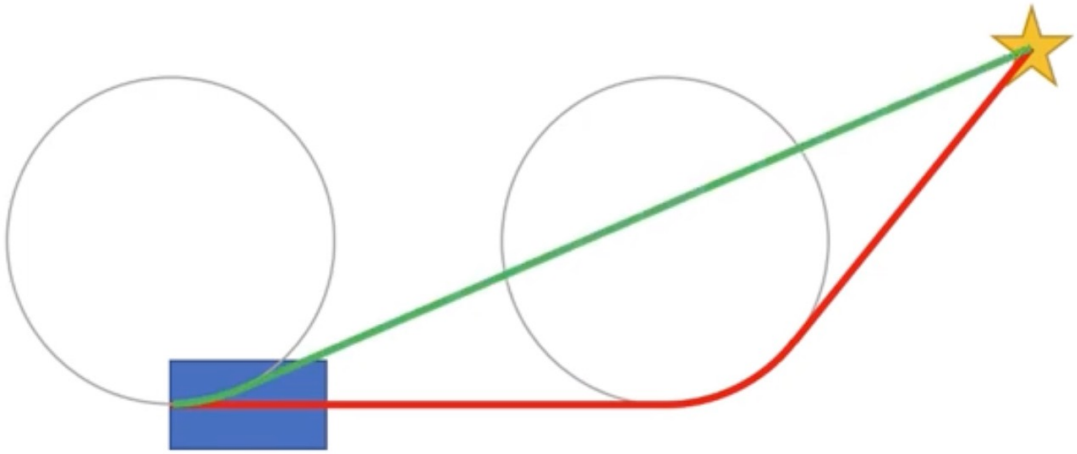
{L, S, L},

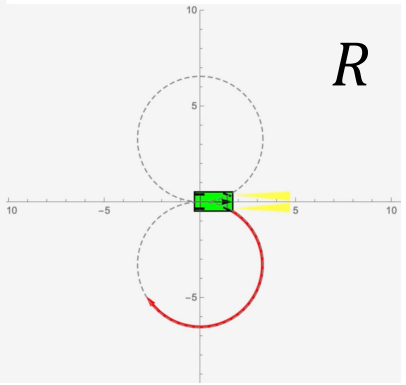
{L, R, S},

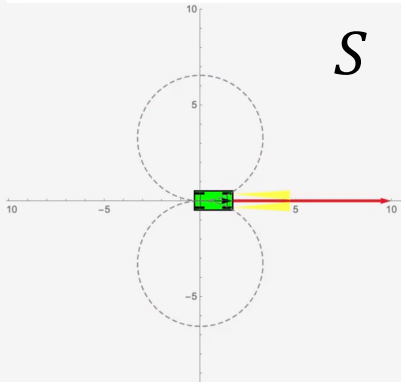
{R, S, R},

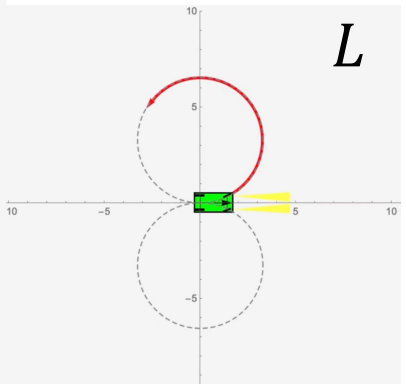
{L, R, L},

$\{L, S, R\} > \{S, L, R\}$



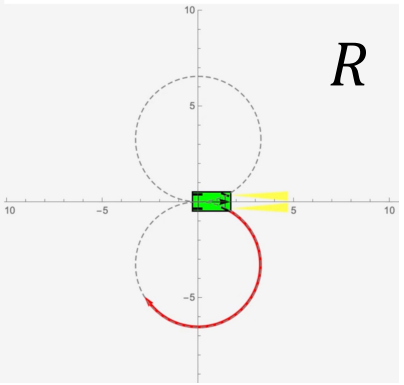


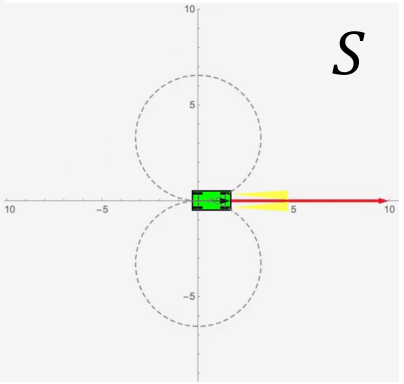


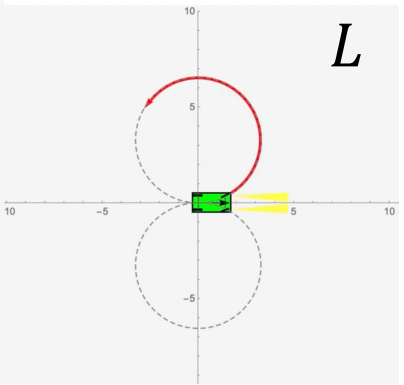


Dubins Curves 6 种可能性

{{S, S, S}, {S, S, L}, {S, S, R}, {S, L, S}, {S, L, L},
{S, L, R}, {S, R, S}, {S, R, L}, {S, R, R}, {L, S, S},
{L, S, L}, {L, S, R}, {L, L, S}, {L, L, L}, {L, L, R},
{L, R, S}, {L, R, L}, {L, R, R}, {R, S, S}, {R, S, L},
{R, S, R}, {R, L, S}, {R, L, L}, {R, L, R}, {R, R, S},
{R, R, L}, {R, R, R}}

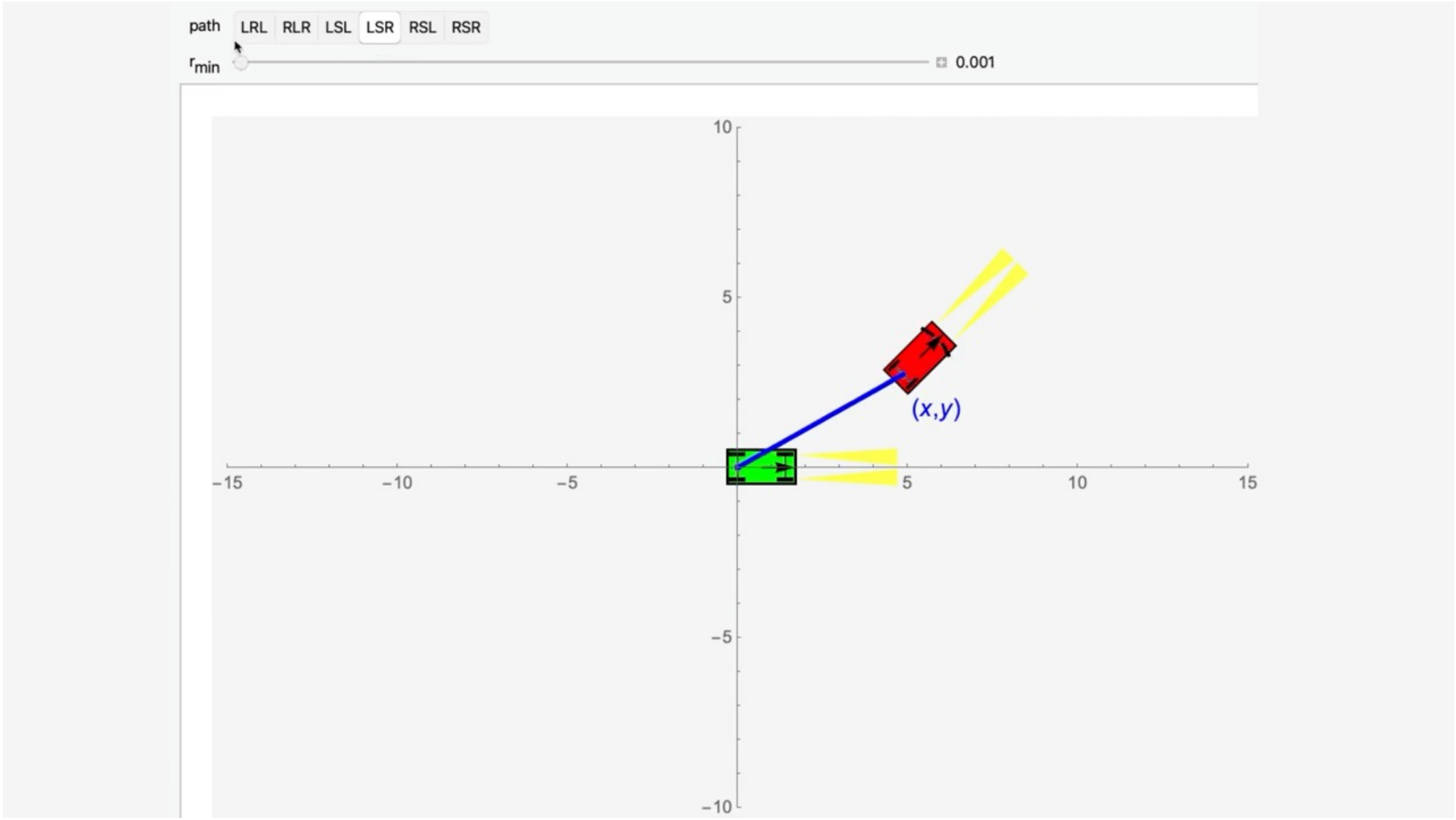






Dubins Curves 演示

https://youtu.be/tjUVE3jto98



Reeds-Shepp Curves

- 当 Dubins car 额外允许倒车时，有48（46）种组合：

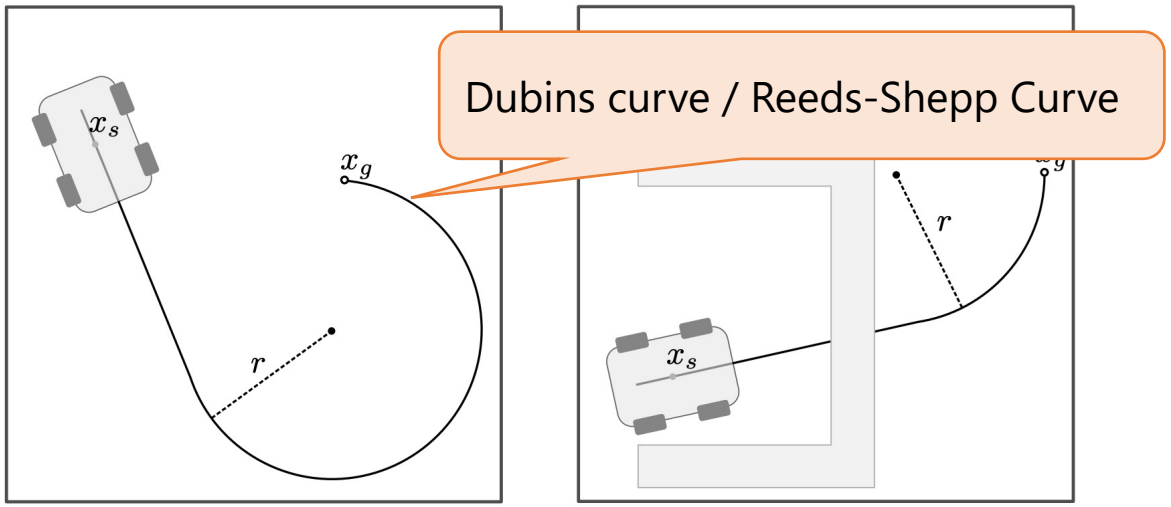
Base word	Sequences of motion primitives
$C\|C\|C$	$(L^+R^-L^+)(L^-R^+L^-)(R^+L^-R^+)(R^-L^+R^-)$
$CC\|C$	$(L^+R^+L^-)(L^-R^-L^+)(R^+L^+R^-)(R^-L^-R^+)$
$C\|CC$	$(L^+R^-L^-)(L^-R^+L^+)(R^+L^-R^-)(R^-L^+R^+)$
CSC	$(L^+S^+L^+)(L^-S^-L^-)(R^+S^+R^+)(R^-S^-R^-)$ $(L^+S^+R^+)(L^-S^-R^-)(R^+S^+L^+)(R^-S^-L^-)$
$CC_\beta\|C_\beta C$	$(L^+R_\beta^+L_\beta^-R^-)(L^-R_\beta^-L_\beta^+R^+)(R^+L_\beta^+R_\beta^-L^-)(R^-L_\beta^-R_\beta^+L^+)$
$C\|C_\beta C_\beta\|C$	$(L^+R_\beta^-L_\beta^-R^+)(L^-R_\beta^+L_\beta^+R^-)(R^+L_\beta^-R_\beta^-L^+)(R^-L_\beta^+R_\beta^+L^-)$
$C\|C_{\pi/2}SC$	$(L^+R_{\pi/2}^-S^-R^-)(L^-R_{\pi/2}^+S^+R^+)(R^+L_{\pi/2}^-S^-L^-)(R^-L_{\pi/2}^+S^+L^+)$ $(L^+R_{\pi/2}^-S^-L^-)(L^-R_{\pi/2}^+S^+L^+)\ (R^+L_{\pi/2}^-S^-R^-)(R^-L_{\pi/2}^+S^+R^+)$
$CSC_{\pi/2}\|C$	$(L^+S^+L_{\pi/2}^+R^-)(L^-S^-L_{\pi/2}^-R^+)(R^+S^+R_{\pi/2}^+L^-)(R^-S^-R_{\pi/2}^-L^+)$ $(R^+S^+L_{\pi/2}^+R^-)(R^-S^-L_{\pi/2}^-R^+)(L^+S^+R_{\pi/2}^+L^-)(L^-S^-R_{\pi/2}^-L^+)$
$C\|C_{\pi/2}SC_{\pi/2}\|C$	$(L^+R_{\pi/2}^-S^-L_{\pi/2}^-R^+)(L^-R_{\pi/2}^+S^+L_{\pi/2}^+R^-)$ $(R^+L_{\pi/2}^-S^-R_{\pi/2}^-L^+)(R^-L_{\pi/2}^+S^+R_{\pi/2}^+L^-)$

https://lavalle.pl/planning/node822.html

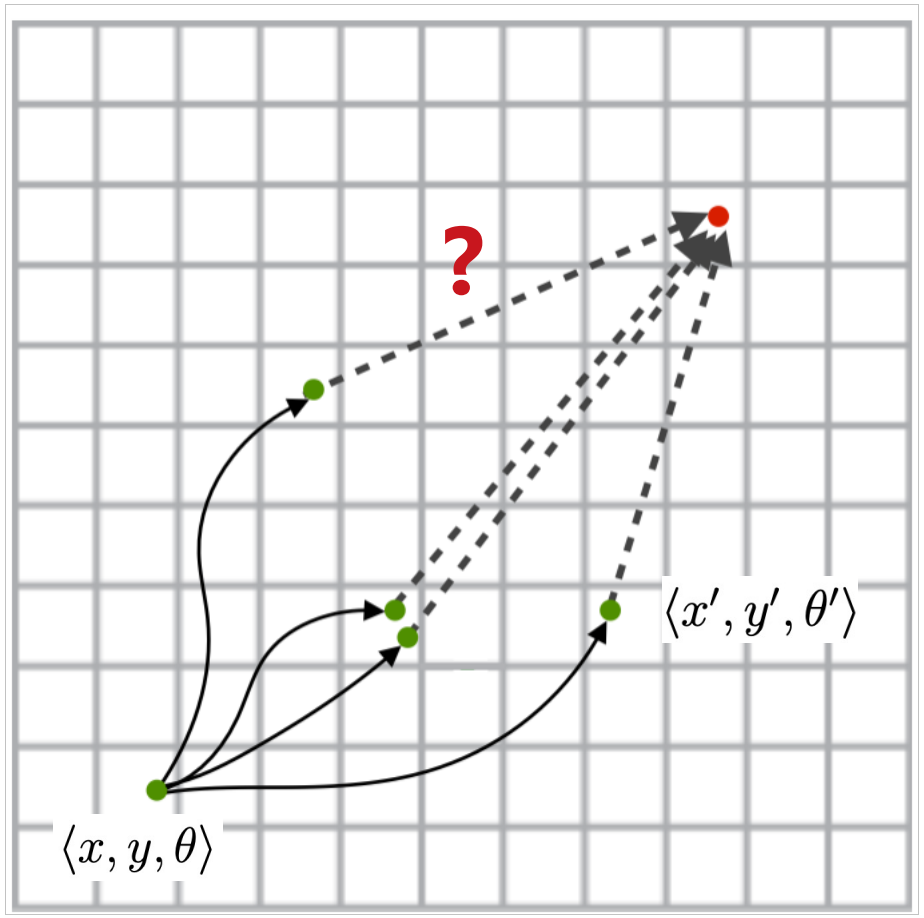
https://github.com/nathanlct/reeds-shepp-curves

Hybrid A* Heuristics

- 启发函数 <- relaxed problem
- **启发1**：Nonholonomic without obstacles h_1
  - 保持非完整约束
  - 去掉所有障碍物

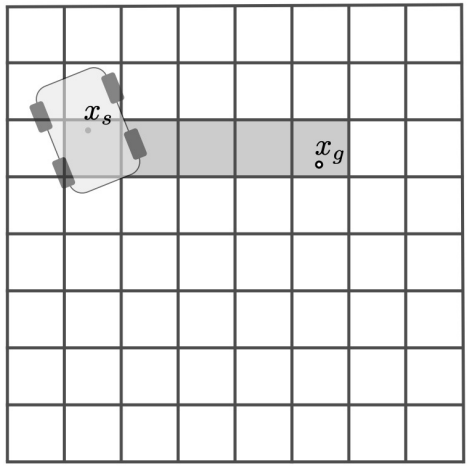


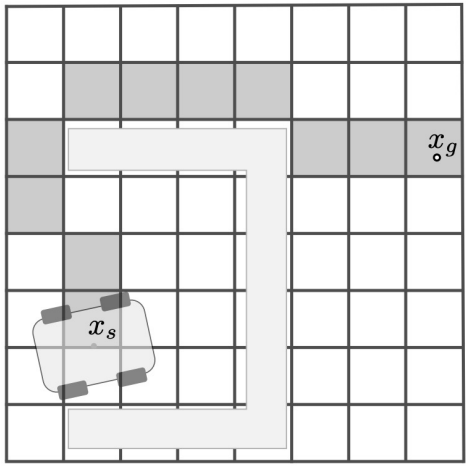
图片来自 Karl's master thesis



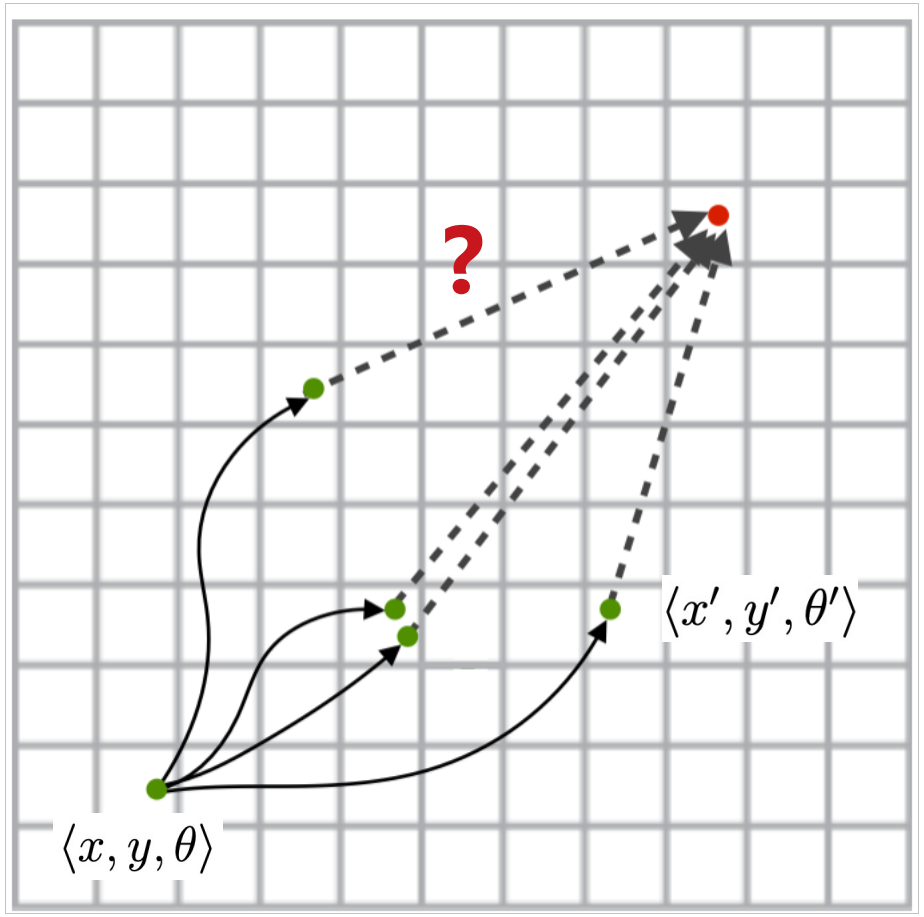
Hybrid A* Heuristics

- 启发函数 <- relaxed problem
- **启发2**：Holonomic with obstacles h_2
  - 去掉运动学限制
  - 保留障碍物
  - -> A* ，D* Lite



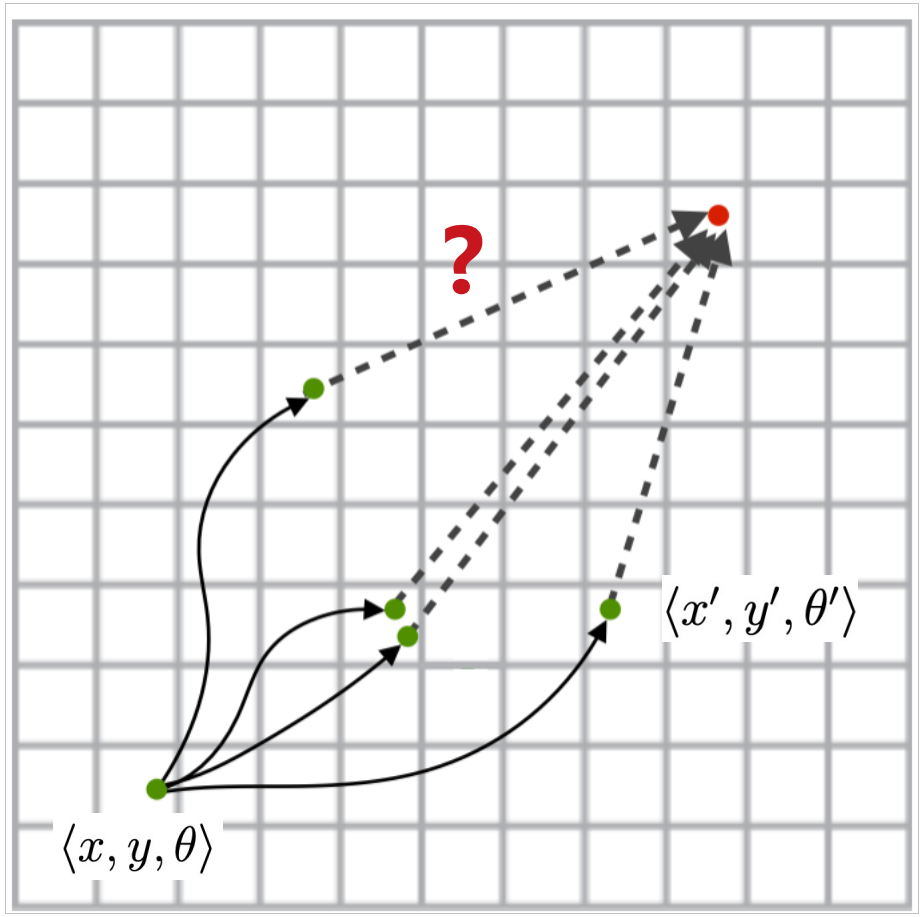


图片来自 Karl's master thesis

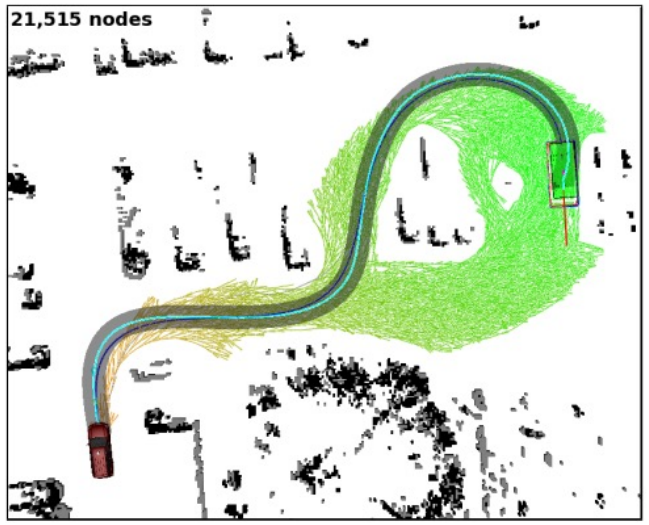


Hybrid A* Heuristics

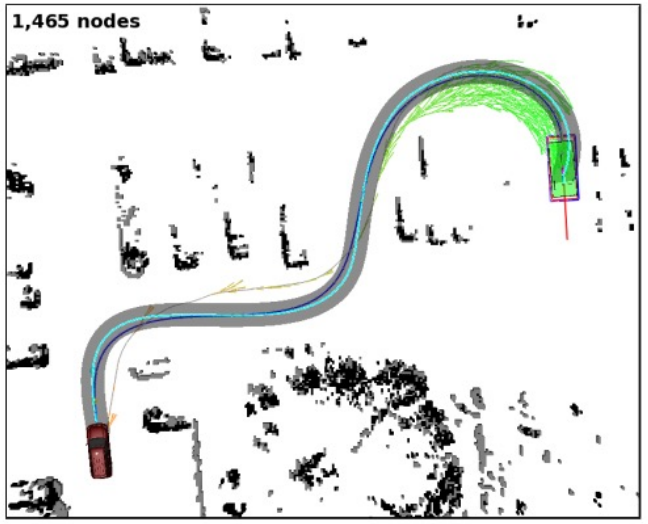
- 启发函数 <- relaxed problem:
- **启发1**：Nonholonomic without obstacles $h_1(q)$
  - 保持非完整约束
  - 去掉所有障碍物
- **启发2**：Holonomic with obstacles $h_2(q)$
  - 去掉运动学限制 -> A* ，D* Lite
  - 保留障碍物
- **合并启发**：
  - $\boldsymbol{h(q) = max\{h_1(q), h_2(q)\}}$
  - $h(q)$ 一定比 $h_1(q)$ 和 $h_2(q)$ 更好



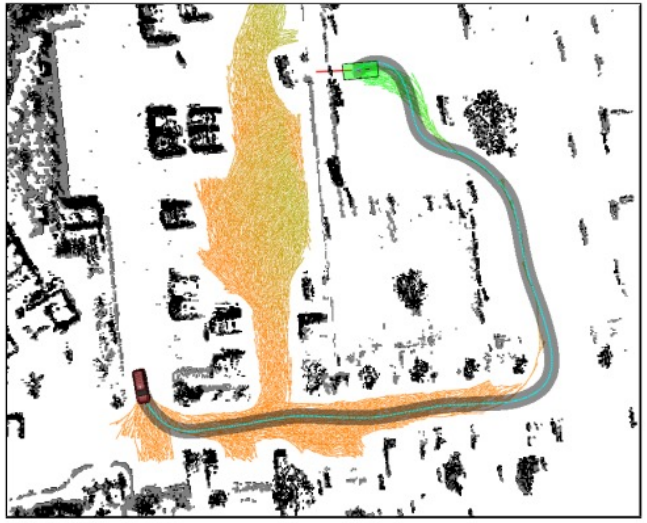
Hybrid A* Heuristics



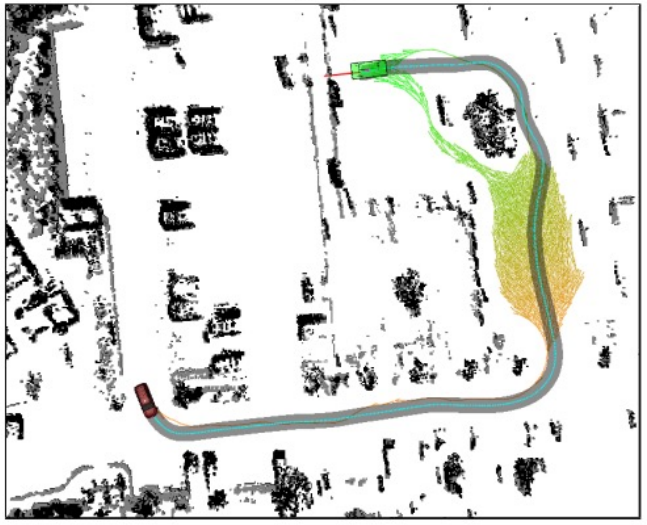
Euclidean Distance



Nonholonomic without obstacles



Nonholonomic without obstacles



+ Holonomic with obstacles

Hybrid A* 效果（DARPA Urban Challenge 2007）

Developed by Dmitri Dolgov, who currently serves as the co-chief executive officer of Waymo

Free-form path planner used by Junior from Standford: https://youtu.be/qXZt-B7iUyw

第八讲总结

- 层级式自动驾驶规划
  - 任务规划：“城市地图中怎么从A到B”
  - 行为规划：“车流中，何时进行跟车、变道、超车、转向…”
  - 运动规划：“以什么轨迹进行变道、超车，…，停车、绕行，…”
    - 路径规划
    - 速度曲线生成
  - 运动控制：“如何控制方向盘、油门、刹车以精准地跟踪参考轨迹”
- 图搜索算法（任务规划、路径规划）
  - 静态图：A*
  - 动态图：LPA*，D* Lite
  - 自动驾驶专用：Hybrid A*

第八讲扩展阅读

- **A***：Hart, Peter E., Nils J. Nilsson, and Bertram Raphael. "A formal basis for the heuristic determination of minimum cost paths." IEEE transactions on Systems Science and Cybernetics 4, no. 2 (1968): 100-107.
- **Lifelong Planning A***：Koenig, Sven, Maxim Likhachev, and David Furcy. "Lifelong planning A∗." Artificial Intelligence 155, no. 1-2 (2004): 93-146.
- **D* Lite**：Koenig, Sven, and Maxim Likhachev. "Fast replanning for navigation in unknown terrain." IEEE transactions on robotics 21, no. 3 (2005): 354-363.
- **Hybrid A***：Montemerlo, Michael, Jan Becker, Suhrid Bhat, Hendrik Dahlkamp, Dmitri Dolgov, Scott Ettinger, Dirk Haehnel et al. "Junior: The stanford entry in the urban challenge." Journal of field Robotics 25, no. 9 (2008): 569-597.
- **Hybrid A***：Dolgov, Dmitri, Sebastian Thrun, Michael Montemerlo, and James Diebel. "Path planning for autonomous vehicles in unknown semi-structured environments." The international journal of robotics research 29, no. 5 (2010): 485-501.
- **Parking Planner based on Hybrid A***: Li, Bai, Tankut Acarman, Youmin Zhang, Yakun Ouyang, Cagdas Yaman, Qi Kong, Xiang Zhong, and Xiaoyan Peng. "Optimization-based trajectory planning for autonomous parking with irregularly placed obstacles: A lightweight iterative framework." IEEE Transactions on Intelligent Transportation Systems 23, no. 8 (2021): 11970-11981.

第九讲：运动规划II

蔡盼盼 副教授

上海交通大学清源研究院

研究领域：机器人规划、机器人学习、自动驾驶

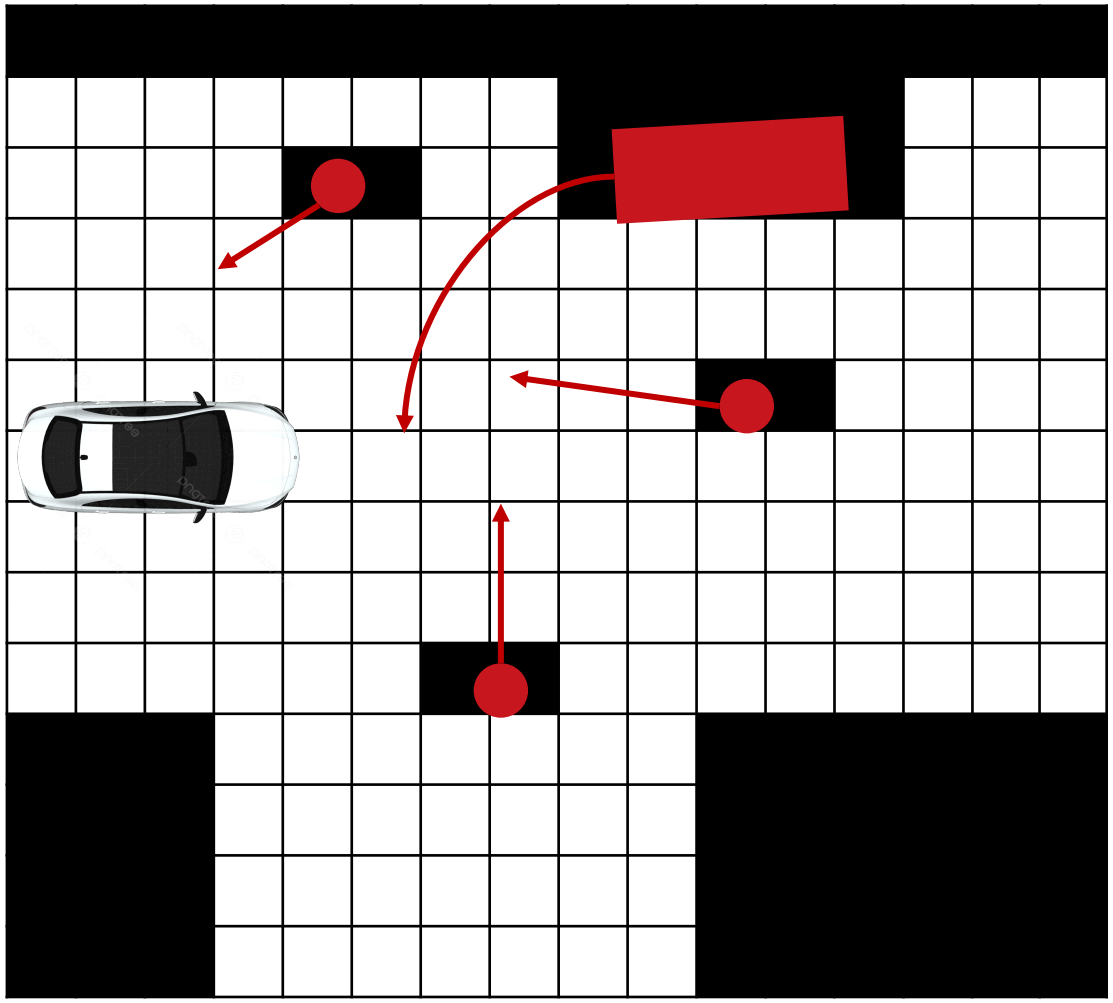
邮箱：Cai_panpan@sjtu.edu.cn

网站：https://cindycia.github.io/

动态场景路径规划？

- **思路1**：Hybrid D* Lite?
  - 每次执行完一段路径
  - 接受一次环境更新
  - 迅速地重新规划

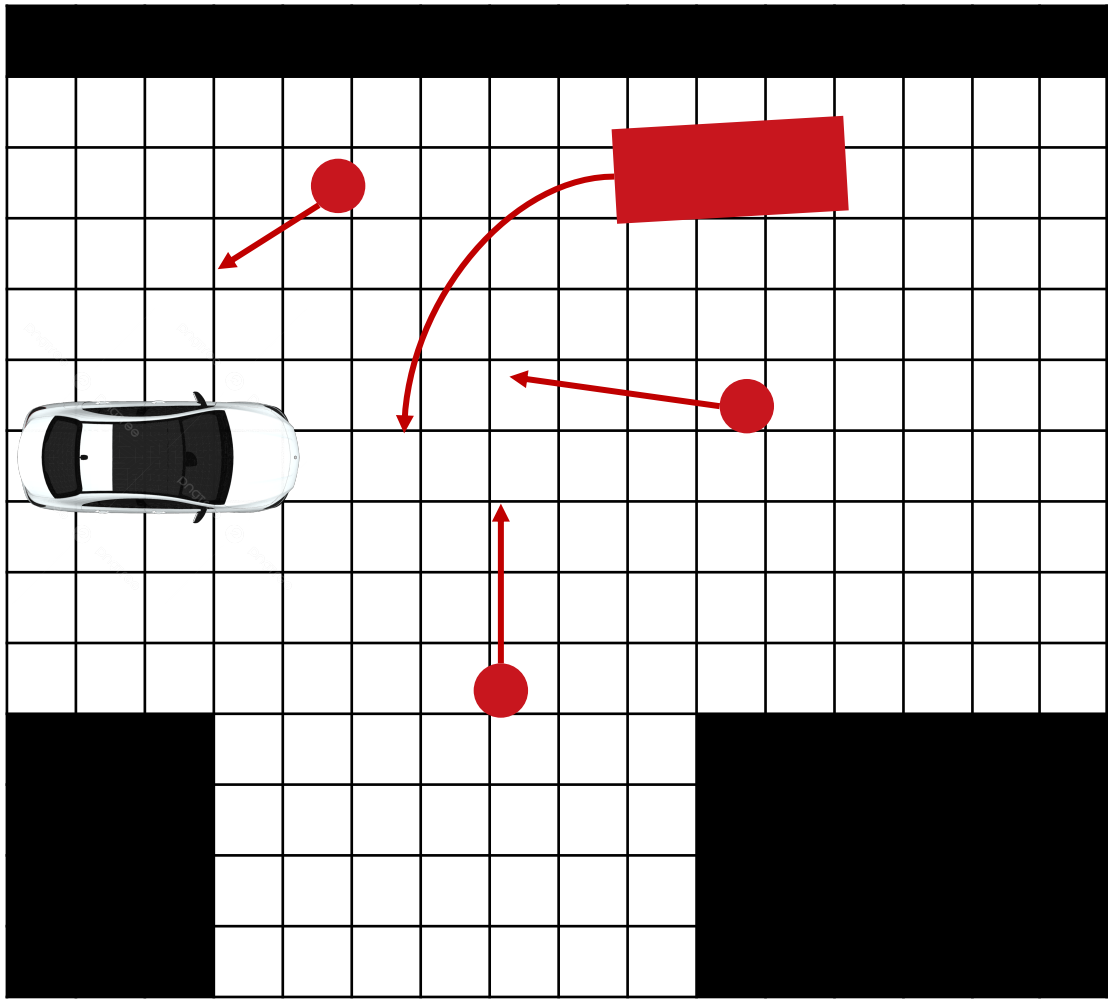
- **问题**：不能对危险进行提前反应
  - 只对已经观察到的变化做出反应
  - 不会预测未来的运动，提前应对



动态场景路径规划？

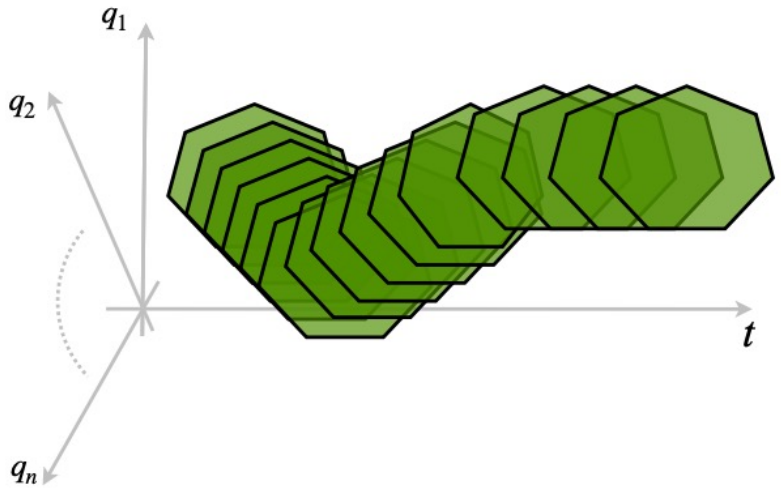
- **思路2**：联合状态空间（joint state space）
  - $q = (q_{ego}, q^1_{exo}, q^2_{exo}, q^3_{exo}, q^4_{exo})$
  - $q'_{ego} = f(q_{ego}, u)$
  - $q'_{exo} = m(q_{exo})$
  - 进行 hybrid A* 搜索

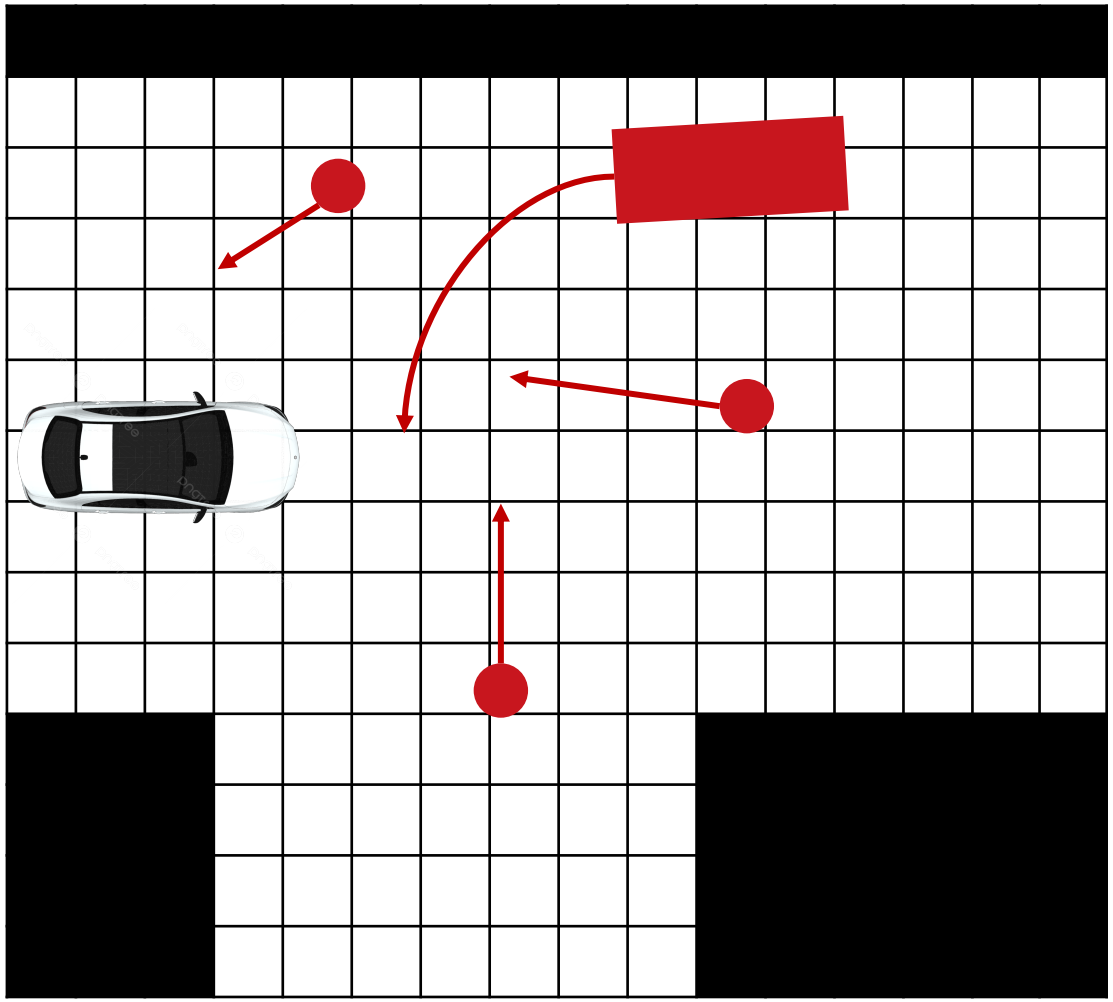
- **问题**：栅格维度太高！
  - 搜索具有指数复杂度（worst-case）



动态场景路径规划？

- **思路3**：时空搜索（Space-time search）
  - **5D 时空栅格**：向状态空间中引入时间维度
  - **扩展**：$< q', t+1 > = < F(q,u), t >$
  - **碰撞检测**：
    - **预测** $(t, t+1)$ 时间段内他人/车的位置
    - 将他人/车与自车进行**碰撞检测**
    - 若整个时间段都没有碰撞，才接受扩展





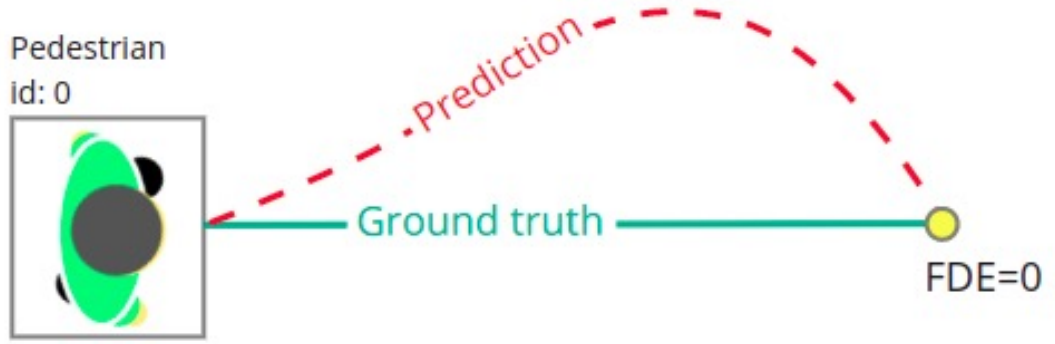
运动预测问题（Motion Prediction Problem）

- 输入：
  - 动态障碍物类型、几何特征
  - 动态障碍物当前状态
  - 动态障碍物历史轨迹
  - （可选）高清地图
  - （可选）当前感知数据
- 输出：
  - 动态障碍物未来轨迹
    - 参数曲线 $< x(t), y(t), \theta(t) >$ 的参数，或
    - 离散点序列 $\{< x_1, y_1, \theta_1 >, < x_2, y_2, \theta_2 >, \ldots, < x_T, y_T, \theta_T >\}$

运动预测评价指标

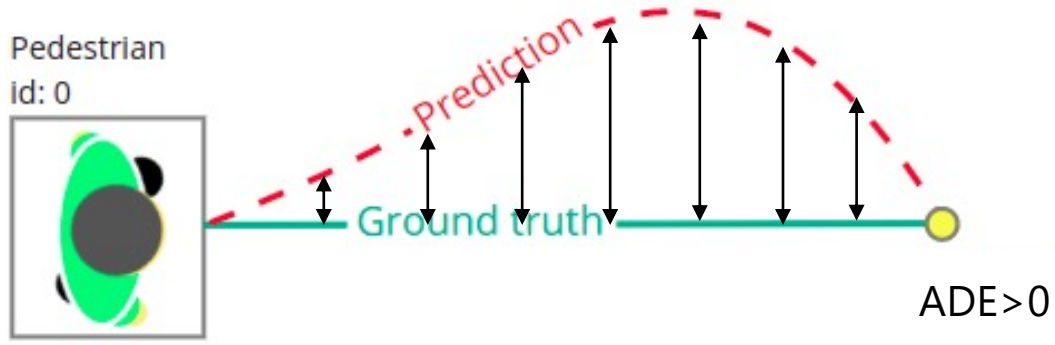
- 评价指标一：Final Displacement Error

$$FDE = \frac{\sum_{i=1}^{n} \sqrt{(\hat{x}_i^{T_{pred}} - x_i^{T_{pred}})^2 + (\hat{y}_i^{T_{pred}} - y_i^{T_{pred}})^2}}{n}$$



- 评价指标二：Average Displacement Error

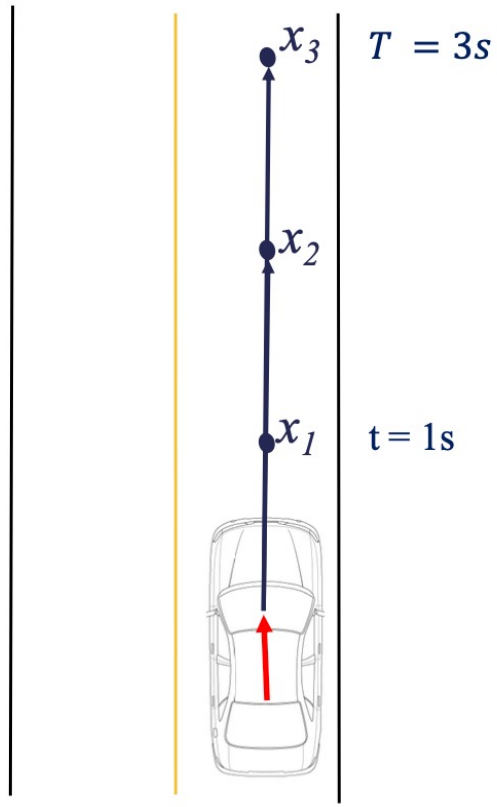
$$ADE = \frac{\sum_{i=1}^{n} \sum_{t=T_{obs}+1}^{T_{pred}} \left[(\hat{x}_i^t - x_i^t)^2 + (\hat{y}_i^t - y_i^t)^2 \right]}{n(T_{pred} - (T_{obs} + 1))}$$



运动预测模型（Motion Prediction Models）

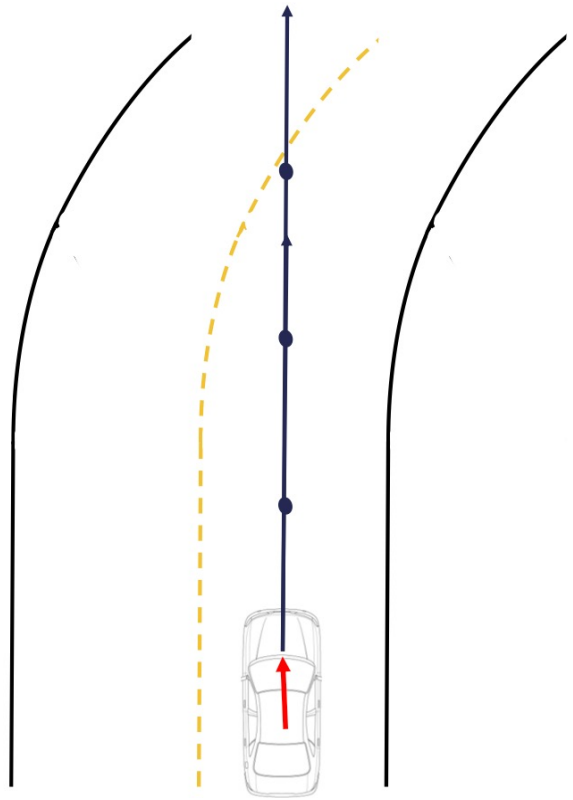
- 简单模型 Const-vel, const-speed
- 解析模型 Social forces, ORCA, ….
- 深度学习方法 CNN, LSTM, GNN, Transformer, …

匀速运动模型（Constant Velocity Model）



Algorithm Constant Velocity Prediction(x_{obj})

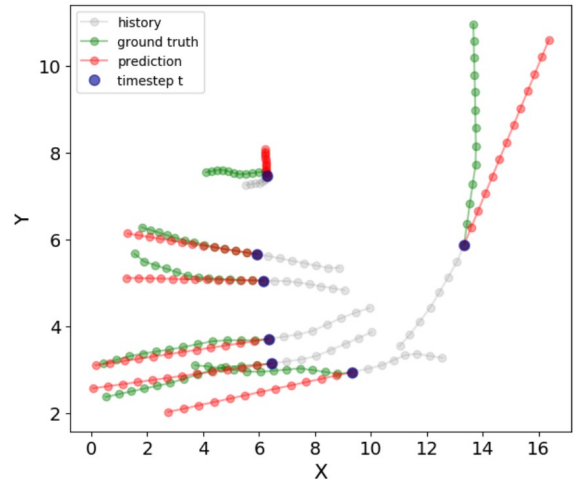
1. $t \leftarrow 0$
2. $x_0 = x_{obj}$
3. **while** $t * dt < T$ **do**
4. $t = t + 1$
5. $x_t.\text{pos} \leftarrow x_{t-1}.\text{pos} + dt * x_{t-1}.\text{vel}$
6. $x_t.vel \leftarrow x_{t-1}.\text{vel}$
7. **end while**
8. **return** $x_{1:T}$

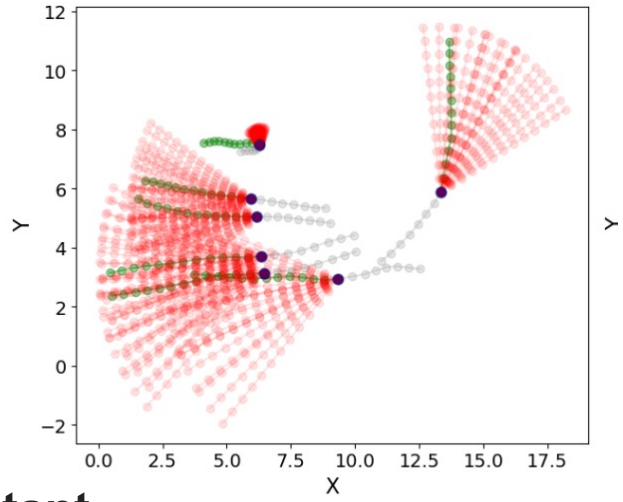


匀速运动模型（Constant Velocity Model）

- 预测行人运动： Const-vel vs. 深度学习

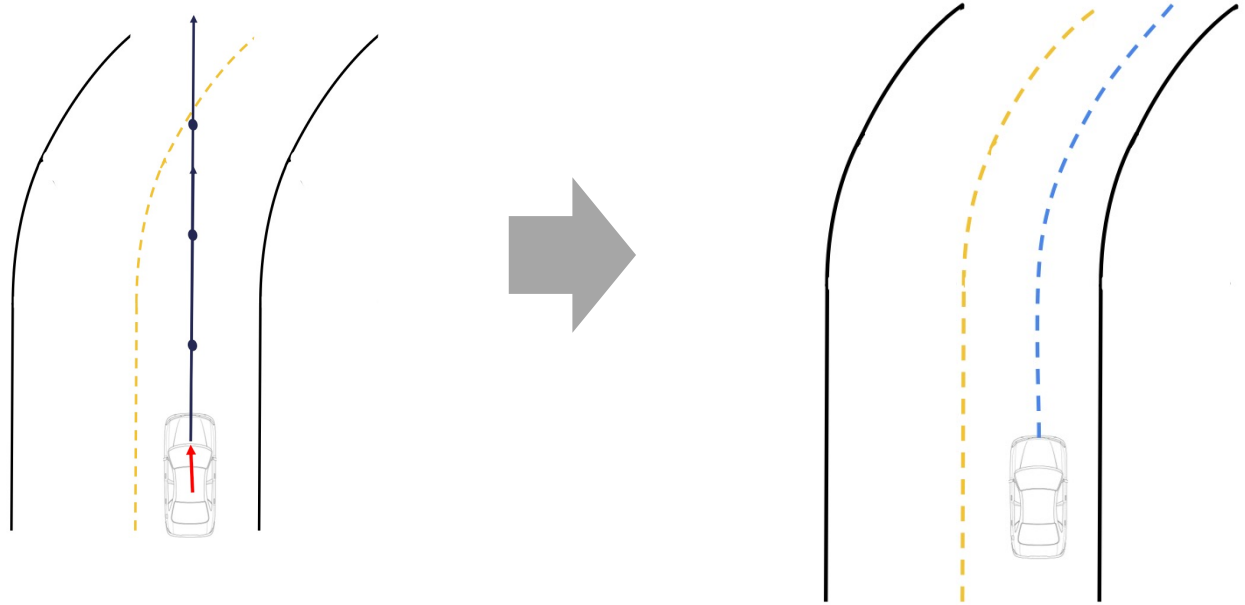
Metric	Dataset	ConstAcc	Lin	FF	LSTM	RED	SR-LSTM	OUR	SoPhie	S-GAN	OUR-S
ADE	ETH-Uni	1.35	0.58	0.67	0.54	0.60	0.63	0.58	0.70	0.59	**0.43**
	Hotel	0.95	0.39	1.59	2.91	0.47	0.37	0.27	0.76	0.38	**0.19**
	Zara1	0.59	0.44	0.39	0.34	0.34	0.41	0.34	0.30	**0.18**	0.24
	Zara2	0.50	0.41	0.38	0.43	0.31	0.32	0.31	0.38	**0.18**	0.21
	UCY-Uni	0.79	0.60	0.69	0.72	0.47	0.51	0.46	0.54	**0.26**	0.34
AVG		0.84	0.48	0.74	0.99	0.44	0.45	0.39	0.54	0.32	**0.28**
FDE	ETH-Uni	3.29	1.11	1.32	1.04	1.14	1.25	1.15	1.43	1.04	**0.80**
	Hotel	2.41	0.81	3.12	6.07	0.94	0.74	0.51	1.67	0.79	**0.35**
	Zara1	1.50	0.93	0.81	0.74	0.76	0.90	0.76	0.63	**0.32**	0.48
	Zara2	1.30	0.83	0.77	0.95	0.70	0.70	0.69	0.78	**0.34**	0.45
	UCY-Uni	2.03	1.19	1.38	1.60	1.00	1.10	1.02	1.24	**0.49**	0.71
AVG		2.11	0.97	1.48	2.08	0.91	0.94	0.83	1.15	0.60	**0.56**

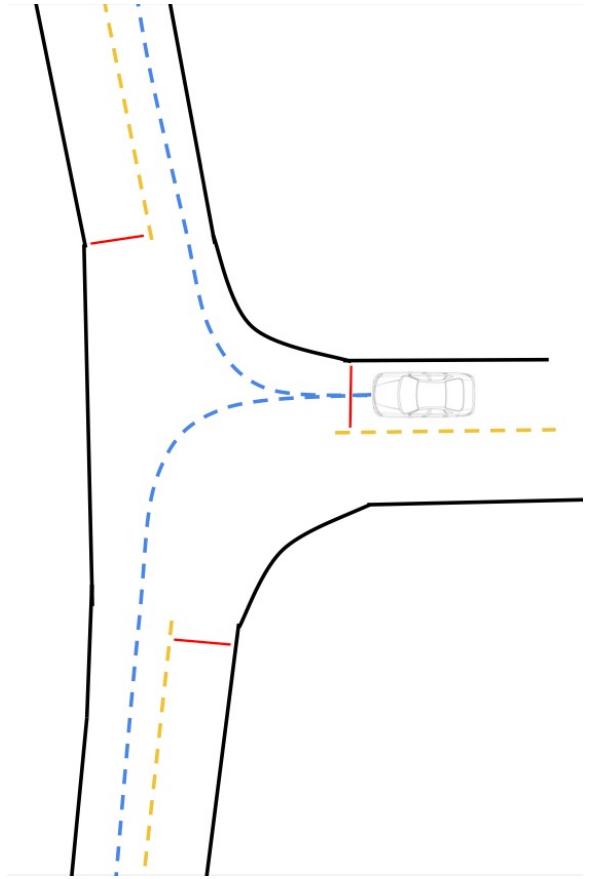




Schöller, Christoph, Vincent Aravantinos, Florian Lay, and Alois Knoll. "**What the constant velocity model can teach us about pedestrian motion prediction**." *IEEE Robotics and Automation Letters* 5, no. 2 (2020): 1696-1703.

匀速运动模型+地图信息（Constant Speed Model）





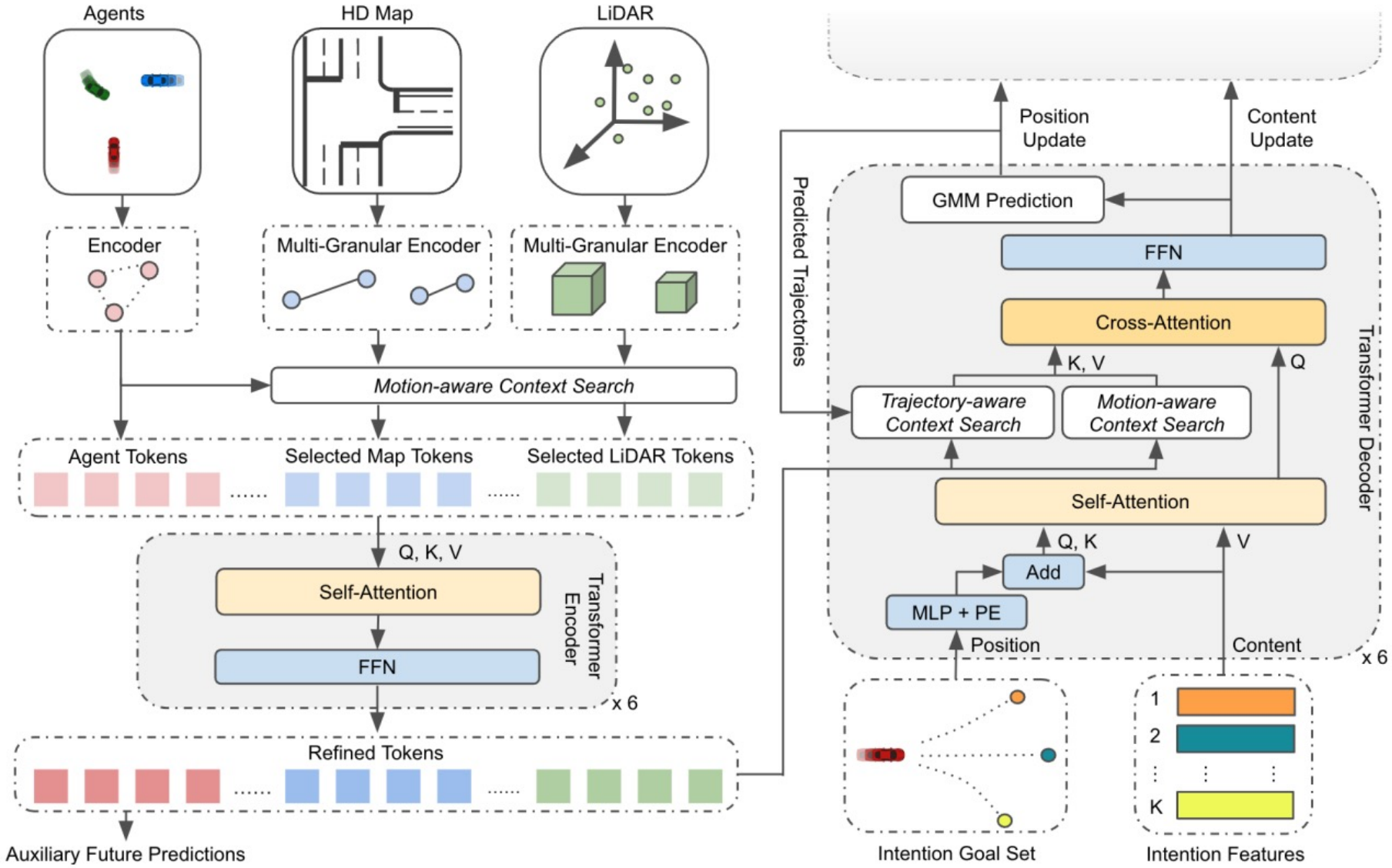
缺点：

- 缺乏交互
- 无法预测复杂行为

深度学习运动预测模型

MRTR

（2023 Waymo Open Dataset 1st place）

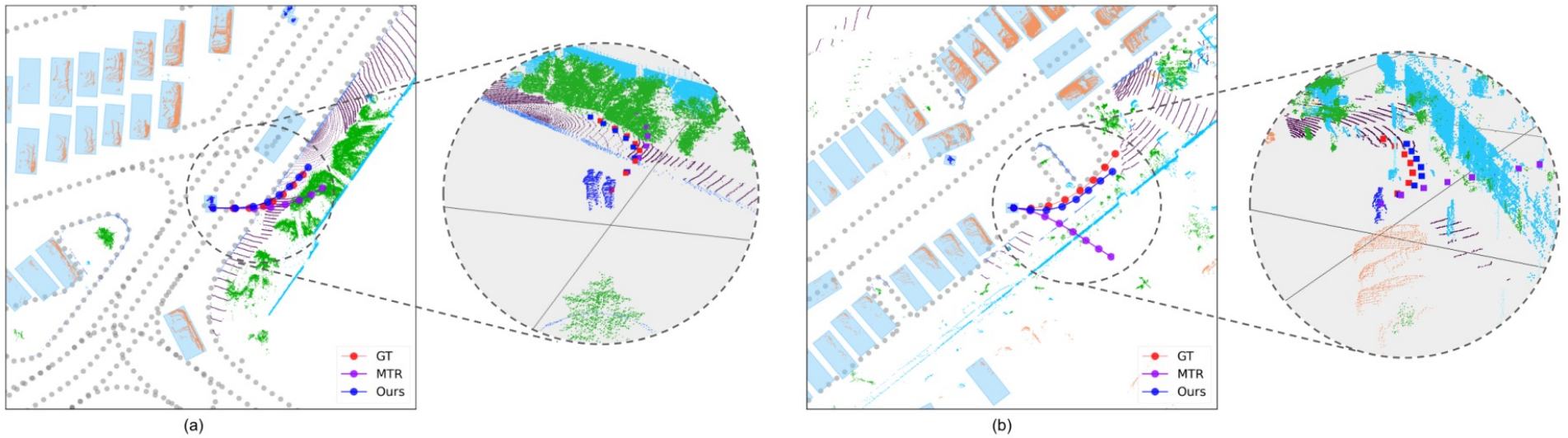


Gan, Yiqian, Hao Xiao, Yizhe Zhao, Ethan Zhang, Zhe Huang, Xin Ye, and Lingting Ge. "MGTR: Multi-granular transformer for motion prediction with lidar." *arXiv preprint arXiv:2312.02409* (2023).

深度学习运动预测模型

Method Name	Ob... type	Eva... time	Soft mAP	mAP	minADE	minFDE	Miss rate	Overlap Rate	Date (Pacific Daylight Time)
	All	Avg	Show						
MGTR_ens	All	Avg	0.4764	0.4658	0.5825	1.2009	0.1258	0.1270	9/16/2023, 10:06:08 AM

MRTR
（2023 Waymo Open Dataset 1st place）



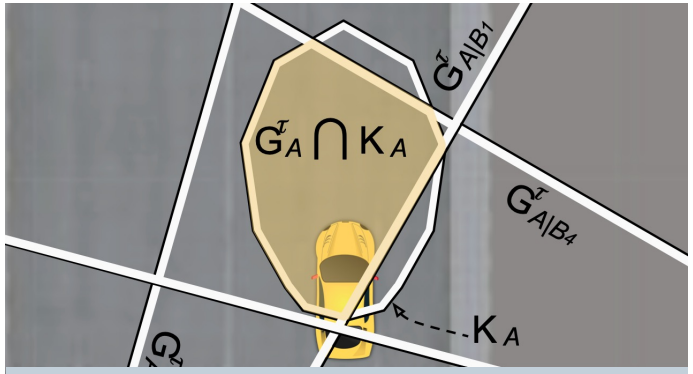
- **优势**：数据覆盖越稠密，预测效果越好
- **劣势**：慢，难以支持实时规划
  - 0.1s 内需要完成预测+规划
  - 若要考虑未来自车行为对他人轨迹的影响，需要再每个节点重新调用一次运动预测！

解析运动模型

GAMMA [RAL'22]

$$v' = \underset{v \in G(\theta) \cap K(\theta)}{\operatorname{argmin}} \|v - v^{I(\theta)}\|^2$$

交互避障　运动学结构　行为意图



内隐参数：意图、注意力、合作度

通过对内隐参数进行概率推断，提供概率性预测

更准确的多模态预测

Distributional Models					
SGAN	SoPhie	深度学习模型	S-Ways	SGCN	GAMMA
0.81/1.52	0.70/1.43	1.01/1.75	0.39/0.64	0.63/1.03	**0.30/0.65**
0.72/1.61	0.76/1.67	0.43/0.80	0.39/0.66	0.32/0.55	**0.18/0.40**
0.60/1.26	0.54/1.24	0.44/0.91	0.55/1.31	0.37/0.70	**0.32/0.79**
0.34/0.69	0.30/0.63	0.26/0.45	0.44/0.64	0.29/0.53	**0.24/0.57**
0.42/0.84	0.38/0.78	0.26/0.57	0.51/0.92	0.25/0.45	**0.19/0.46**

运行时间 0.392ms

	SGAN ($bs = 1$)	SGAN ($bs = 4$)	SRLSTM ($bs = 1$)	SRLSTM ($bs = 4$)	GAMMA
		深度学习模型			
Time	3.15	1.80	2.32	2.19	**0.392**
Speed-up	1x	1.75x	1.36x	1.44x	**8.04x**

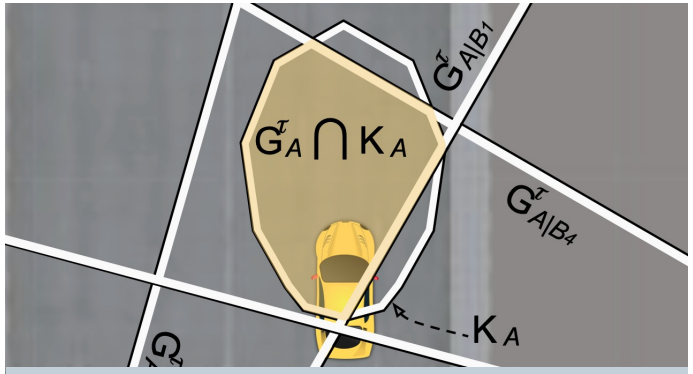
Luo, Yuanfu, Panpan Cai, Yiyuan Lee, and David Hsu. "Gamma: A general agent motion model for autonomous driving." *IEEE Robotics and Automation Letters* 7, no. 2 (2022): 3499-3506.

解析运动模型

GAMMA [RAL'22]

$$v' = \underset{v \in G(\theta) \cap K(\theta)}{\operatorname{argmin}} \|v - v^{I(\theta)}\|^2$$

交互避障　运动学结构　行为意图



内隐参数：意图、注意力、合作度

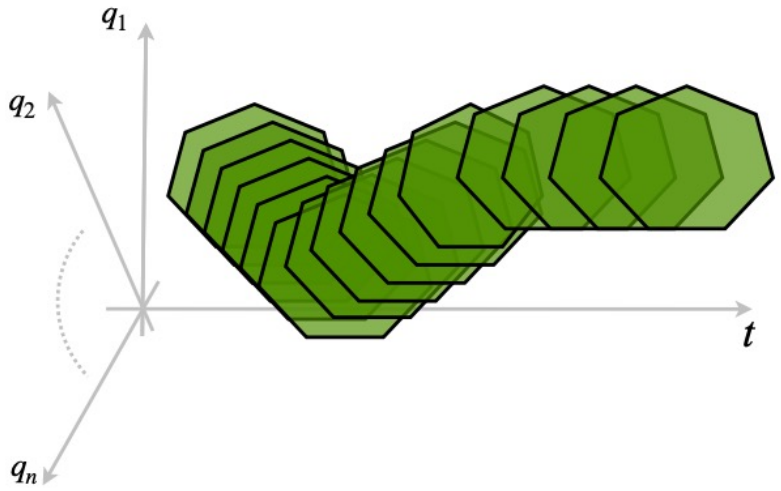
通过对内隐参数进行概率推断，提供概率性预测

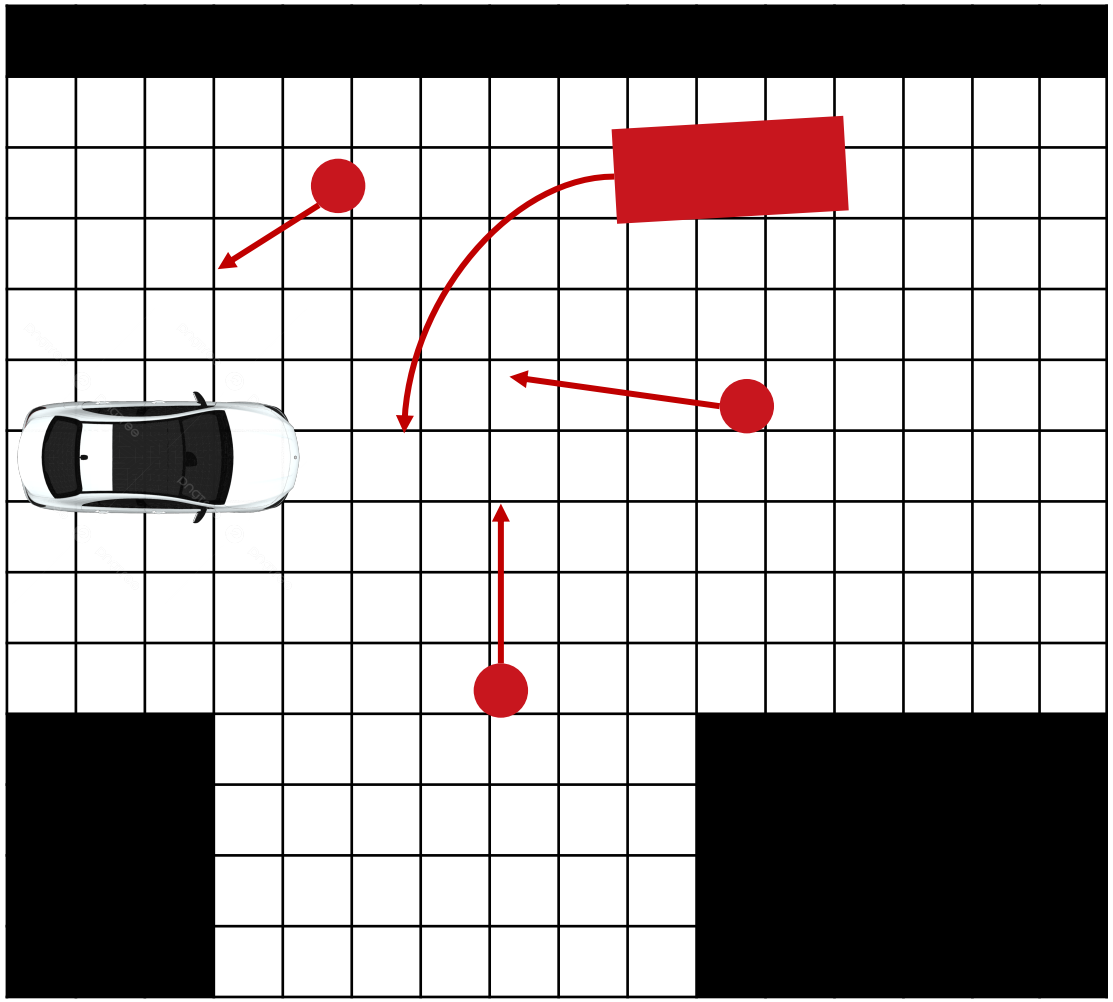
- **优势**：快，可以支持在实时规划中大量调用
- **劣势**：无法直接利用离线数据

Luo, Yuanfu, Panpan Cai, Yiyuan Lee, and David Hsu. "Gamma: A general agent motion model for autonomous driving." *IEEE Robotics and Automation Letters* 7, no. 2 (2022): 3499-3506.

动态场景路径规划？

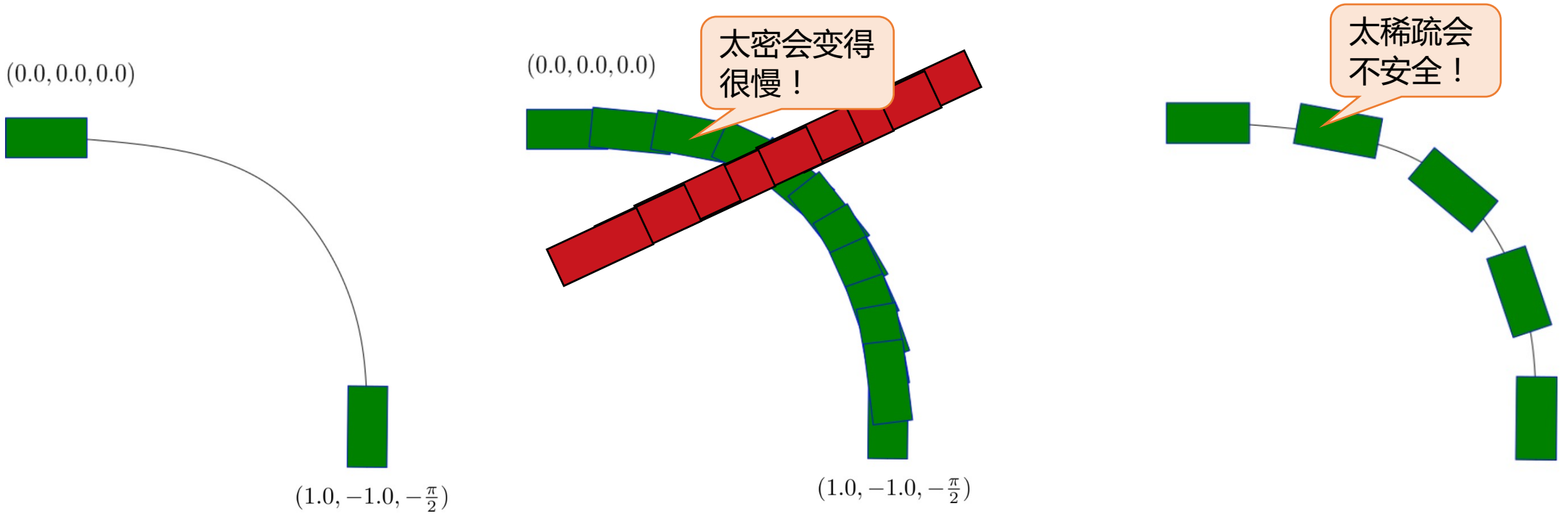
- **思路3**：时空搜索（Space-time search）
  - **5D 时空栅格**：向状态空间中引入时间维度
  - **扩展**：$< q', t+1 > = < F(q,u), t >$
  - **碰撞检测**：
    - **预测** $(t, t+1)$ 时间段内他人/车的位置
    - 将他人/车与自车轨迹进行**碰撞检测**
    - 若整个时间段都没有碰撞，才接受扩展





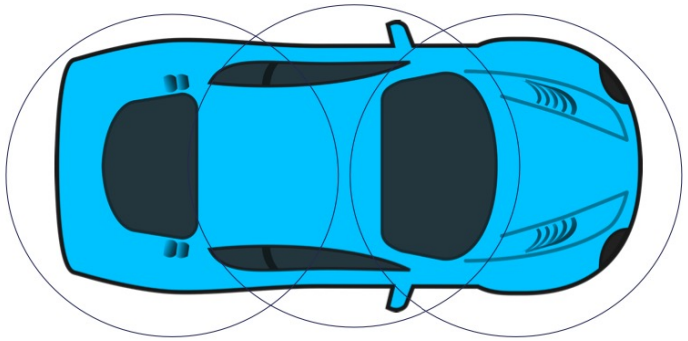
离散碰撞检测（Discrete Collision Checking）

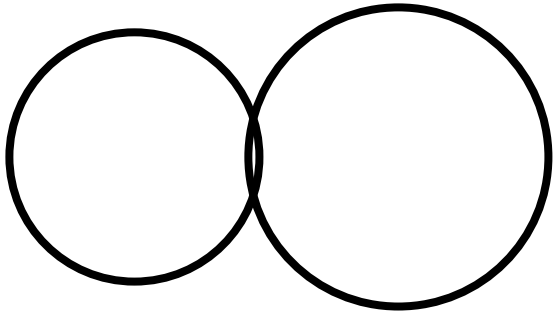
- 将轨迹离散化
- 在每个离散时刻考虑当时的几何形状，进行碰撞检测
- 需要仔细选择离散化粒度



快速碰撞检测 — 圆盘近似

- 将车体表示为三个圆盘的集合，人表示为一个圆盘





$$|x_1 - x_2|^2 + |y_1 - y_2|^2 \leq |r_1 + r_2|^2$$

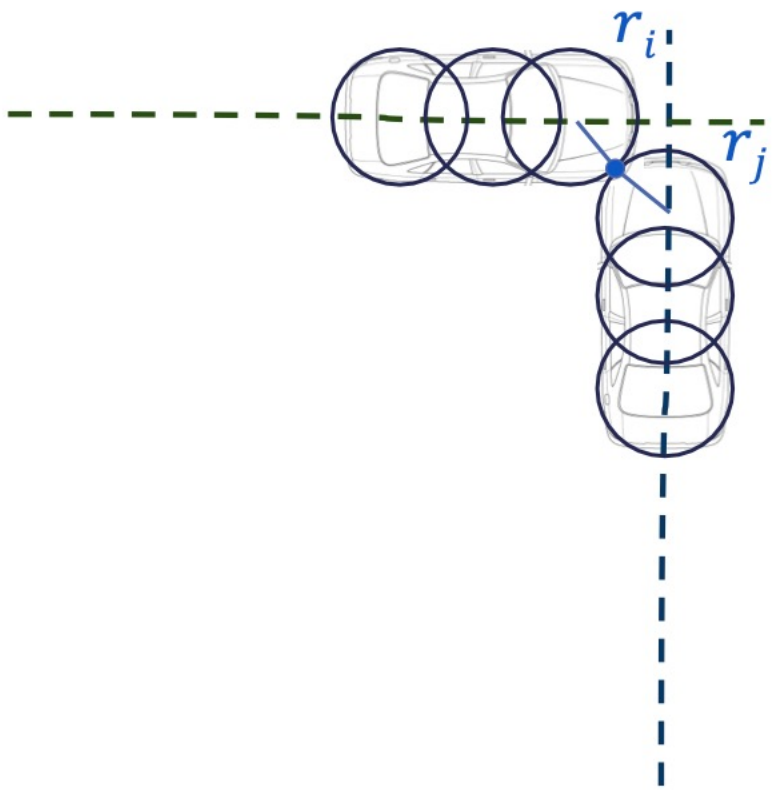
快速碰撞检测 — 圆盘近似

- 将车体表示为三个圆盘的集合
- 当任意圆盘与他人/车发生重叠时，记作碰撞：

$$d_{i,j} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}, \qquad r_i + r_j \geq d_{i,j}$$

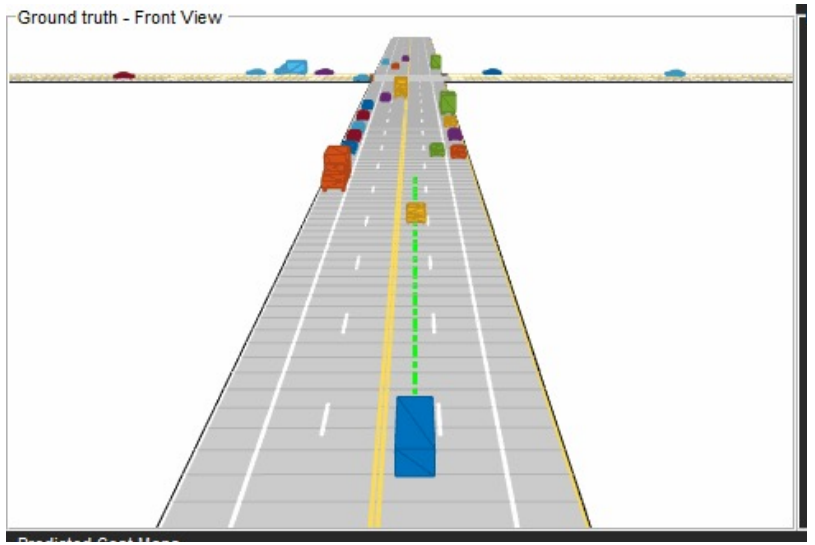
- 计算碰撞点：

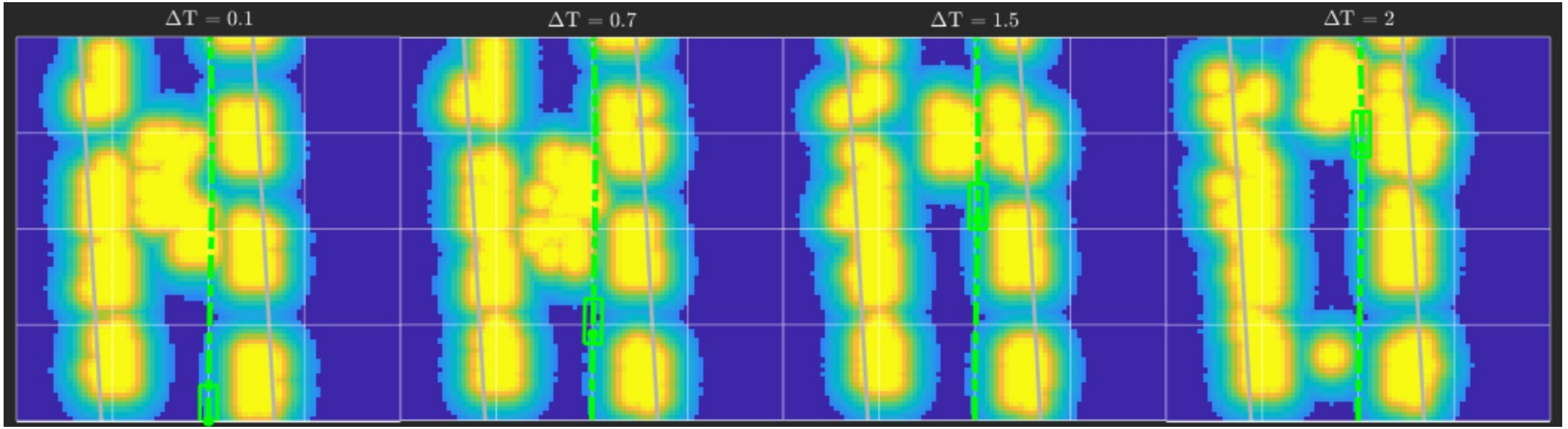
$$C_X = \frac{(x_i * r_j) + (x_j * r_i)}{(r_i + r_j)}, \qquad C_Y = \frac{(y_i * r_j) + (y_j * r_i)}{(r_i + r_j)}$$



碰撞检测 — 时空占用栅格

- 任何几何形状都可以被快速栅格化（GPU并行）
- 通过检查是否占用相同的格子进行碰撞检测
- 给每一个未来时间步生成一个占用栅格





https://www.mathworks.com/help/nav/ug/motion-planning-in-urban-environments-using-dynamics-occupancy-grid-map.html

时空占用栅格生成

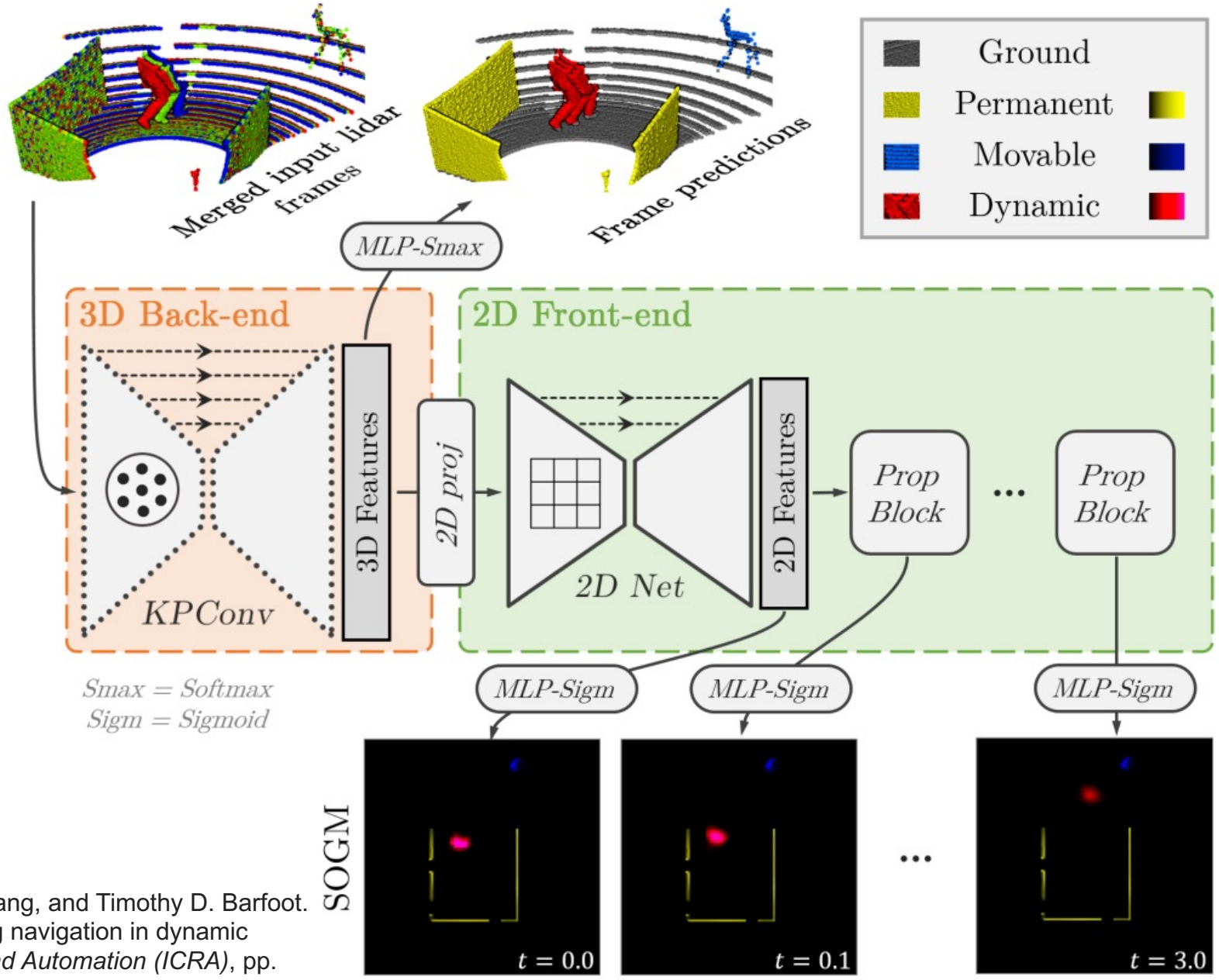
方法一：

- 在每一个时间步，用**运动模型**预测动态障碍物当时的状态，然后将更新后的几何形状投射到栅格上

时空占用栅格生成

- **方法二**：
  - 直接用深度学习生成未来的占用栅格

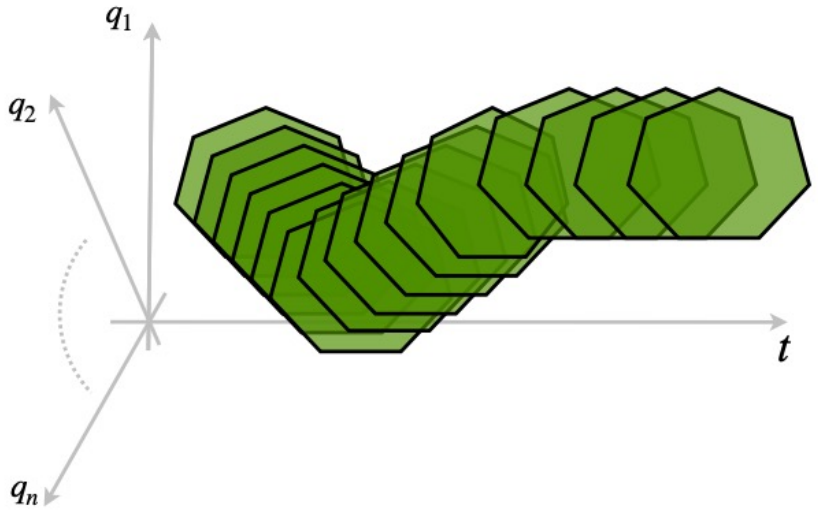
- **优势**：相关数据越多，预测效果越好
- **劣势**：慢，难以支持实时规划



Thomas, Hugues, Matthieu Gallet de Saint Aurin, Jian Zhang, and Timothy D. Barfoot. "Learning spatiotemporal occupancy grid maps for lifelong navigation in dynamic scenes." In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 484-490. IEEE, 2022.

基于 Hybrid A* 的动态场景路径规划

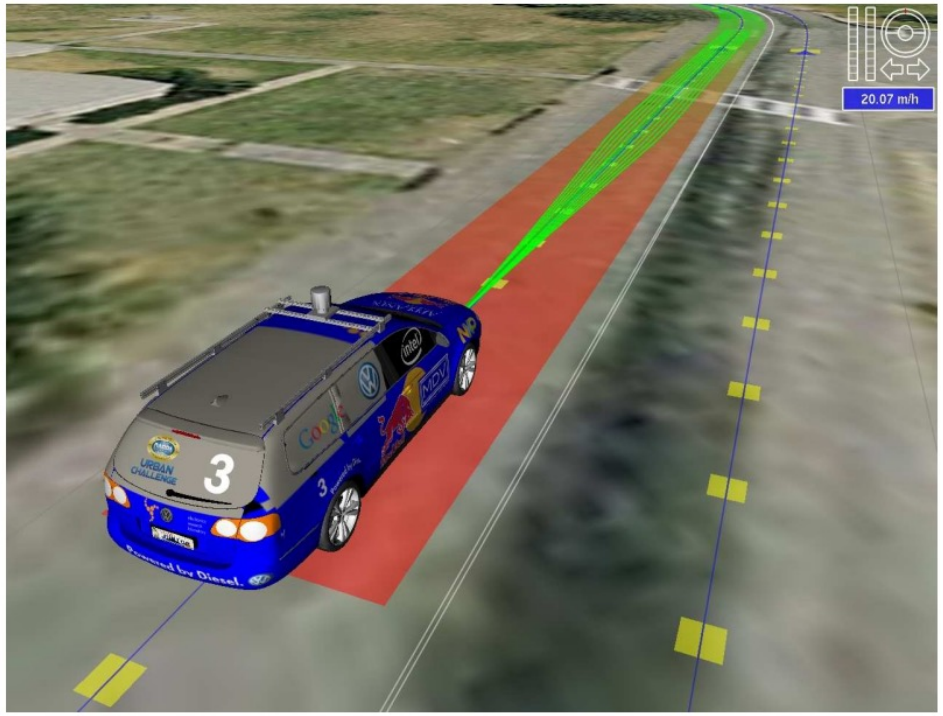
- **思路**：时空搜索（Space-time search）
  - **5D 时空栅格**：向状态空间中引入时间维度
  - **扩展**：$< q', t+1 > = < f(q,u), t >$
  - **碰撞检测**：预测+离散化 or 时空占用栅格
  - **搜索算法**：Hybrid A*



- **问题**：
  - 预测模型不准确，长远规划不可靠
  - 搜索的计算量可能超过实时规划时间
- 主要适合低速场景
  - 停车场、单行道掉头、低速避开静态障碍物

高速、动态场景路径规划

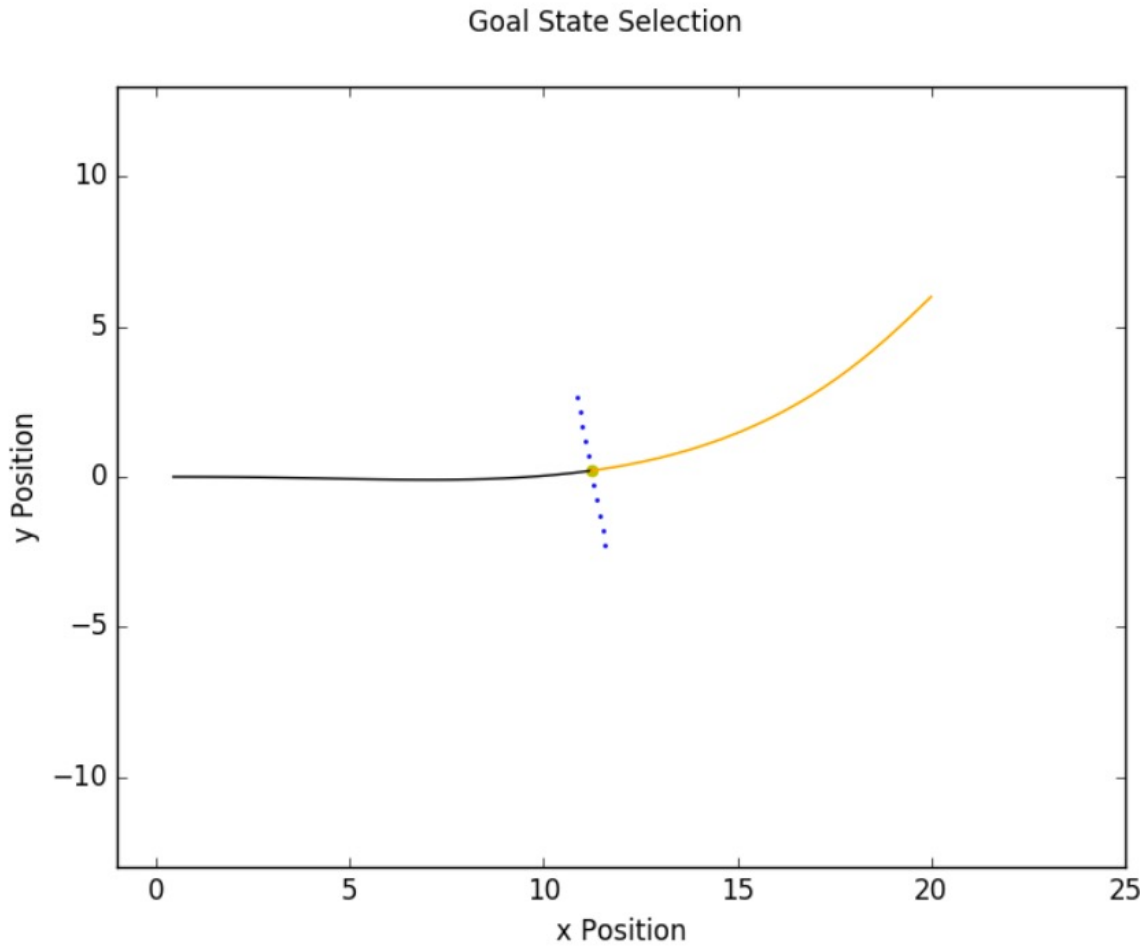
- **结构化道路**：共形格子规划器（Conformal lattice planner）



Junior from Stanford, 2007 DARPA Urban Challenge

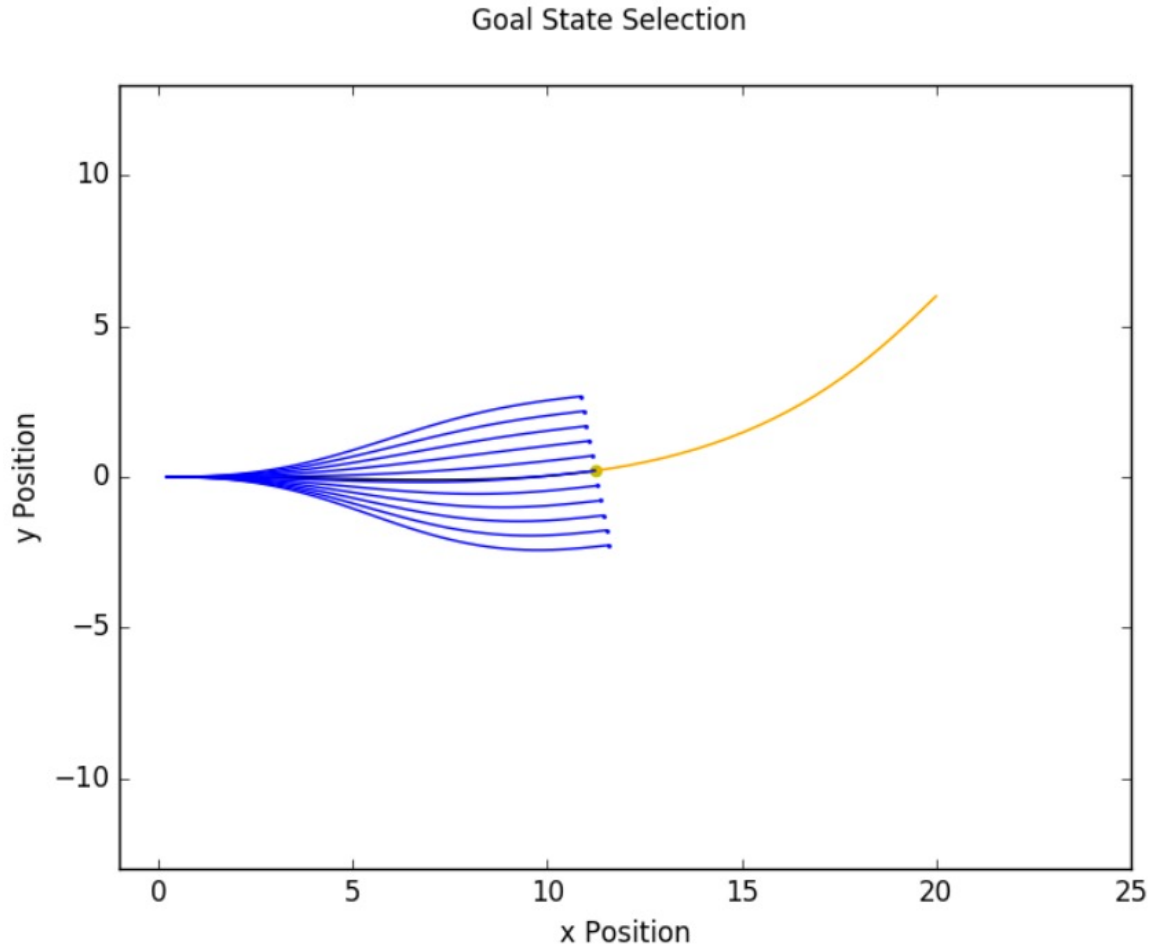
Conformal lattice planner

- 选择前方道路中心线上的目标点
- 重复N次：
  - 将目标点沿横向偏移固定的距离



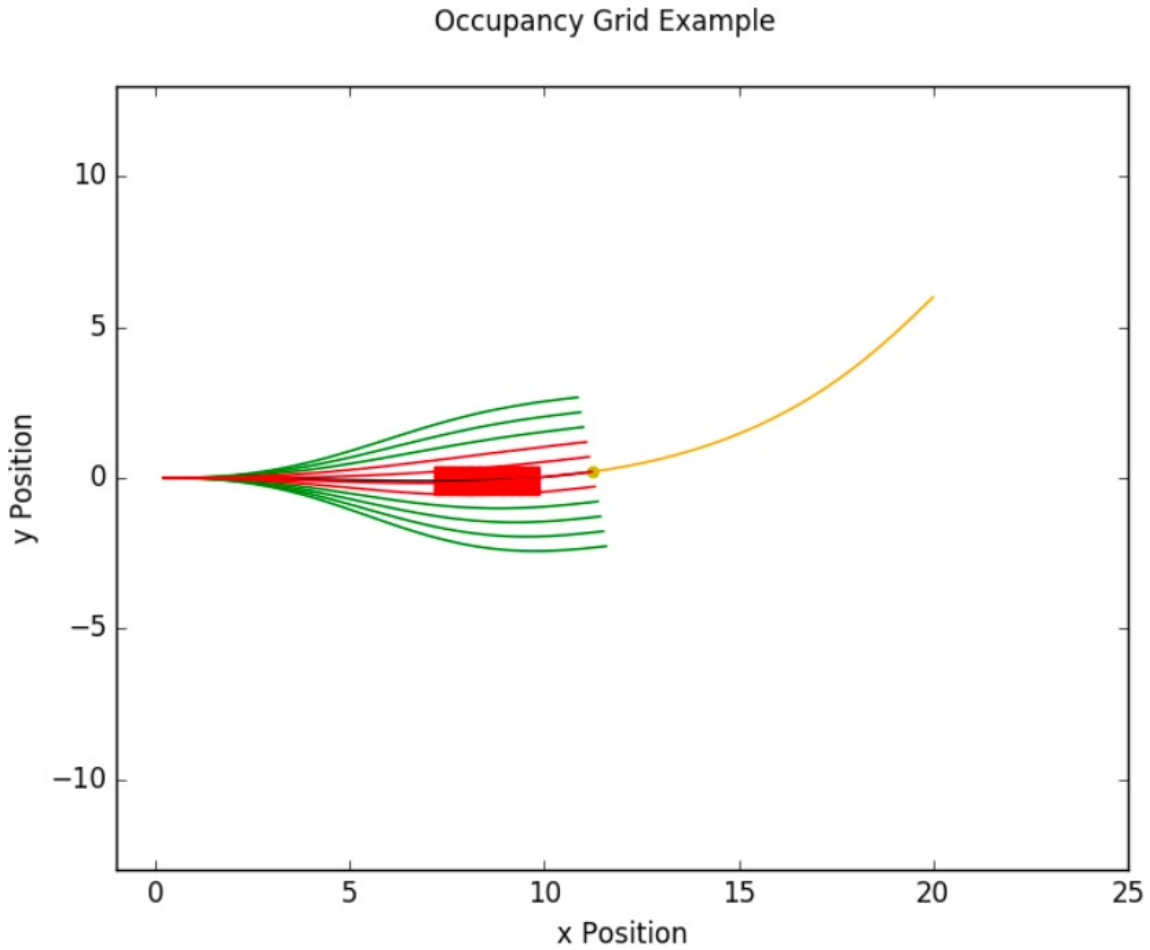
Conformal lattice planner

- 选择前方道路中心线上的目标点
- 重复N次：
  - 将目标点沿横向偏移固定的距离
  - 通过解边界值问题，生成当前状态到目标点的行车轨迹



Conformal lattice planner

- 选择前方道路中心线上的目标点
- 重复N次：
  - 将目标点沿横向偏移固定的距离
  - 通过解边界值问题，生成当前状态到目标点的行车轨迹
  - 对该轨迹做碰撞检测

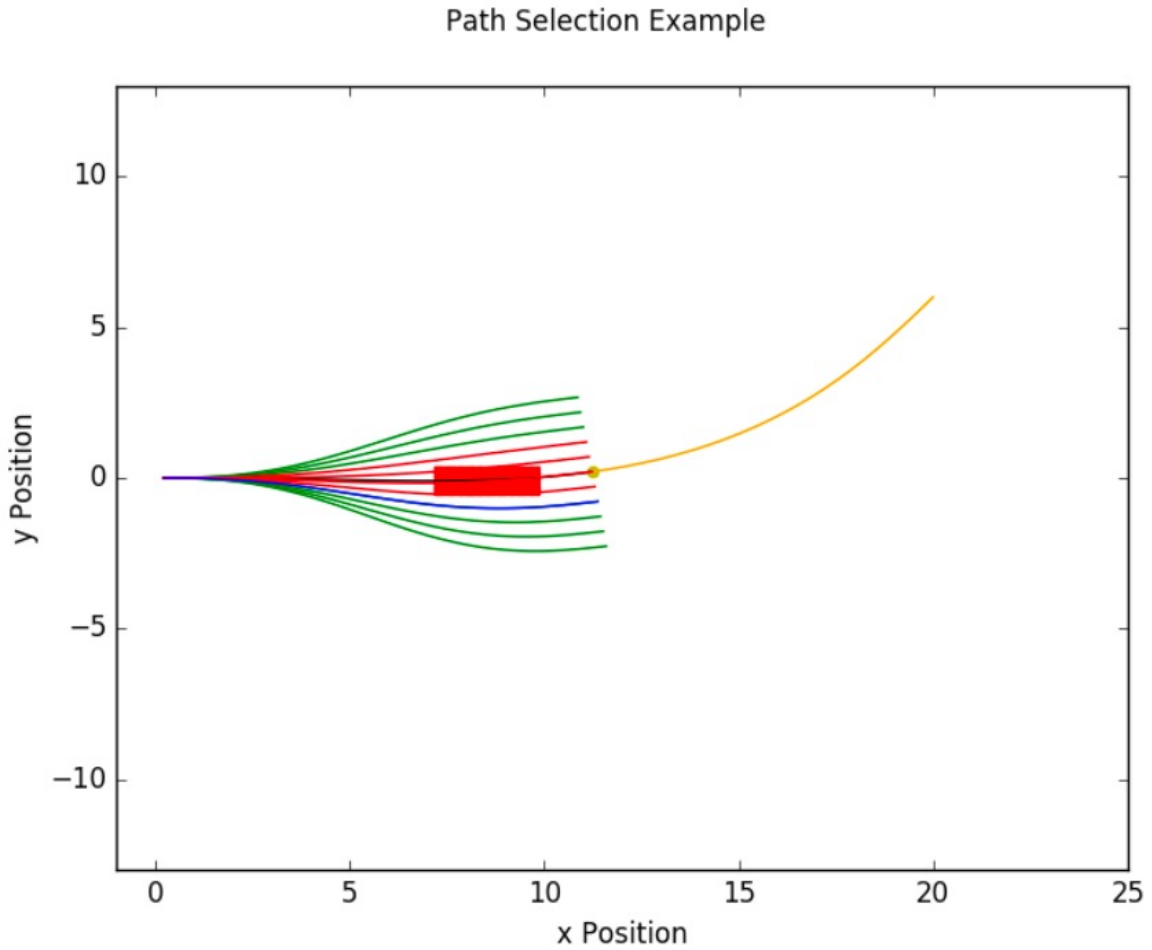


Conformal lattice planner

- 选择前方道路中心线上的目标点
- 重复N次：
  - 将目标点沿横向偏移
  - 通过解边界值问题，生成当前状态到目标点的行车轨迹
  - 对该轨迹做碰撞检测
- 选择最优的无碰撞路径

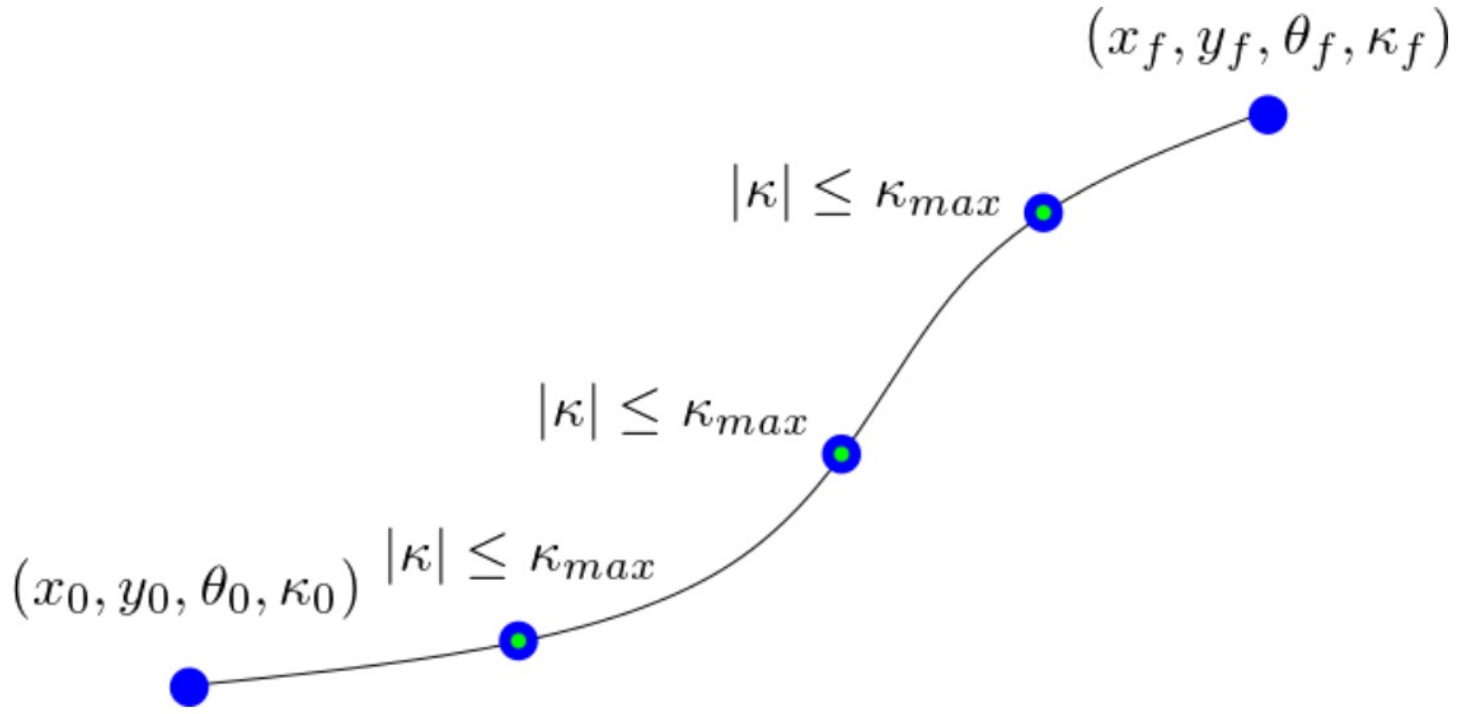
如何求解？

优化目标？



边界值问题求解

- 运动规划中的边界值问题：
  - **边界值**：路径起点、终点必须是指定状态
  - **其他限制**：
    - 路径必须平滑
    - 路径必须满足运动学限制



边界值问题的光滑解

- 参数化曲线（Parametric Curve）
  - 由参数等式表达的曲线
  - 参数通常是弧长（arc length）或归一化之后的弧长

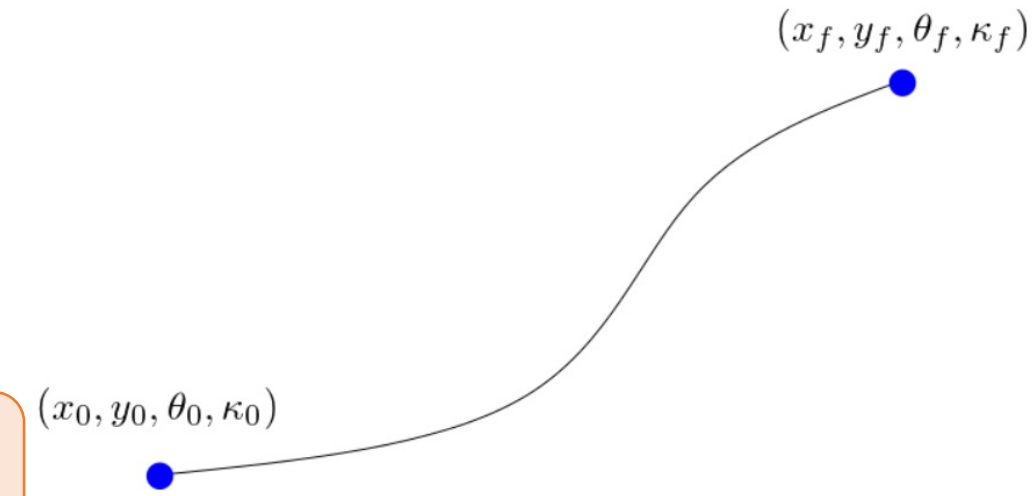
$$\mathbf{r}(u) = \langle x(u), y(u) \rangle$$
$$u \in [0,1]$$

- 例：三次样条（Cubic Splines）

$$x(u) = \alpha_3 u^3 + \alpha_2 u^2 + \alpha_1 u + \alpha_0$$
$$y(u) = \beta_3 u^3 + \beta_2 u^2 + \beta_1 u + \beta_0$$

唯一解

缺乏灵活参数去适应
额外的运动学限制



边界值问题的光滑解

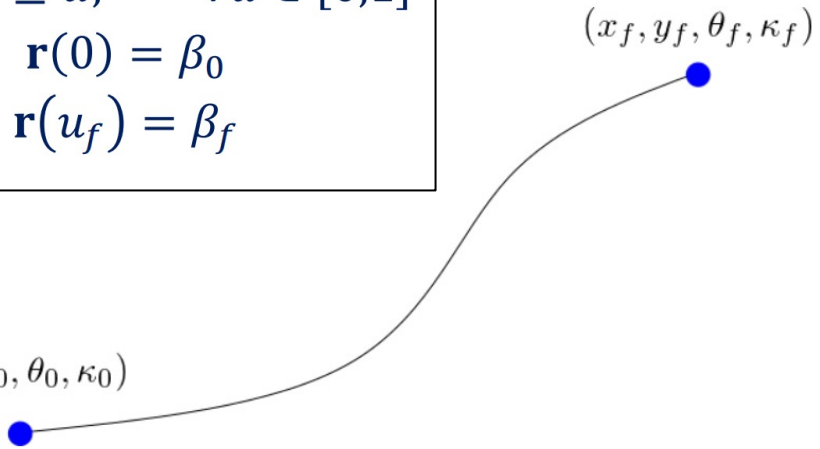
例：五次样条（Quintic Splines）

$$\min f(\mathbf{r}(u)) \text{ s.t. } \begin{cases} c(\mathbf{r}(u)) \le \alpha, & \forall u \in [0,1] \\ \mathbf{r}(0) = \beta_0 \\ \mathbf{r}(u_f) = \beta_f \end{cases}$$

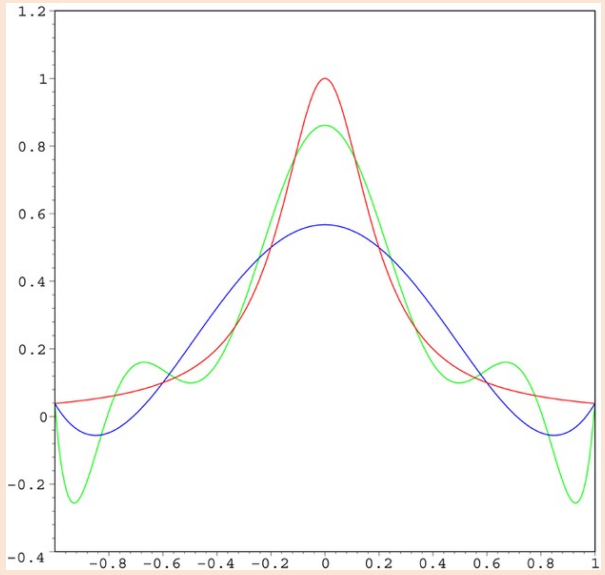
$$x(u) = \alpha_5 u^5 + \alpha_4 u^4 + \alpha_3 u^3 + \alpha_2 u^2 + \alpha_1 u + \alpha_0$$

$$y(u) = \beta_5 u^5 + \beta_4 u^4 + \beta_3 u^3 + \beta_2 u^2 + \beta_1 u + \beta_0$$

$$u \in [0,1]$$

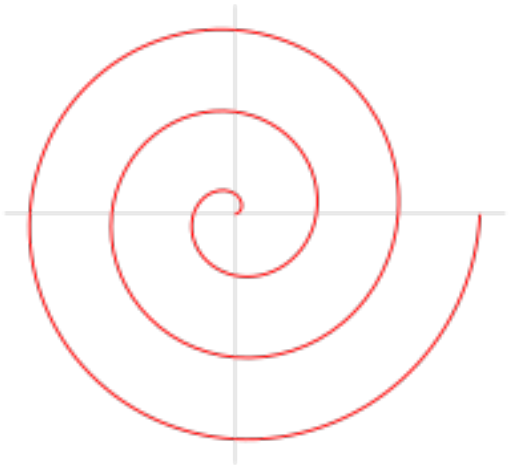


路径曲率难以控制



边界值问题的光滑解

- **例**：三阶螺旋线（Cubic Spirals）
  - **螺旋线**：将曲率表达为弧长的函数
  - **曲率（curvature）**：曲线上某个点的切线方向角度相对于弧长的转动率

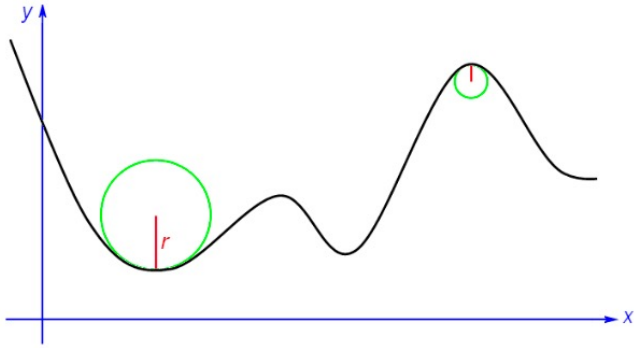


一阶螺旋线

$k(s) = as + b, \qquad a > 0$

曲率

$$\kappa = \frac{1}{r}$$



曲率

$$\kappa = \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{3/2}}$$

边界值问题求解

例：三阶螺旋线（Cubic Spirals）

- 将曲率（curvature）表达为弧长的三次多项式函数
- 可以利用积分计算 $\theta(s), x(s), y(s)$

$$\kappa(s) = a_3 s^3 + a_2 s^2 + a_1 s + a_0$$

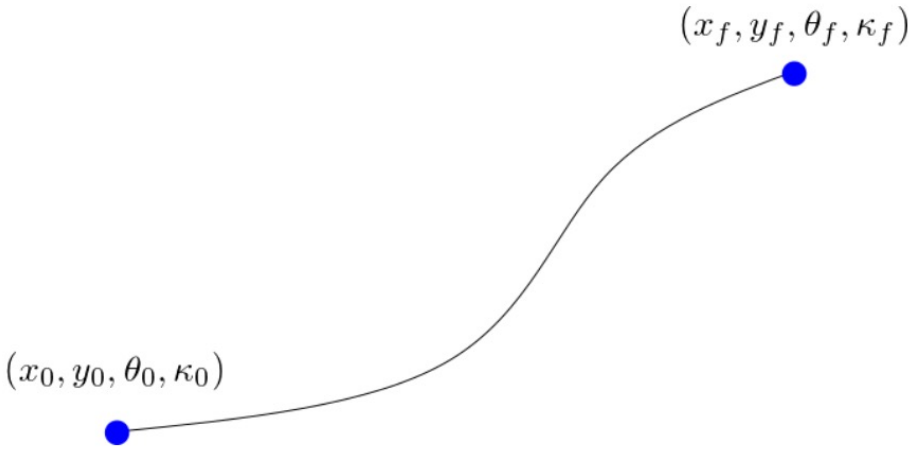
$$\theta(s) = \theta_0 + \int_0^s a_3 s'^3 + a_2 s'^2 + a_1 s' + a_0 ds'$$

$$= \theta_0 + a_3 \frac{s^4}{4} + a_2 \frac{s^3}{3} + a_1 \frac{s^2}{2} + a_0 s$$

$$x(s) = x_0 + \int_0^s \cos\big(\theta(s')\big) ds'$$

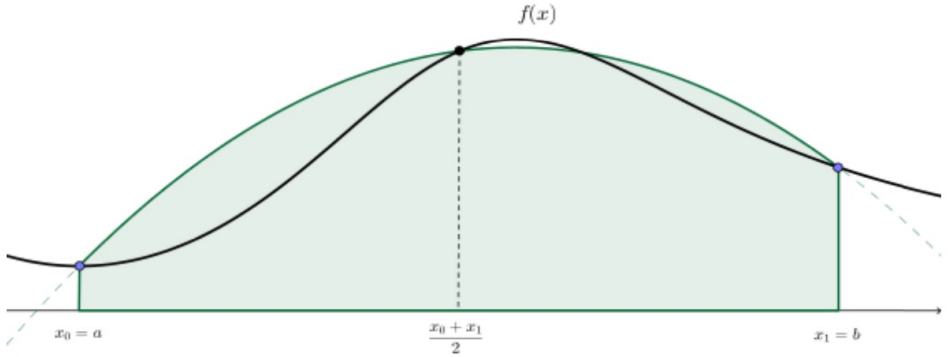
$$y(s) = y_0 + \int_0^s \sin\big(\theta(s')\big) ds'$$

没有解析解，需要数值积分方法



边界值问题求解

- **数值积分**：Simpson's Rule
  - 对函数 f(x) 进行二次多项式插值
  - 对二次插值曲线进行积分



$$\int_0^s f(s')ds' \approx \frac{s}{3n}\left(f(0) + 4f\left(\frac{s}{n}\right) + 2f\left(\frac{2s}{n}\right) + \cdots + f(s)\right)$$

Simpson's Rule

$$x(s) = x_0 + \int_0^s \cos\big(\theta(s')\big)ds'$$
$$y(s) = y_0 + \int_0^s \sin\big(\theta(s')\big)ds'$$

$$x_S(s) = x_0 + \frac{s}{24}\left[\cos(\theta(0)) + 4\cos\left(\theta\left(\frac{s}{8}\right)\right) + 2\cos\left(\theta\left(\frac{2s}{8}\right)\right) + 4\cos\left(\theta\left(\frac{3s}{8}\right)\right) + 2\cos\left(\theta\left(\frac{4s}{8}\right)\right) + 4\cos\left(\theta\left(\frac{5s}{8}\right)\right) + 2\cos\left(\theta\left(\frac{6s}{8}\right)\right) + 4\cos\left(\theta\left(\frac{7s}{8}\right)\right) + \cos(\theta(s))\right]$$

$$y_S(s) = y_0 + \frac{s}{24}\left[\sin(\theta(0)) + 4\sin\left(\theta\left(\frac{s}{8}\right)\right) + 2\sin\left(\theta\left(\frac{2s}{8}\right)\right) + 4\sin\left(\theta\left(\frac{3s}{8}\right)\right) + 2\sin\left(\theta\left(\frac{4s}{8}\right)\right) + 4\sin\left(\theta\left(\frac{5s}{8}\right)\right) + 2\sin\left(\theta\left(\frac{6s}{8}\right)\right) + 4\sin\left(\theta\left(\frac{7s}{8}\right)\right) + \sin(\theta(s))\right]$$

边界值问题求解

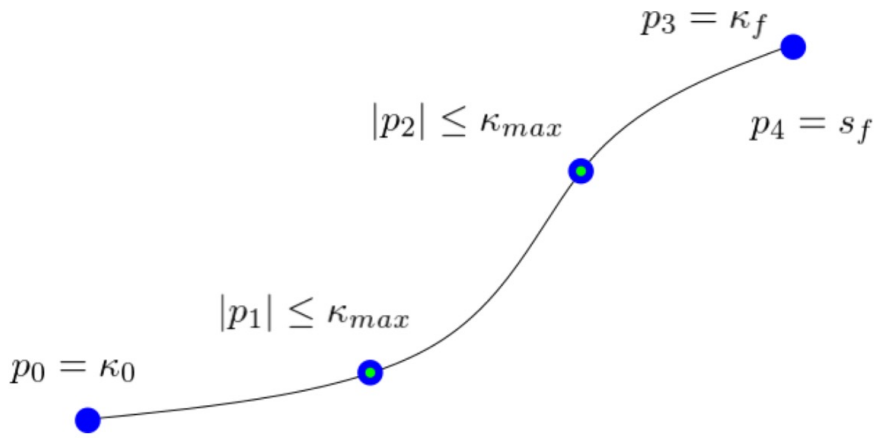
$$\kappa(s) = a_3 s^3 + a_2 s^2 + a_1 s + a_0$$

约束优化问题

- 近似曲率约束：
  - 只考虑4个均匀分布的点：$0, \frac{s_f}{3}, \frac{2s_f}{3}, s_f$
- 状态边界值限制：
  - 起始边界值已经在积分初始值中考虑
  - 终点边界值是约束条件
- 优化目标：最小化 Bending Energy

$$f_{be}(a_0, a_1, a_2, a_3, s_f)$$

$$= \int_0^{s_f} (a_3 s^3 + a_2 s^2 + a_1 s + a_0)^2 \, ds$$



$$\left|\kappa\left(\frac{s_f}{3}\right)\right| \leq \kappa_{max}, \qquad \left|\kappa\left(\frac{2s_f}{3}\right)\right| \leq \kappa_{max}$$

$$x_S(0) = x_0, \qquad x_s(s_f) = x_f$$

$$y_S(0) = y_0, \qquad y_S(s_f) = y_f$$

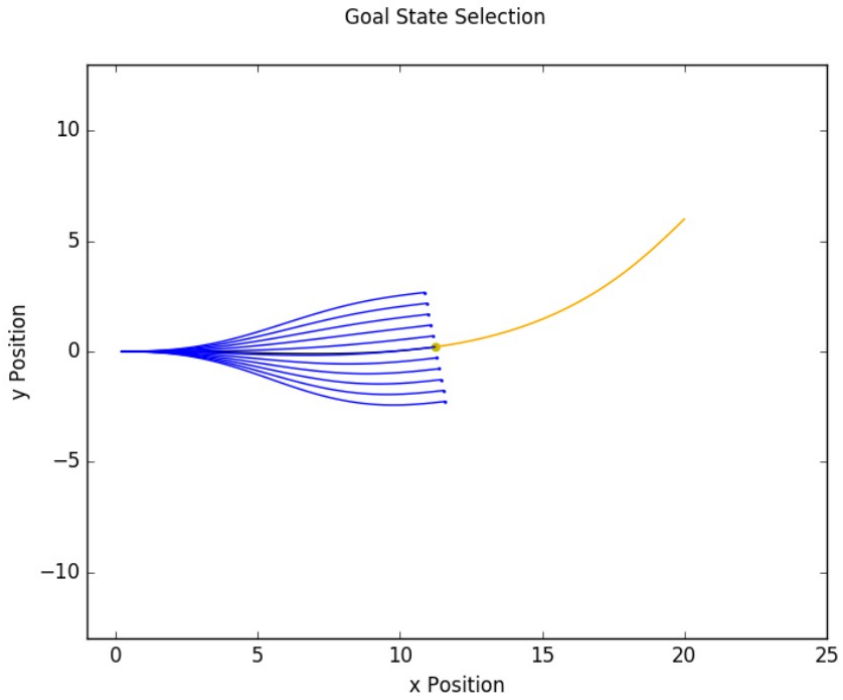
$$\theta(0) = \theta_0, \qquad \theta(s_f) = \theta_f$$

$$\kappa(0) = \kappa_0, \qquad \kappa(s_f) = \kappa_f$$

边界值问题求解

近似地求解约束优化问题

- **Lagrangian 方法**：将硬性约束（hard constraints）转换为软性约束 (soft constraints)，从而最小化约束违反度
- **Lagrangian multiplier (α, β, γ)** 是非常大的数值



$$\min f_{be}(a_0, a_1, a_2, a_3, s_f) + \alpha\left(x_S(s_f) - x_f\right) + \beta\left(y_S(s_f) - y_f\right) + \gamma\left(\theta_S(s_f) - \theta_f\right)$$

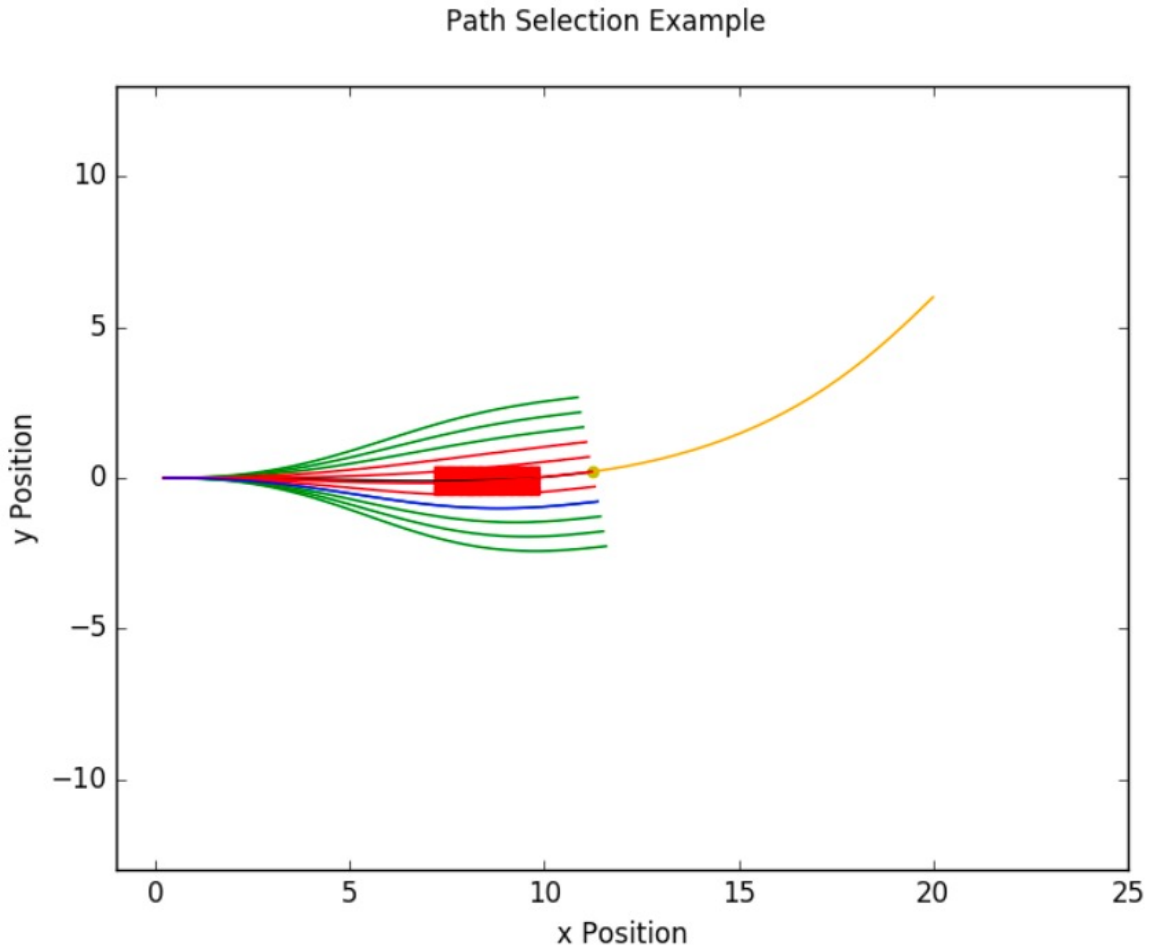
$$\text{s.t.} \begin{cases} \left|\kappa\left(\frac{s_f}{3}\right)\right| \le \kappa_{max} \\ \left|\kappa\left(\frac{2s_f}{3}\right)\right| \le \kappa_{max} \\ \kappa(s_f) = \kappa_f \end{cases}$$

不一定精确到达终点

Conformal lattice planner

- 选择前方道路中心线上的目标点
- 重复N次：
  - 将目标点沿横向偏移固定的距离
  - 通过解**边界值问题**，生成当前状态到目标点的行车轨迹
  - 对该轨迹做碰撞检测
- 选择最优的无碰撞路径

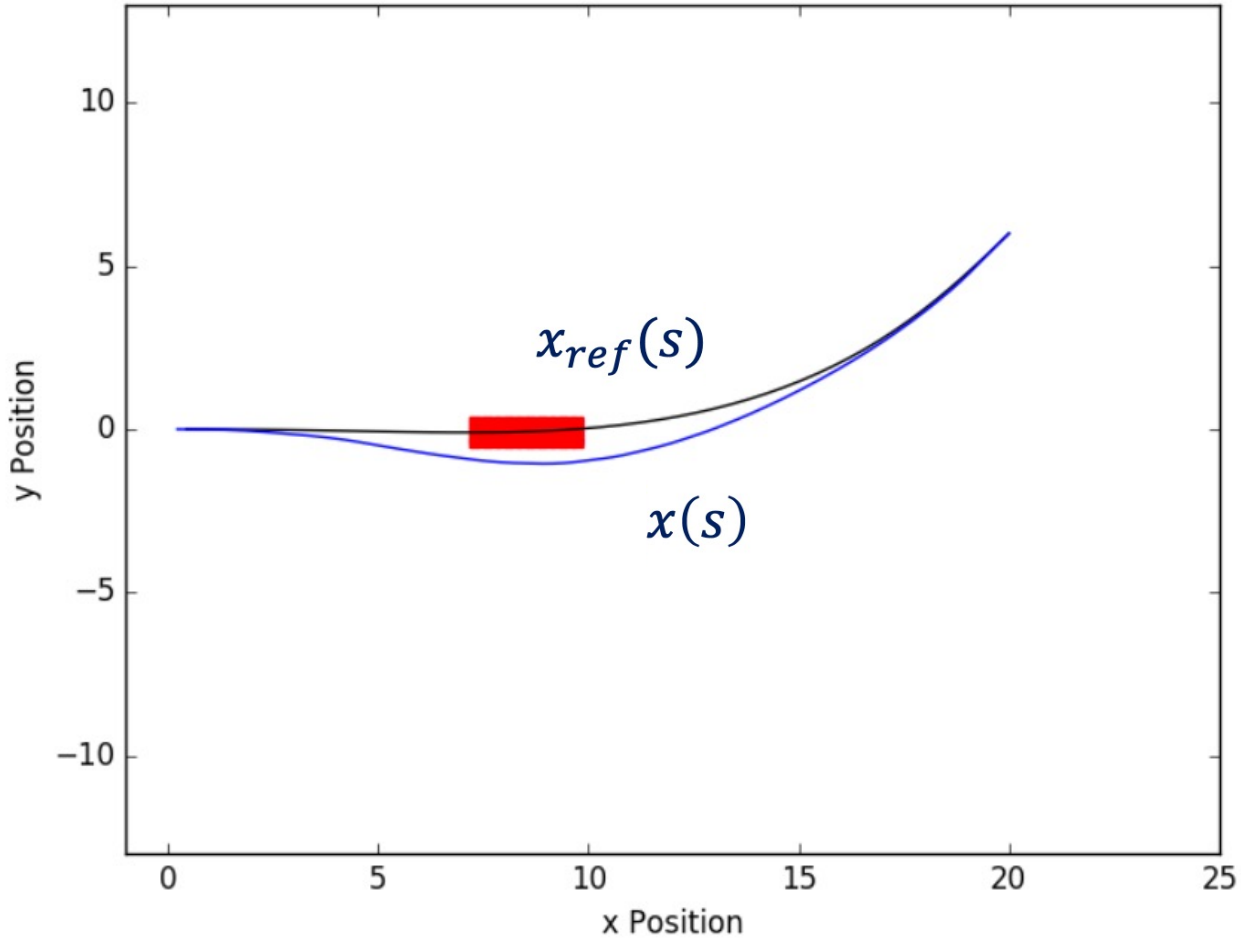
优化目标？



路径优化目标

- **累积路径追踪误差：**

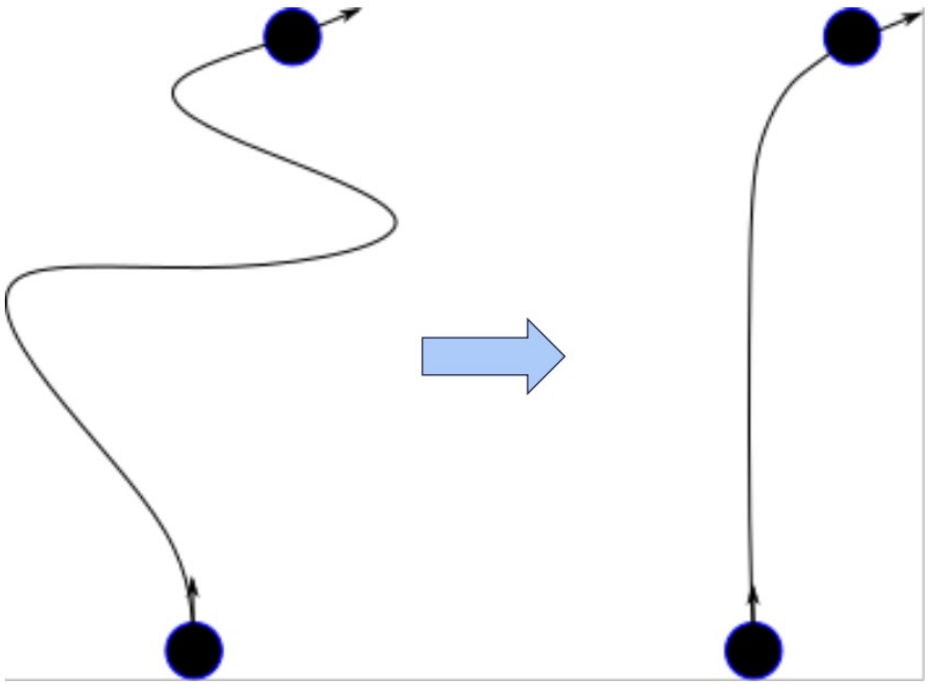
$$\int_0^{s_f} \|x(s) - x_{ref}(s)\| ds$$



路径规划的通用目标

- **路径长度**：$$s_f = \int_{x_i}^{x_f} \sqrt{1 + \left(\frac{dy}{dx}\right)^2}\, dx$$

- **运行时间**：$$T_f = \int_0^{s_f} \frac{1}{v(s)}\, ds$$

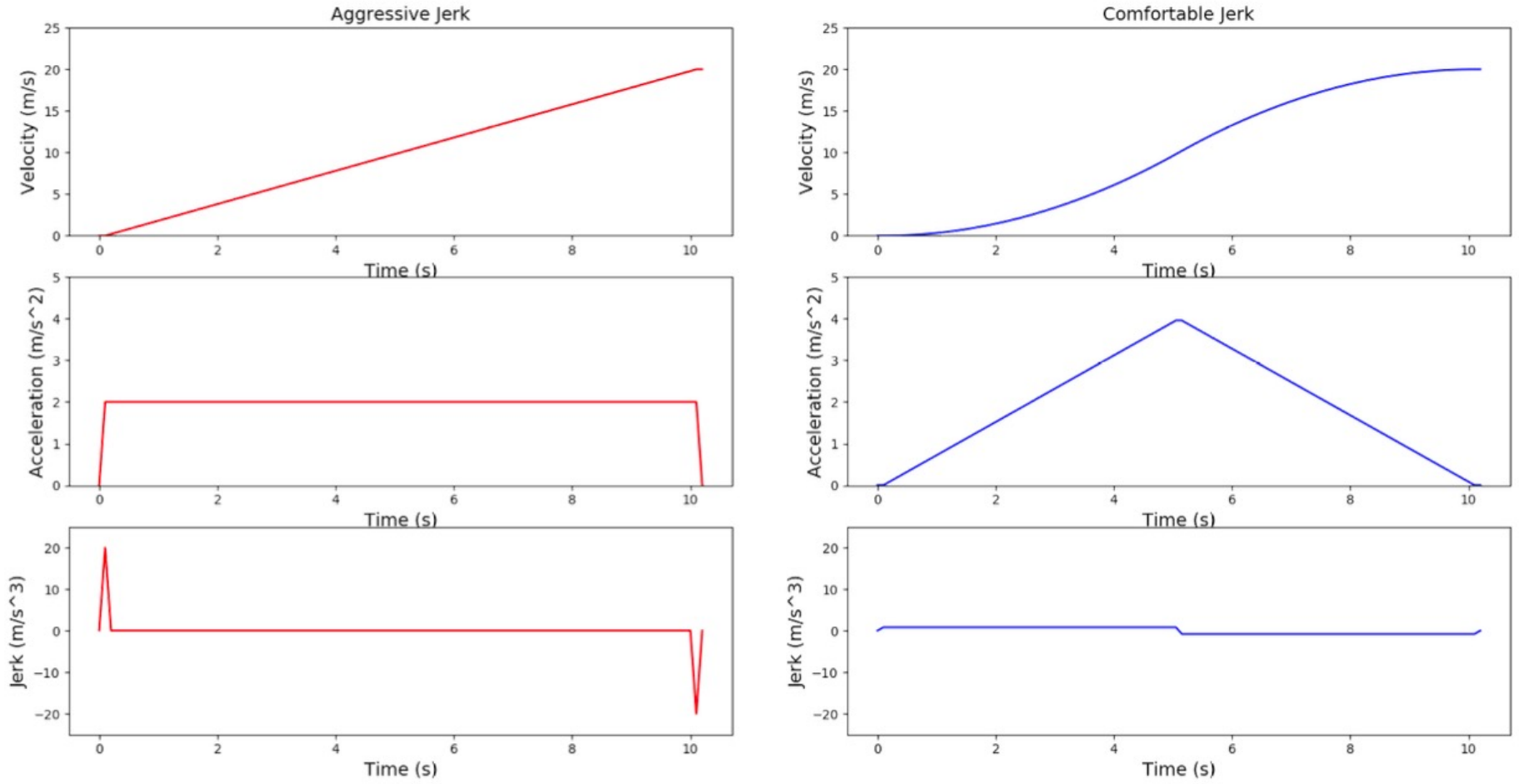


路径规划的通用目标

- **光滑度**：累积 jerk

$$\int_0^{s_f} \|\ddot{x}(s)\|^2 ds$$

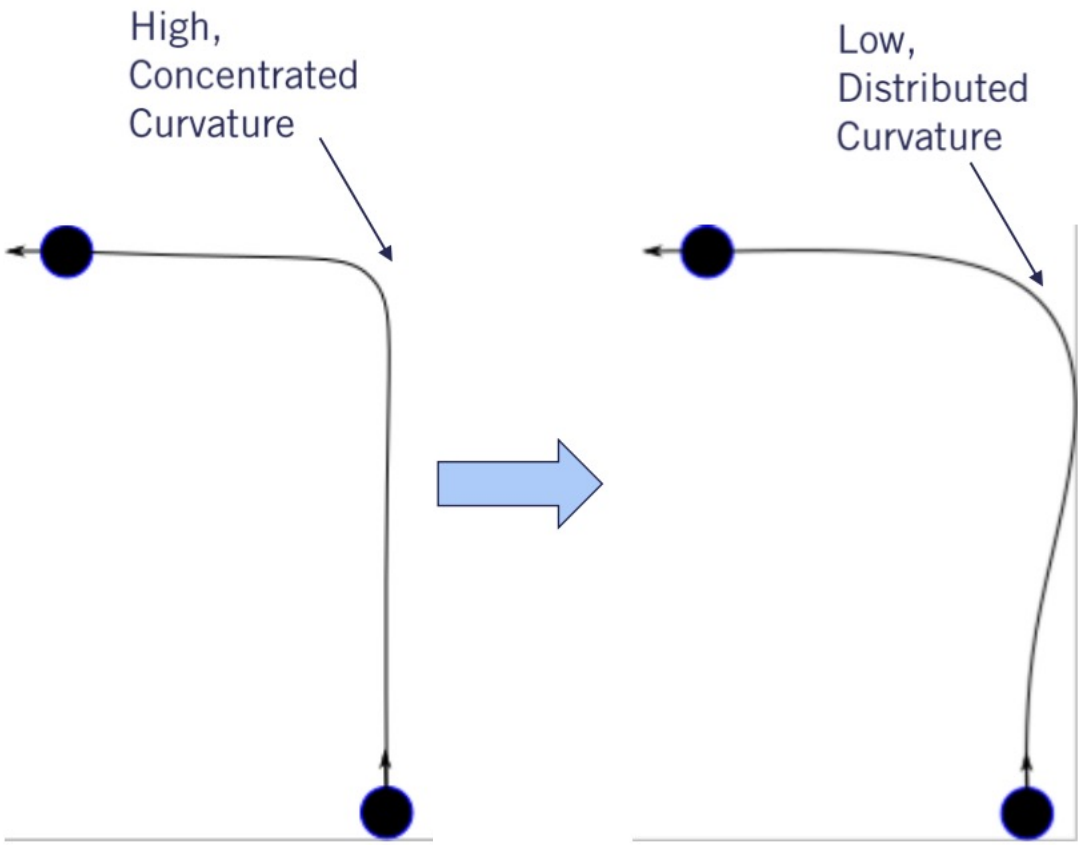
Jerk



路径规划的通用目标

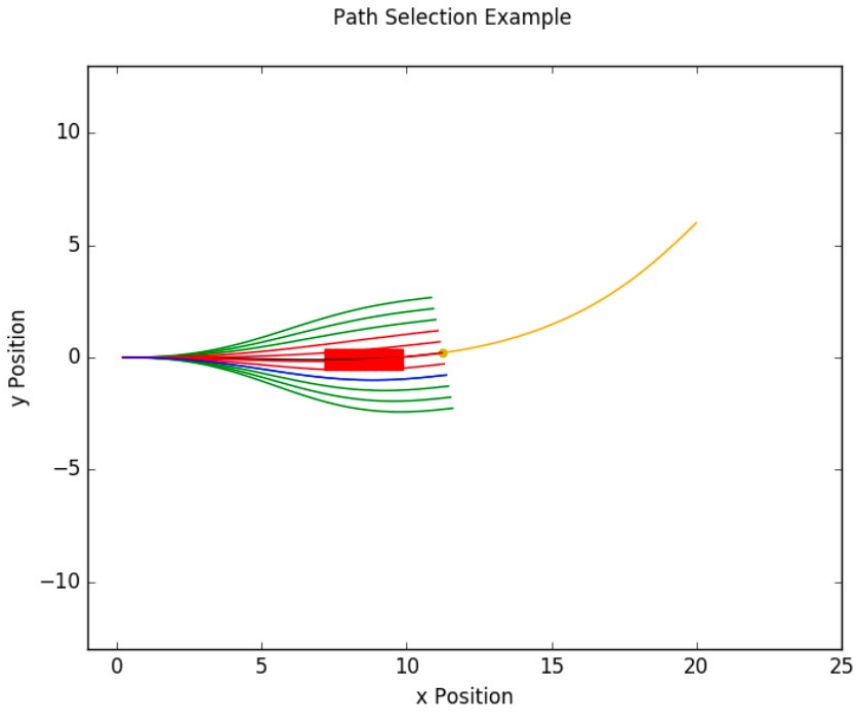
- **光滑度**：累积曲率

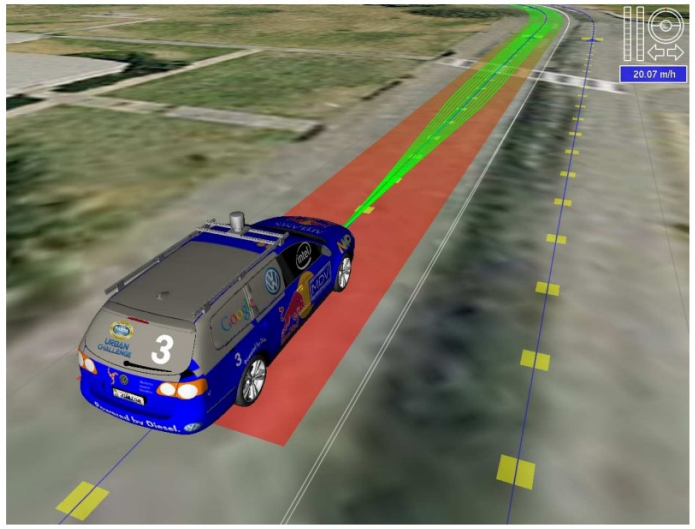
$$\int_0^{s_f} \|\kappa(s)\|^2 ds$$



完整的 Conformal lattice planner

- 选择前方道路中心线上的目标点
- 重复N次：
  - 将目标点沿横向偏移固定的距离
  - 通过解边界值问题，生成当前状态到目标点的行车轨迹
  - 对该轨迹做碰撞检测
- 选择最优的无碰撞路径：
  - $\min \boldsymbol{\gamma_T C_T} + \gamma_L C_L + \gamma_S C_S$
  - $\boldsymbol{C_T}$：累积路径追踪误差（必须）
  - C_L：路径长度（可选）
  - C_S：累积路径曲率（可选）





自动驾驶路径规划

任务规划（Mission Planning）

↓

行为规划（Behavior Planning）

↓

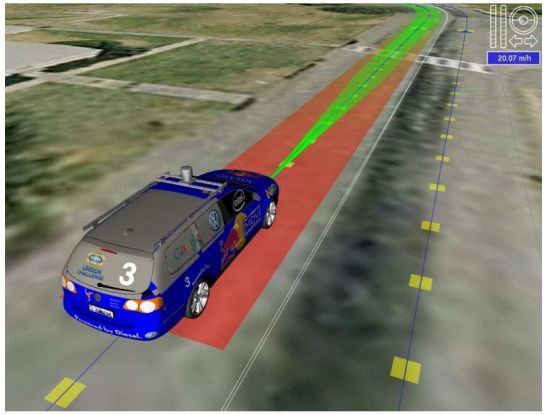
运动规划（Motion Planning）

路径规划（Path Planning） | 速度分布（Velocity Profile）

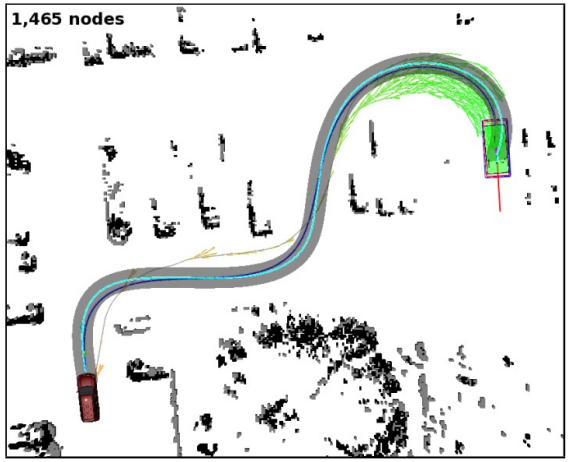
↓

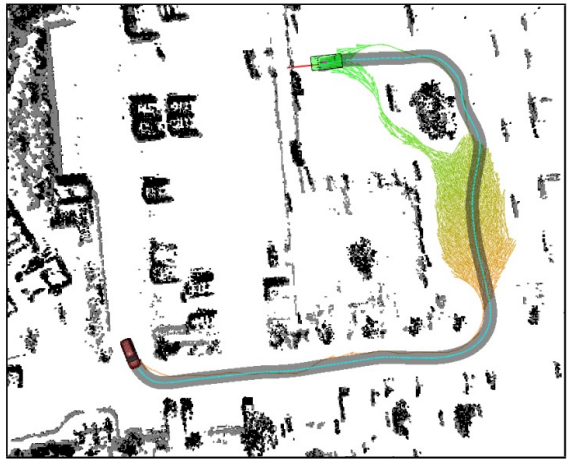
运动控制（Motion Control）

结构化场景 -> Conformal Lattice Planner

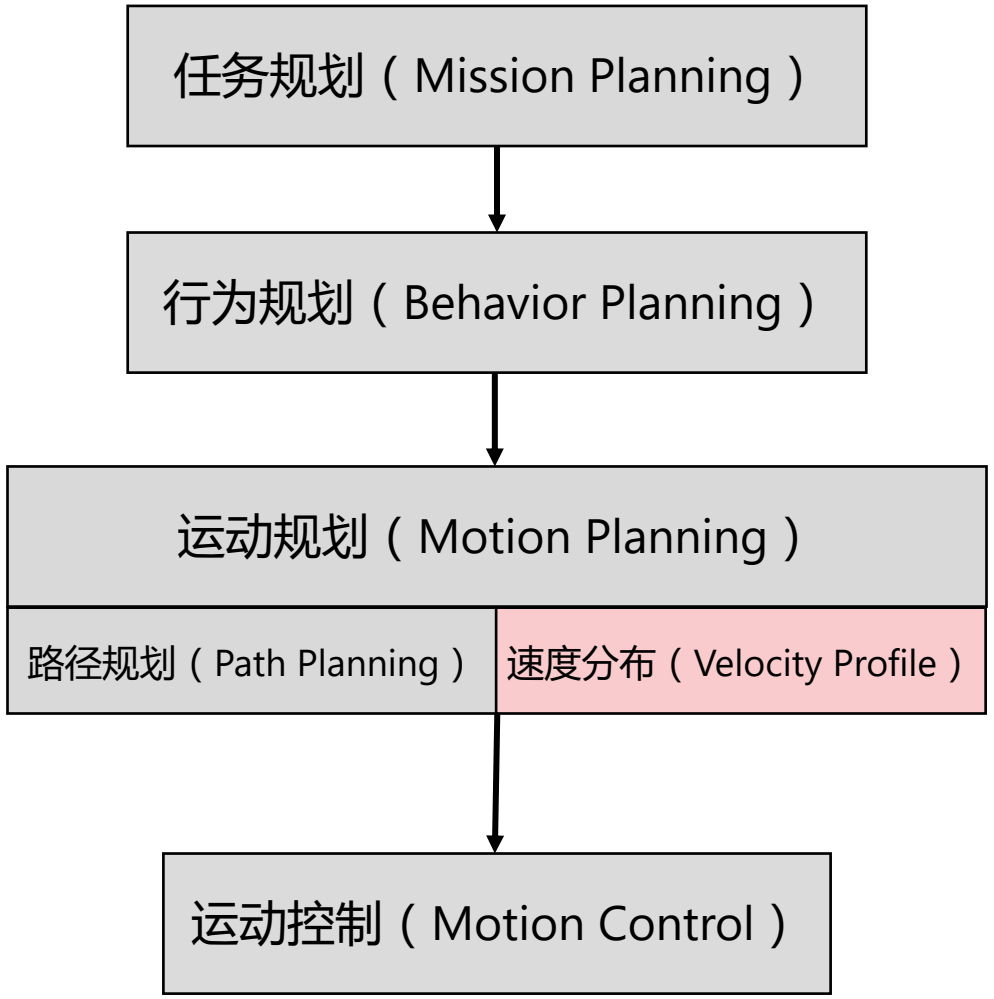


非结构化场景 -> Hybrid A* + 复合 heuristics





自动驾驶速度分布生成


任务规划（Mission Planning）
行为规划（Behavior Planning）
运动规划（Motion Planning）
路径规划（Path Planning）
速度分布（Velocity Profile）
运动控制（Motion Control）

自动驾驶速度分布生成

任务规划（Mission Planning）

行为规划（Behavior Planning）

运动规划（Motion Planning）

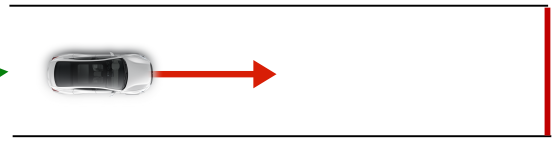
路径规划（Path Planning） | 速度分布（Velocity Profile）

运动控制（Motion Control）

结构化场景

非结构化场景

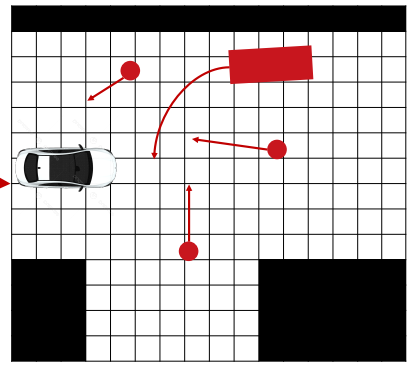
停车



启动

跟车、变道时与前车交互

其他复杂交互



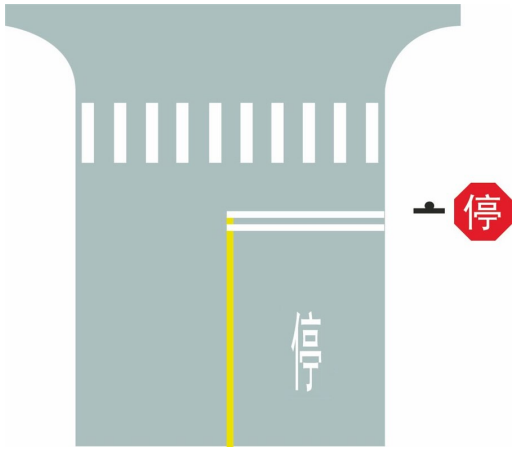
结构化场景中的目标速度 – 影响因素 I

- 参考速度 v_{ref}
  - 由用户指定：*“请开到 80 km/h”*
  - 受到道路限速的限制
  - 收到交通管制装置的约束

$v_{ref} \leq 60\ km/h$

$v_{ref} = 0\ km/h$

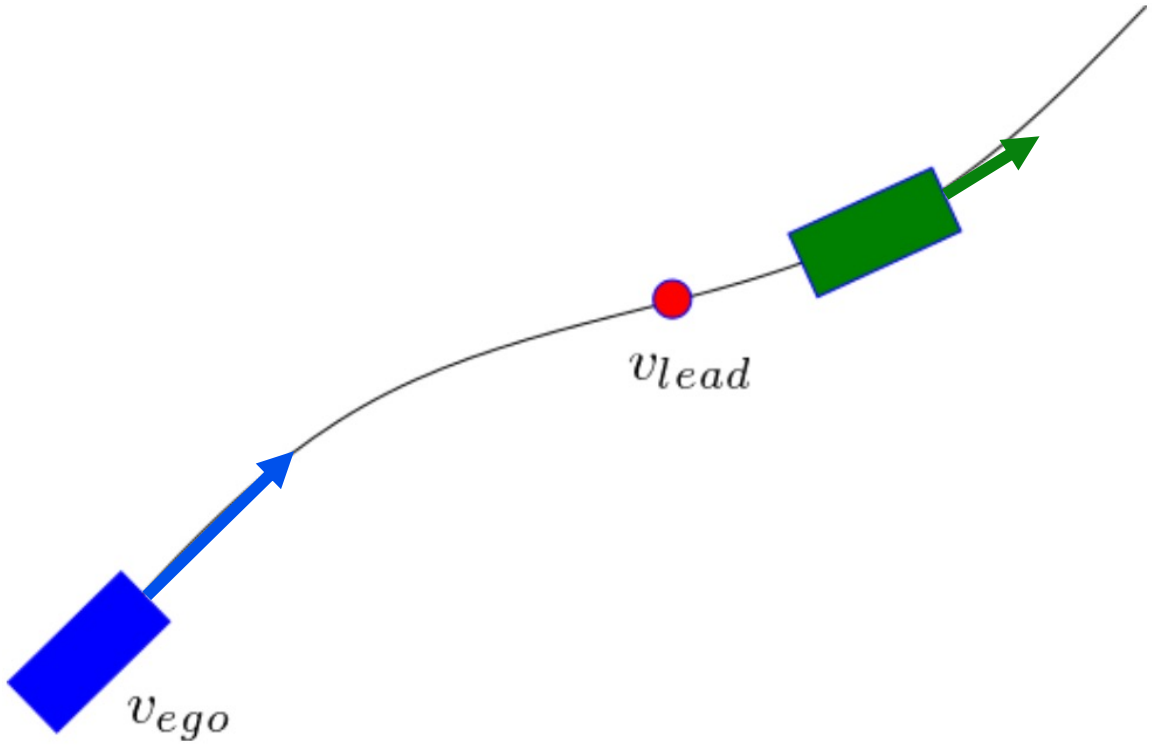
$v_{ref} = 0$（持续2s）



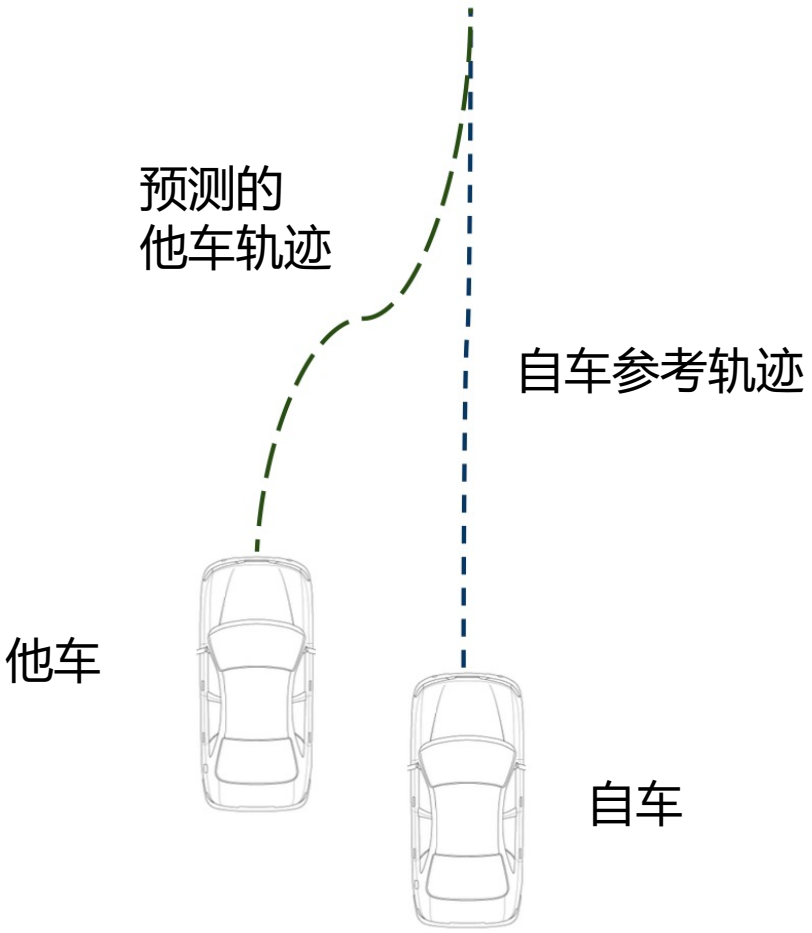
结构化场景中的目标速度 – 影响因素 II

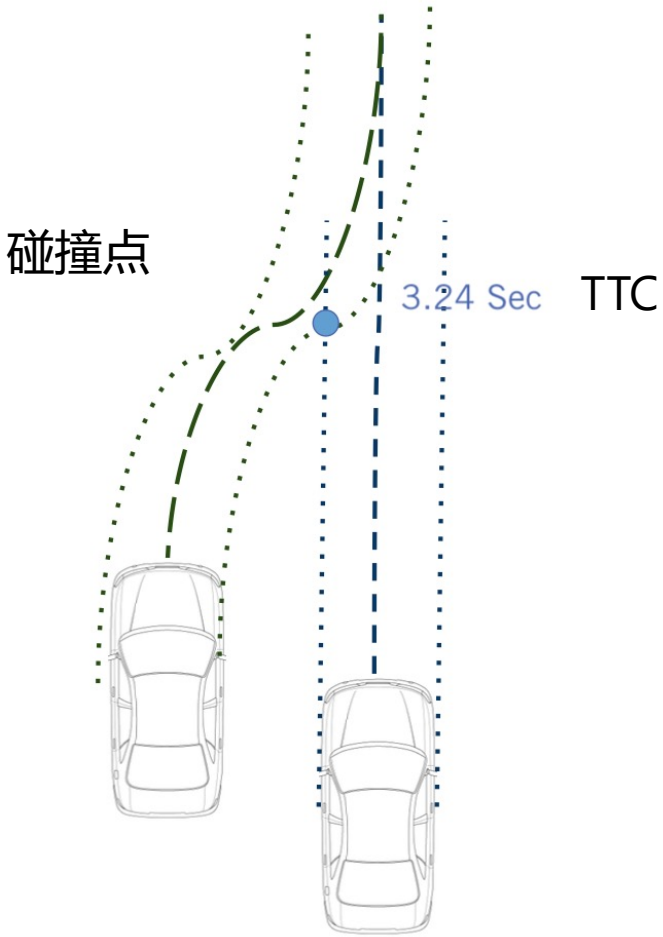
- 前车速度 v_{lead}
  - 在自车到达前车尾部之前，自车速度必须降到前车速度之下

- 红点 = 碰撞位置（ collision point ）



碰撞点（Collision Point）与碰撞时间（Time-To-Collision）

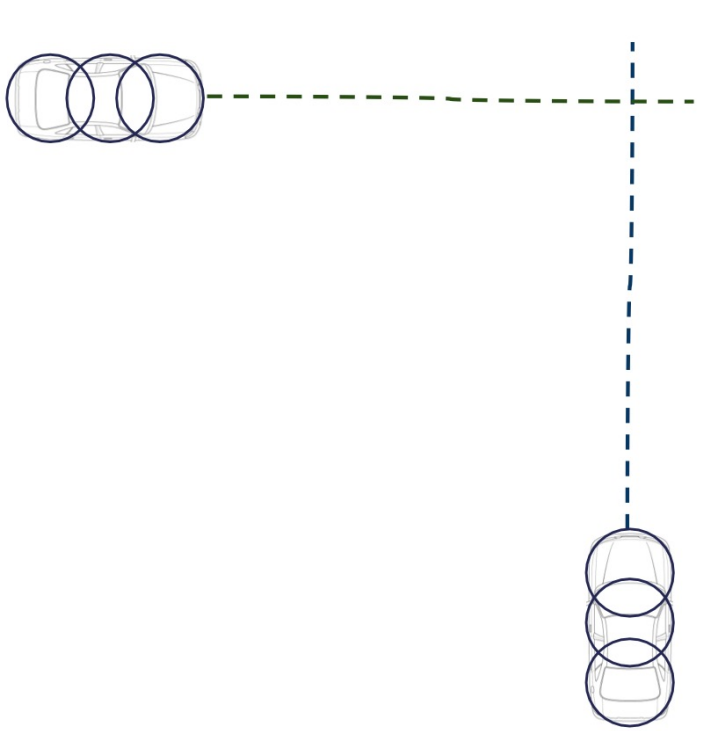


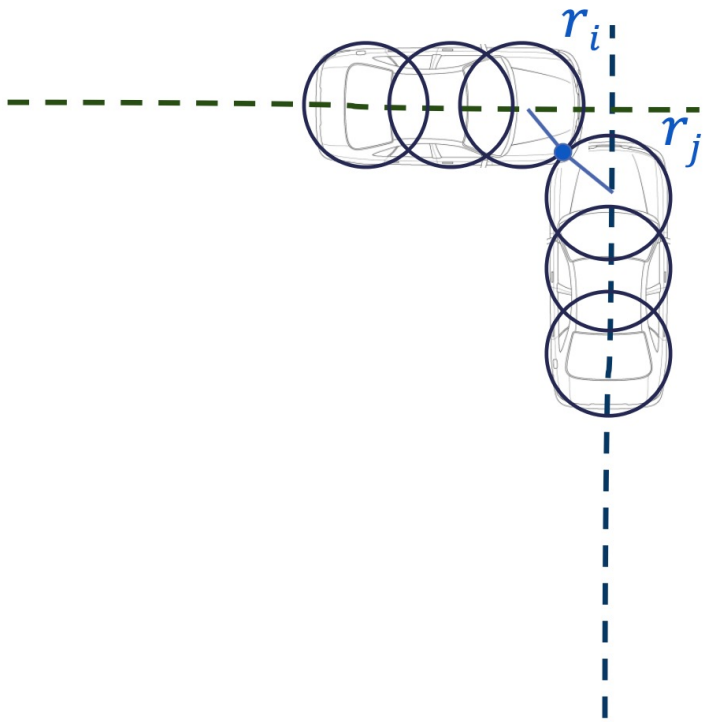


碰撞点（Collision Point）与碰撞时间（Time-To-Collision）

- 将车辆模拟成3个圆盘，按照参考与预测轨迹向前模拟车的未来状态
- 在每一个时间步，进行碰撞检测
- 当碰撞发生时，记录碰撞时间、碰撞点

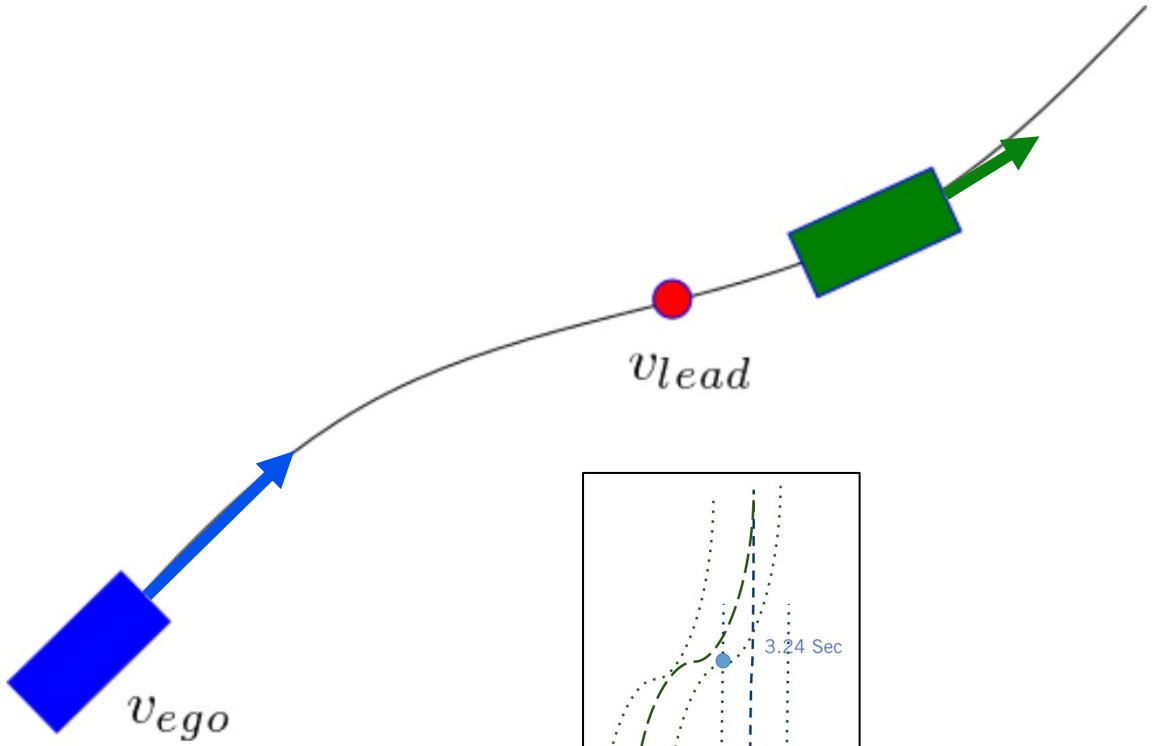
$$C_X = \frac{(x_i * r_j) + (x_j * r_i)}{(r_i + r_j)}, \qquad C_Y = \frac{(y_i * r_j) + (y_j * r_i)}{(r_i + r_j)}$$





结构化场景中的目标速度

- 前车速度 v_{lead}
  - 在到达路径上的**碰撞点（红点）**之前，自车速度必须降到前车速度之下

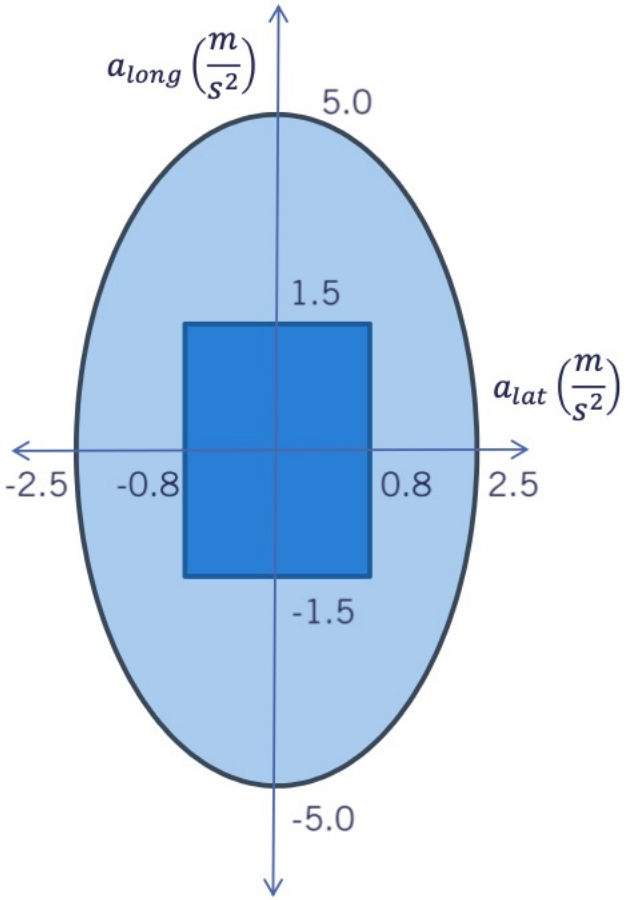


结构化场景中的目标速度 – 影响因素 III

- 运动学与动力学限制的最大速度 v_k
  - **摩擦力椭圆**：车辆加速度在椭圆内不会导致打滑
  - **舒适矩形**：车辆加速度在矩形内，乘客感到舒适
  - **横向加速度限制 -> 纵向速度限制：**
    - 车辆横向加速度与速度、转弯半径的关系：

$$a_{lat} = \frac{v^2}{r}, \qquad a_{lat} \leq a_{lat_{max}}$$

    - 根据曲率定义： $\kappa = \frac{1}{r}$
    - 于是： $v^2 \leq \frac{a_{lat_{max}}}{\kappa}$



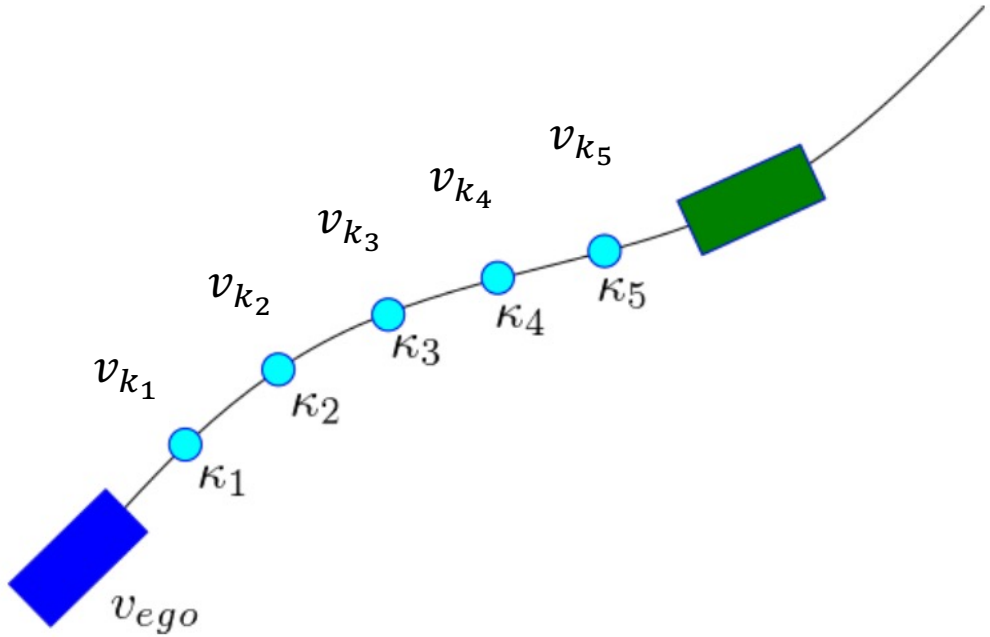
结构化场景中的目标速度分布

- 运动学与动力学限制的最大速度 v_k
  - 在**轨迹上任意一个点**，若当时的曲率是 κ_i，那么：

$$v_k \leq \sqrt{\frac{a_{lat}}{\kappa_i}}$$

  - 车辆在运动到第 i 个离散点时，必须满足：

$$v \leq v_{k_i}$$



结构化场景中的目标速度分布

- 结合以上三项因素，可以算出终点速度 v_f ：

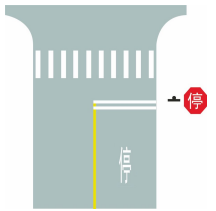
$$v_f = \min(v_{ref}, v_{lead}, v_k)$$

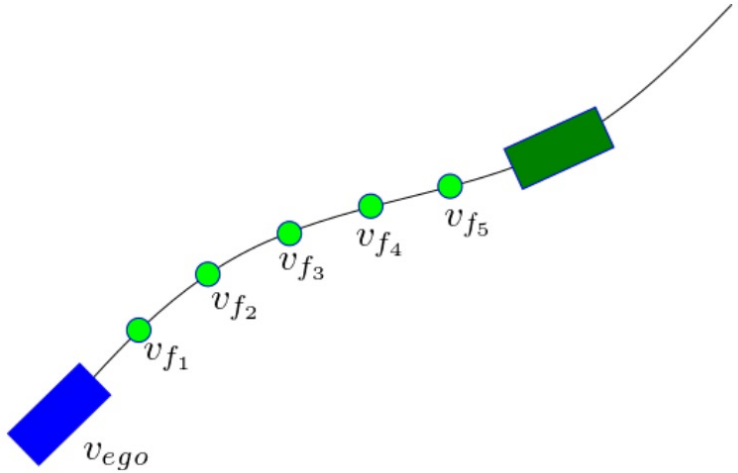
min

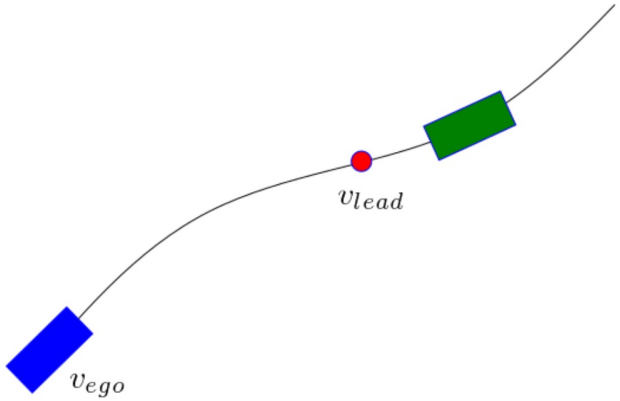
v_{ref}

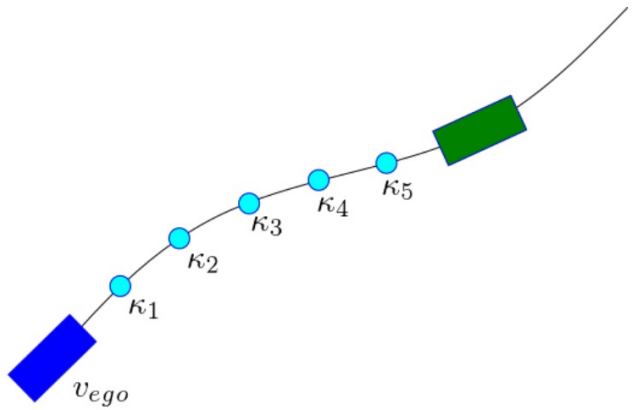
v_{lead}

v_k









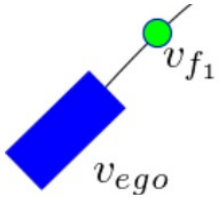
结构化场景中的速度曲线生成

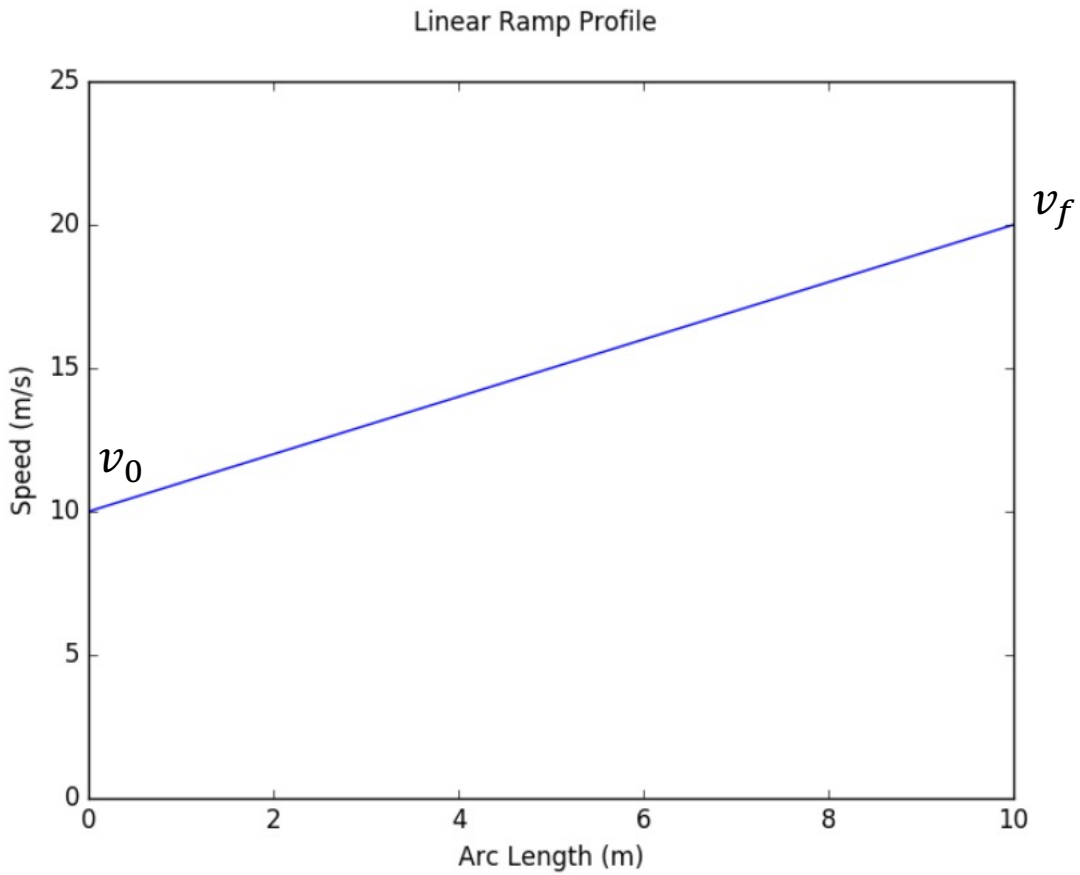
- 在当前路径段内，将速度从 v_0 变化到 v_f
- **方法1**：线性斜坡曲线（Linear Ramp Profile）
  - 将 v_0 与 v_f 沿着弧长维度线性插值
  - 需要的加速度可以计算为：

$$\frac{v_f^2 - v_0^2}{2s} = a$$

  - 若达到最大加速度限制，则不一定能达到 v_f：

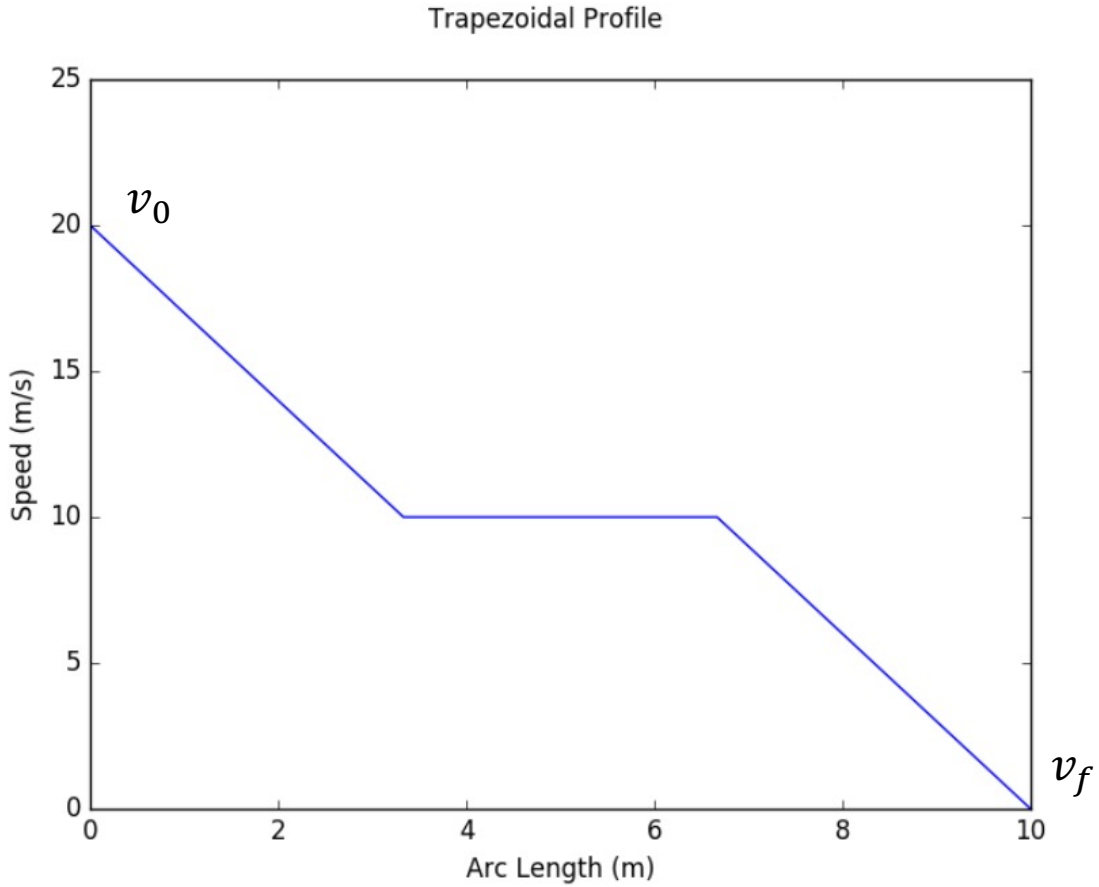
$$\sqrt{2as + v_0^2} = v_f$$





结构化场景中的速度曲线生成

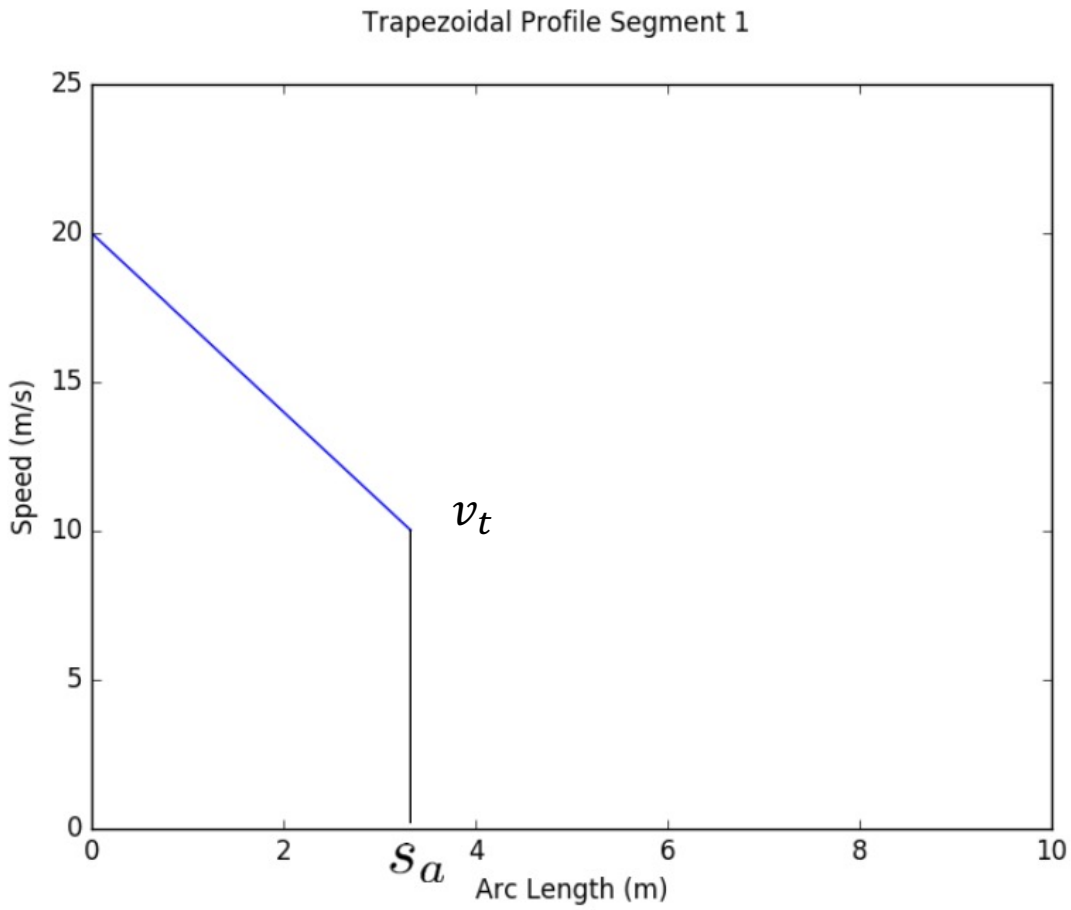
- 在当前路径段内，将速度从 v_0 变化到 v_f
- **方法2**：梯形曲线（Trapezoidal Profile）
  - 匀加/减速-匀速-匀加/减速
  - 停车时适用



结构化场景中的速度曲线生成

- 在当前路径段内，将速度从 v_0 变化到 v_f
- **方法2**：梯形曲线（Trapezoidal Profile）
  - **第一段**：执行匀加速度 s_0 达到临时速度 v_t

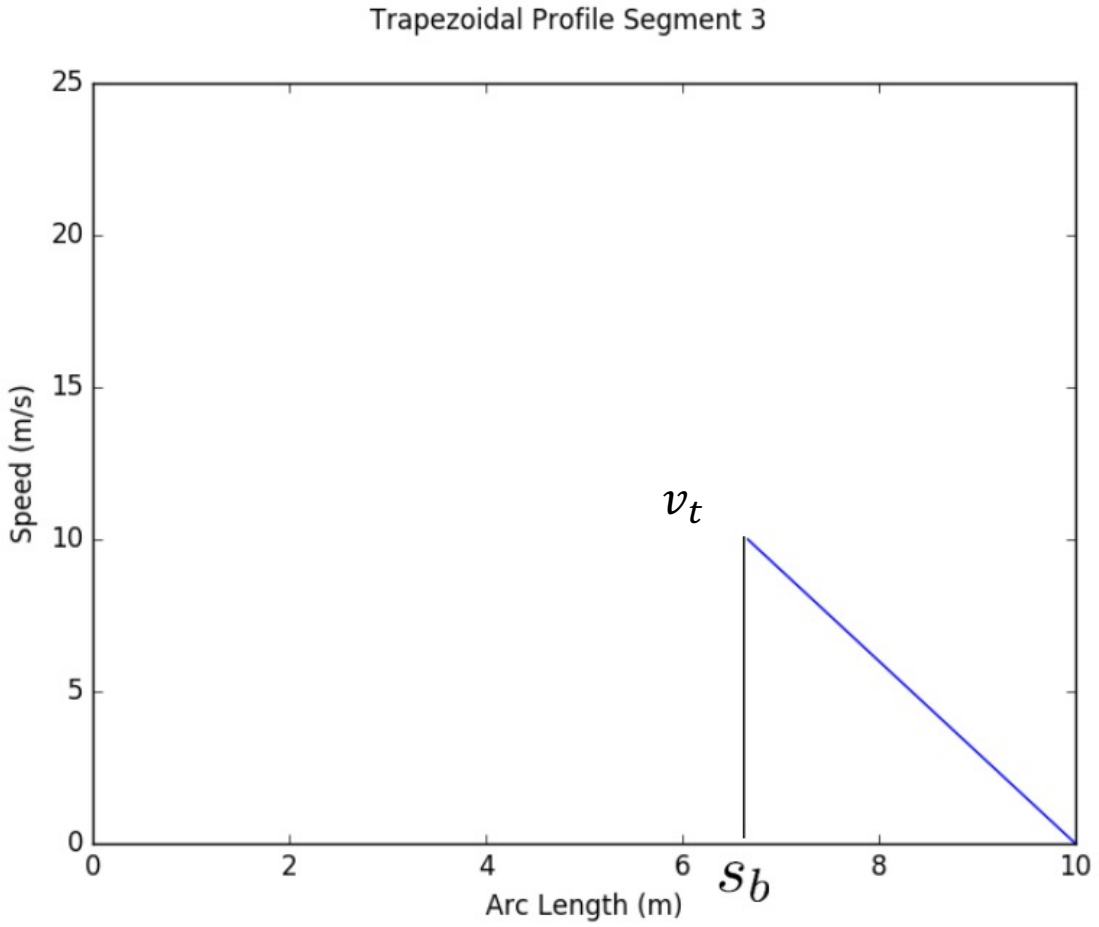
$$\frac{v_t^2 - v_0^2}{2a_0} = s_a$$



结构化场景中的速度曲线生成

- 在当前路径段内，将速度从 v_0 变化到 v_f
- **方法2**：梯形曲线（Trapezoidal Profile）
  - **第一段**：执行匀加速度 s_0 达到临时速度 v_t
  - **第三段**：执行匀加速度 s_0 达到最终速度 0

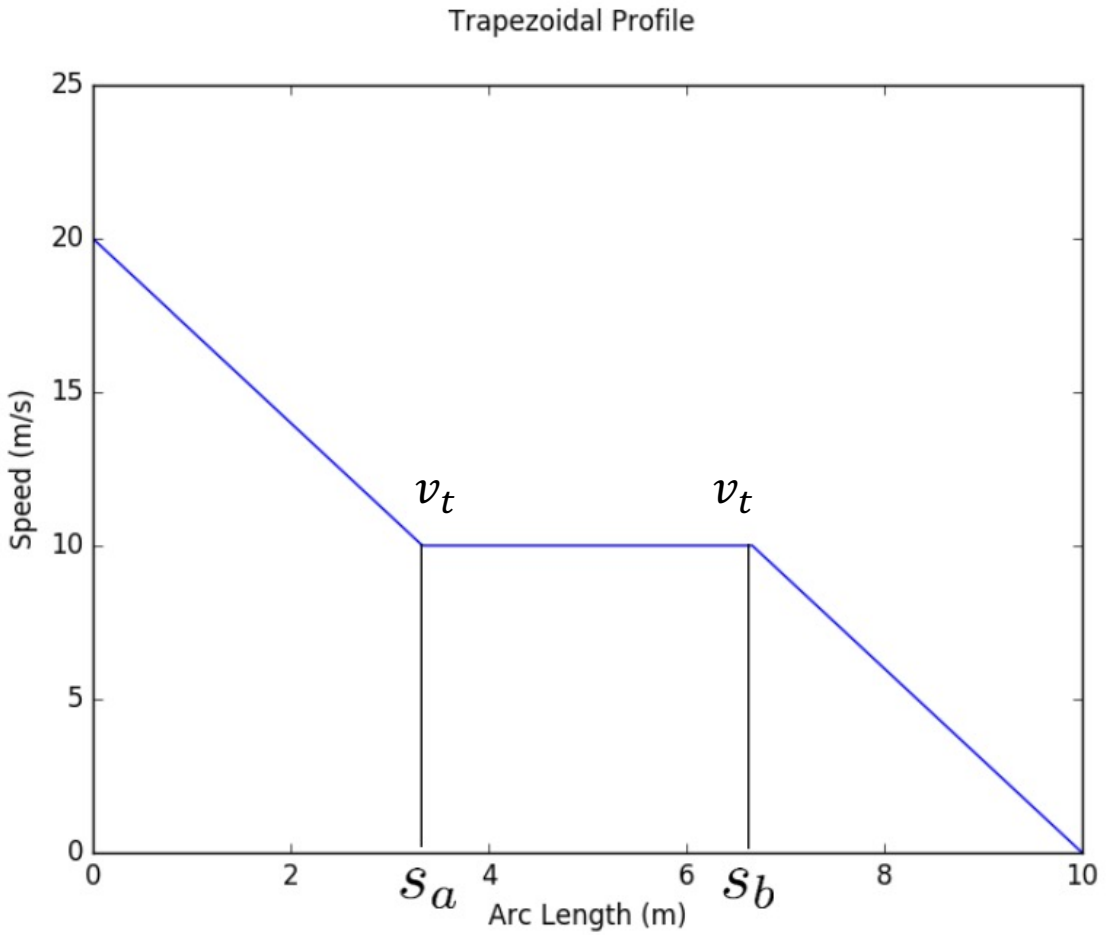
$$\frac{0-v_t^2}{2a_0} = s_f - s_b$$



结构化场景中的速度曲线生成

- 在当前路径段内，将速度从 v_0 变化到 v_f
- **方法2**：梯形曲线（Trapezoidal Profile）
  - **第一段**：执行匀加速度 s_0 达到临时速度 v_t
  - **第二段**：执行匀速运动
  - **第三段**：执行匀加速度 s_0 达到最终速度 0

$$v_{f_i} = \begin{cases} \sqrt{2a_0 s_i + v_i^2}, & s_i \le s_a \\ v_t, & s_a \le s_i \le s_b \\ \sqrt{2a_0(s_i - s_b) + v_i^2}, & s_b \le s_i \le s_f \end{cases}$$



$$\frac{v_t^2 - v_0^2}{2a_0} = s_a$$

$$\frac{0 - v_t^2}{2a_0} = s_f - s_b$$

自动驾驶速度分布生成

任务规划（Mission Planning）

行为规划（Behavior Planning）

运动规划（Motion Planning）

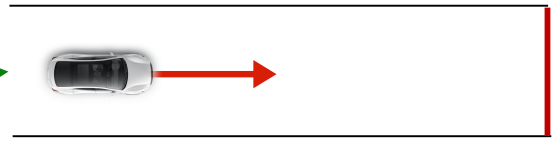
路径规划（Path Planning） | 速度分布（Velocity Profile）

运动控制（Motion Control）

结构化场景

非结构化场景

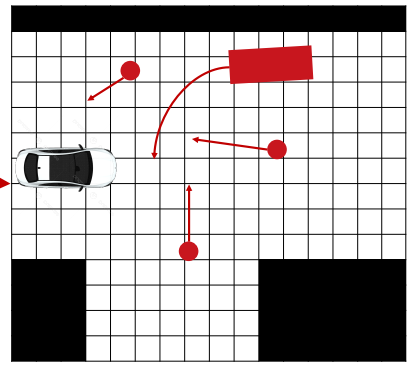
停车



启动

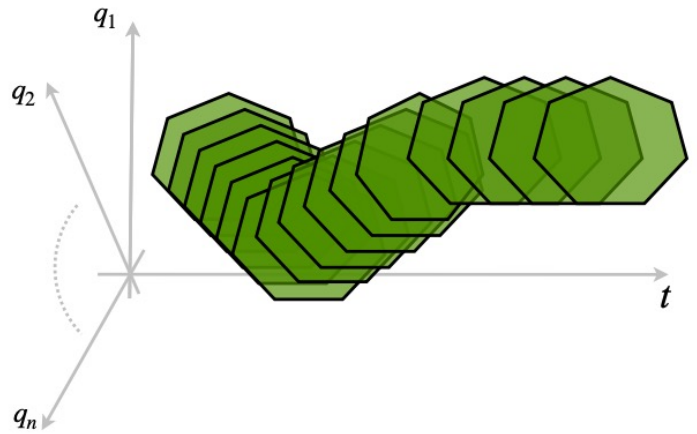
跟车、变道时与前车交互

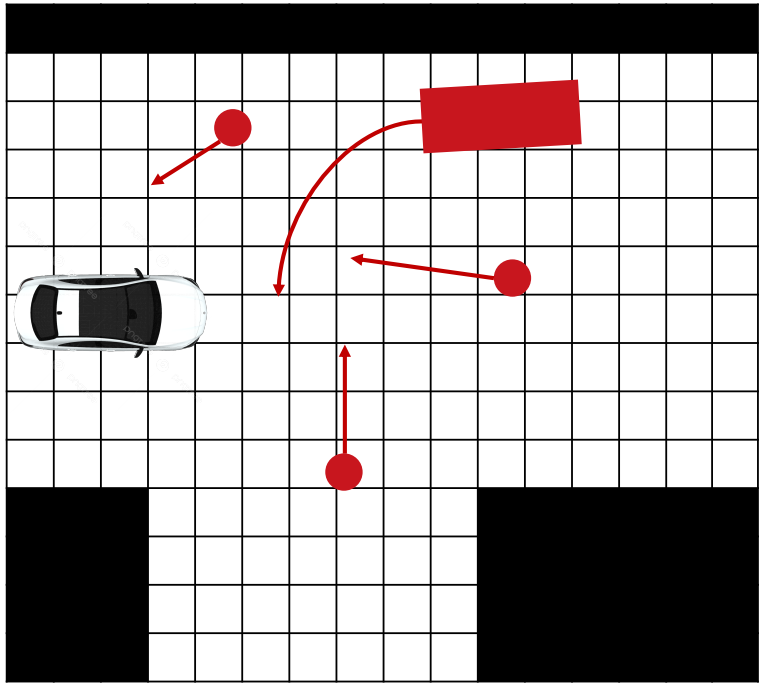
其他复杂交互



Recap：Hybrid A* 时空搜索

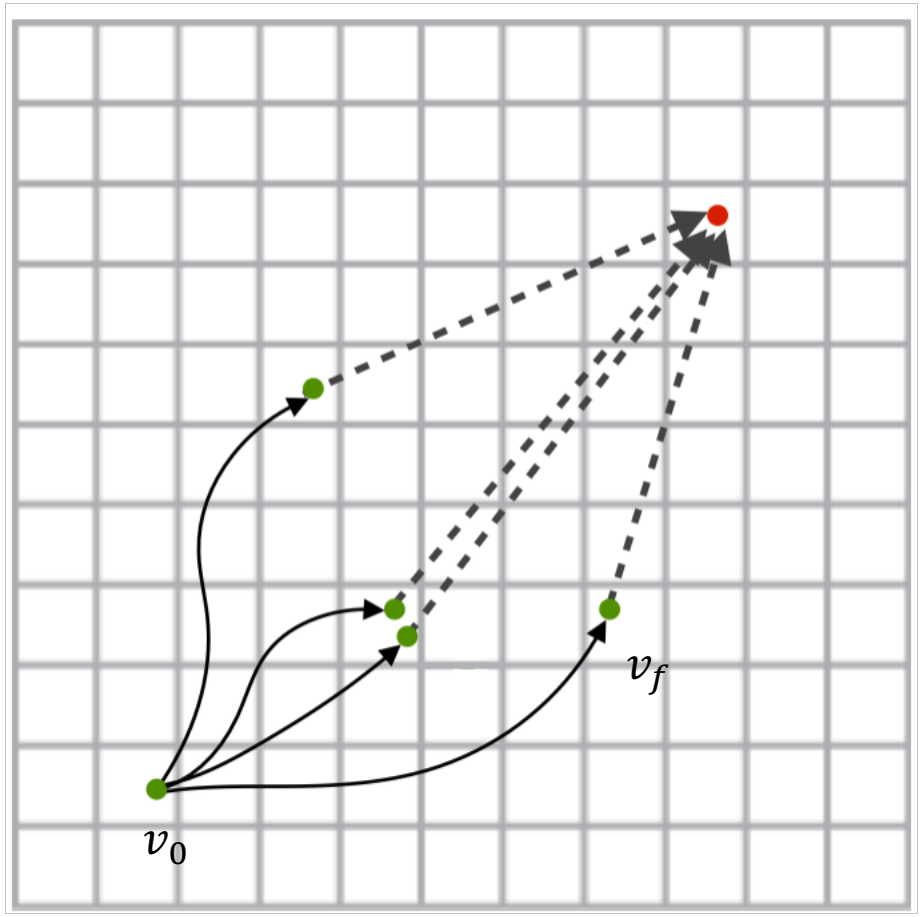
- **5D 时空栅格**：$< x, y, \theta, dir, t >$
- **自车轨迹扩展**：$< q', t+1 > = < f(q, \boldsymbol{u}), t >$
  - 他人/车轨迹预测：$< q'_{exo}, t+1 >= m(h_{exo}, h, c, t)$
  - 若以下条件满足则碰撞：
    - $< q', t+1 >$ 与 $< q'_{exo}, t+1 >$ 几何发生重叠
    - 时间标签必须一致（时空重叠）





Hybrid A* 时空搜索 + 速度控制

- **5D 时空栅格**：$< x, y, \theta, dir, t >$
- **自车轨迹扩展**：$< q', t+1 > = < f(q, \boldsymbol{u}), t >$
  - 他人/车轨迹预测：$< q'_{exo}, t+1 >= m(h_{exo}, h, c, t)$
  - 若以下条件满足则碰撞：
    - $< q', t+1 >$ 与 $< q'_{exo}, t+1 >$ 几何发生重叠
    - 时间标签必须一致（时空重叠）
- **控制输入**：$u =< \phi, v_f >$
  - ϕ = 方向盘角度 或 ϕ = 方向盘的变化率
  - v_f = 终点速度
  - v_f 也需要受到多方面的限制：
    - 交通限速 (<=60)、红绿灯（=0）
    - 曲率限制（与方向盘角度相关）



自动驾驶速度分布生成

任务规划（Mission Planning）

行为规划（Behavior Planning）

运动规划（Motion Planning）

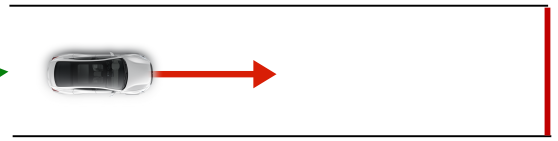
路径规划（Path Planning） | 速度分布（Velocity Profile）

运动控制（Motion Control）

结构化场景

非结构化场景

停车



启动

跟车、变道时与前车交互

其他复杂交互