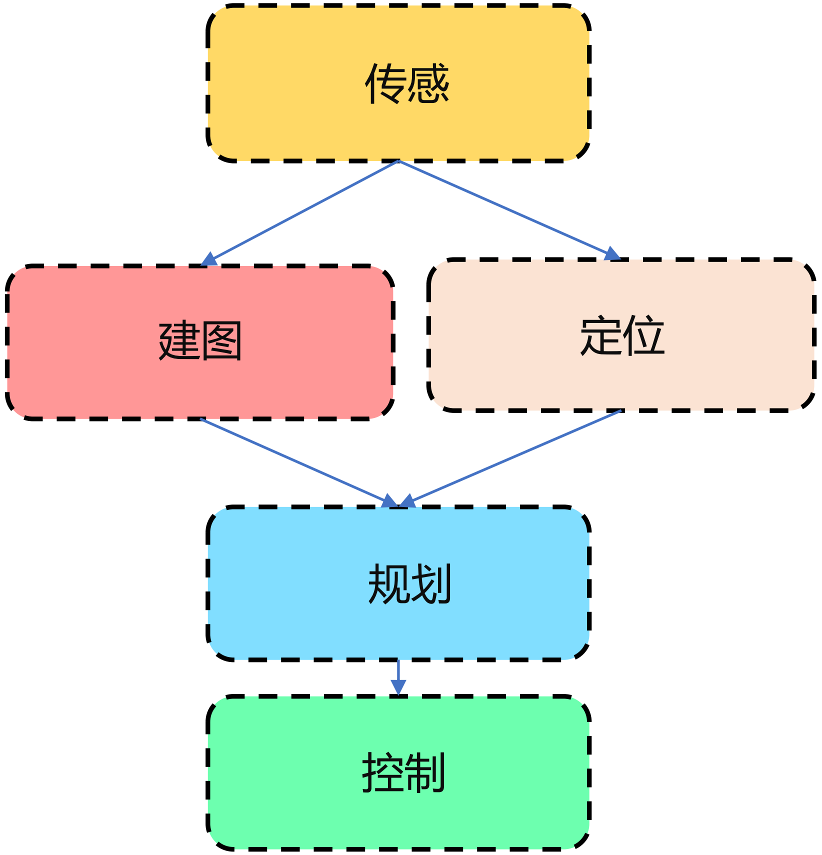
无人机运动规划算法

运动规划基本概念

- 感知-规划-控制：闭环系统



- 基本要求
  - ➢ 安全性：躲避障碍物
  - ➢ 光滑性：节省能量、提升运动连续性
  - ➢ 动力学可行性：可执行、可控制
- 经典链路
  - ➢ 前端：路径搜索
    - 搜索初始可行路径
    - 低维度问题
    - 离散空间
  - ➢ 后端：轨迹生成
    - 搜索机器人可执行轨迹
    - 高维度问题
    - 连续空间

通用导航系统架构

- **估计**
  - 低延迟
  - 高精度 & 一致性
- **感知**
  - 三维 & 稠密感知
  - 地图维护 & 作用于规划

- **规划**
  - 应对复杂 & 未知环境
  - 安全 & 动力学可行性
  - 有限的传感器 & 计算能力
- **控制**
  - 高机动运动
  - 光滑轨迹跟踪

- 定位模块：根据传感器实时输入，实现对机器人自身姿态的轨迹，反馈控制模块。定位模块要求精度、一致性、实时性及恶劣环境下的鲁棒性。
- 建图模块：根据传感器信息，实时构建环境三维几何模型，建立地图用于规划中的碰撞检测。建图模块要求地图稠密度、细粒度及建图实时性。
- 规划模块：以机器人实时位姿估计为轨迹起始状态，导航目标为终止状态，生成符合动力学模型，且保证安全性的运动轨迹。规划模块要求生成轨迹的安全性、光滑性、动力学可行性。

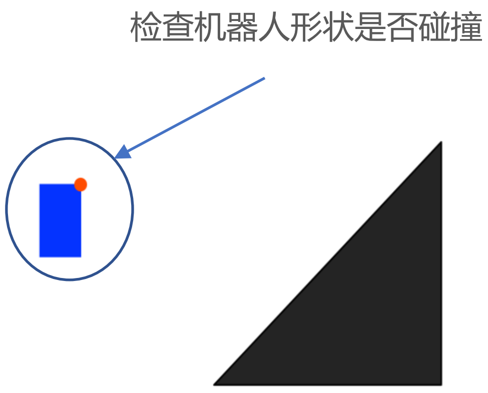
路径规划的作用

构形空间

- 机器人构形: 机器人所有的位置点
- 机器人自由度(DOF): 用来表示机器人构形的最小实数坐标数
- 机器人构形空间: 包含所有可能的机器人构形的n维空间，记作C-space
- 每个机器人姿态都是C-Space中的一个点

构形空间障碍物

- 在工作空间中规划
  - 机器人具有不同的形状和大小
  - 碰撞检测需要知道机器人的几何信息——费时，难度大

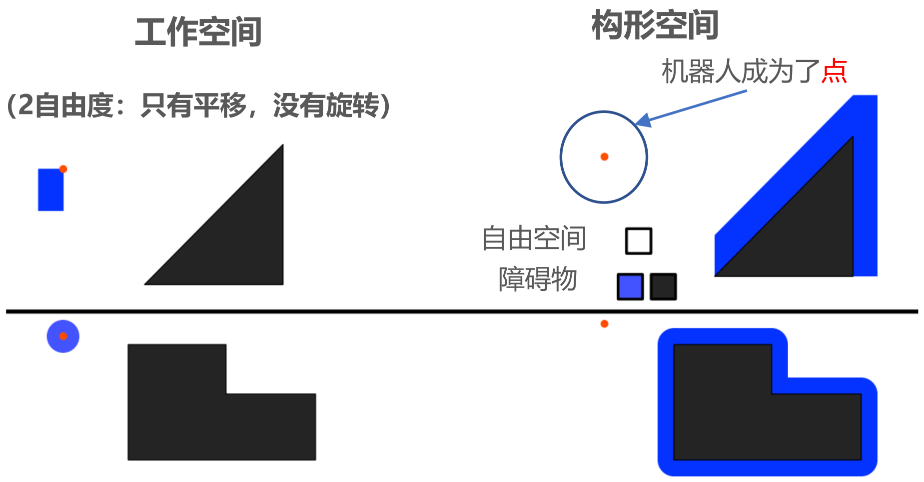


(1) 矩形移动机器人

(2) 圆形移动机器人

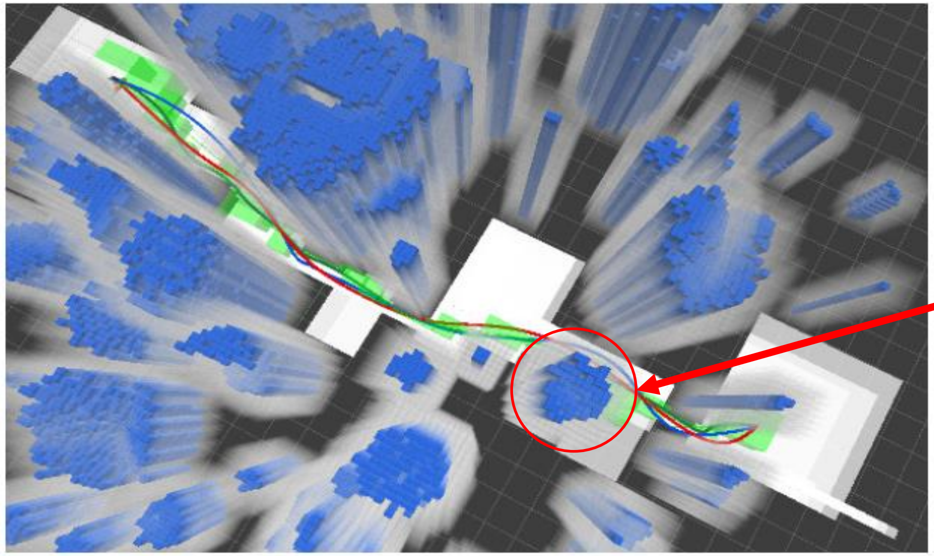
构形空间障碍物

- 在构形空间中规划
  - 机器人表示为C-space中的一个点，如位置（$\boldsymbol{R}^3$中的点），姿态（$SO(3)$中的点）
  - 障碍物需要在构形空间中表示（先于运动规划完成），称为构形空间障碍，或C-obstacle
  - C-space = （C-obstacle） ∪ （C-free）
  - 路径规划就是在C-free中寻找起点q_{start}至终点q_{goal}的路径



工作空间和构形空间障碍物

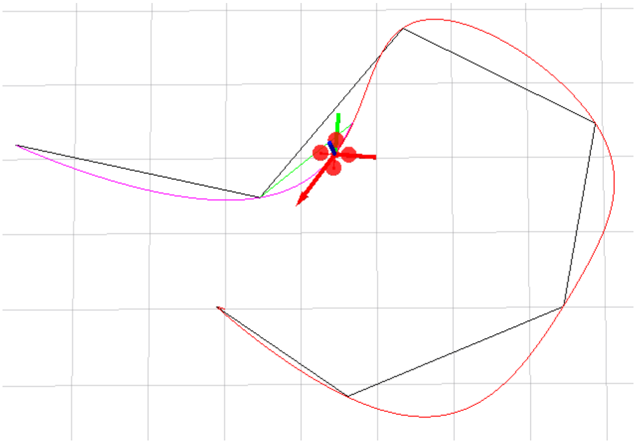
- 工作空间
  - 机器人有形状和大小（不利于运动规划）
- 构形空间： C-space
  - 机器人是一个点（便于运动规划）
  - 运动规划前，障碍物先在C-space中表示
- 在C-space中表示障碍物非常复杂。在实际中使用近似（但是更保守）的表示方法

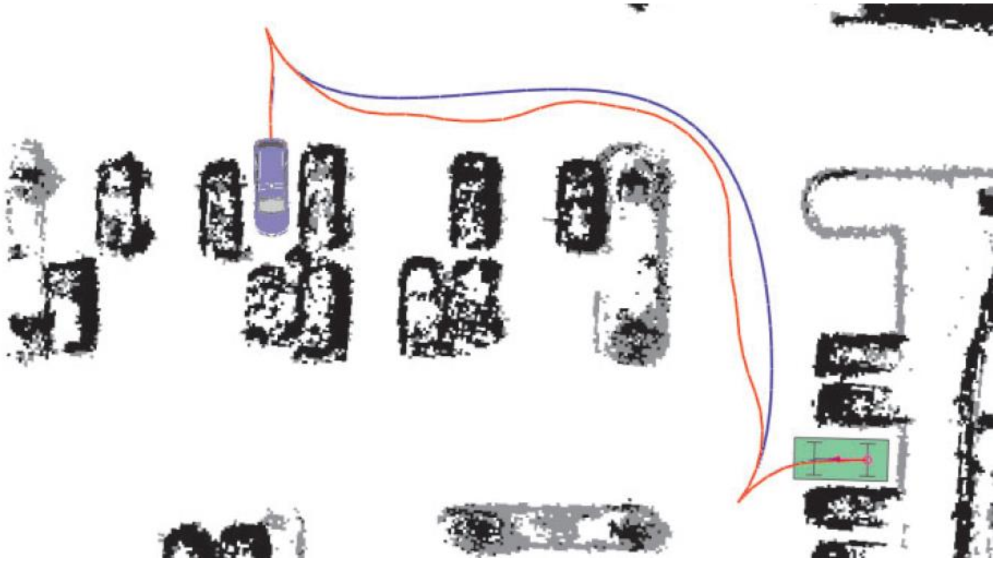


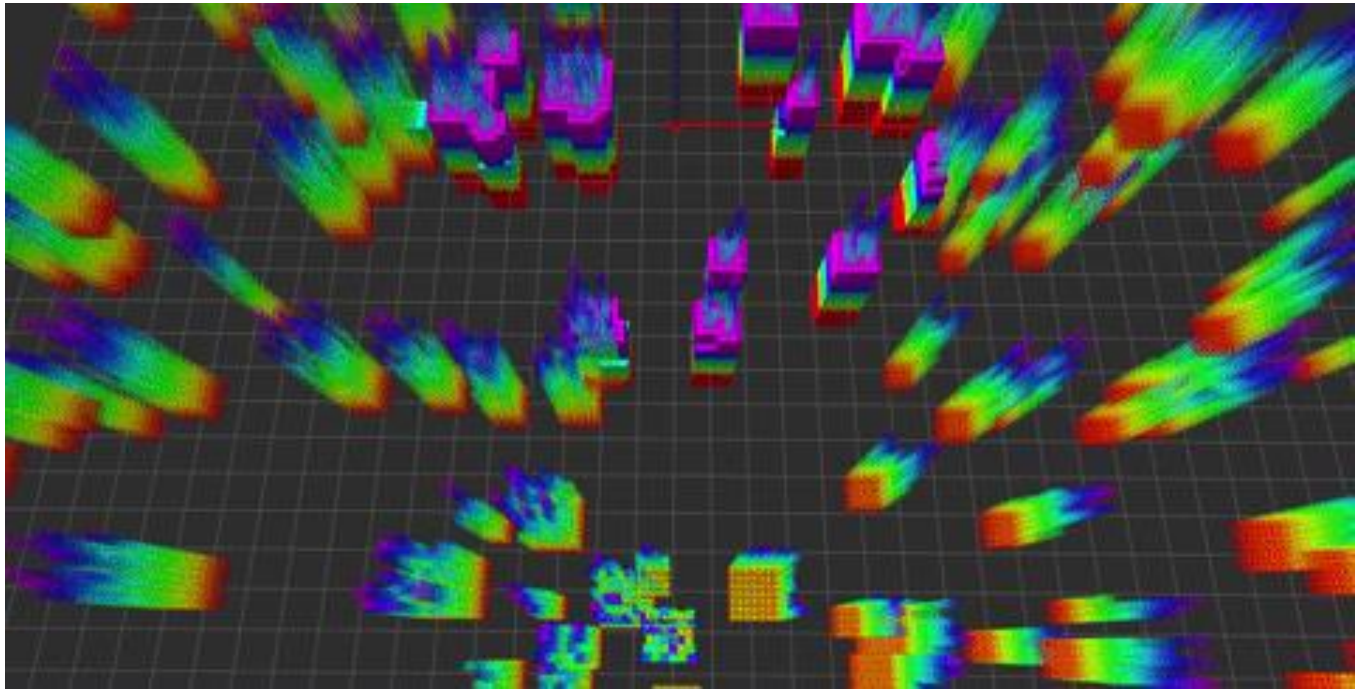
如果我们保守地把机器人建模成一个半径为δ_r的球体，构建C-space可以把所有障碍物向各个方向膨胀δ_r。

轨迹优化的必要性

- 机器人的速度和动力学高阶量无法突变；
- 节约机器人运动过程的能量消耗；
- 保证轨迹执行时间的合理性；
- 确保机器人移动过程中的安全；

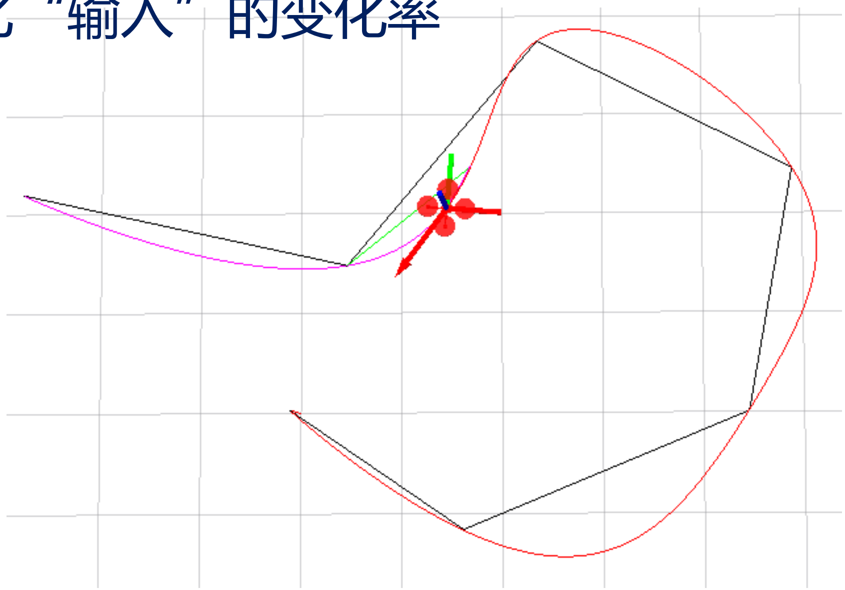






光滑轨迹生成

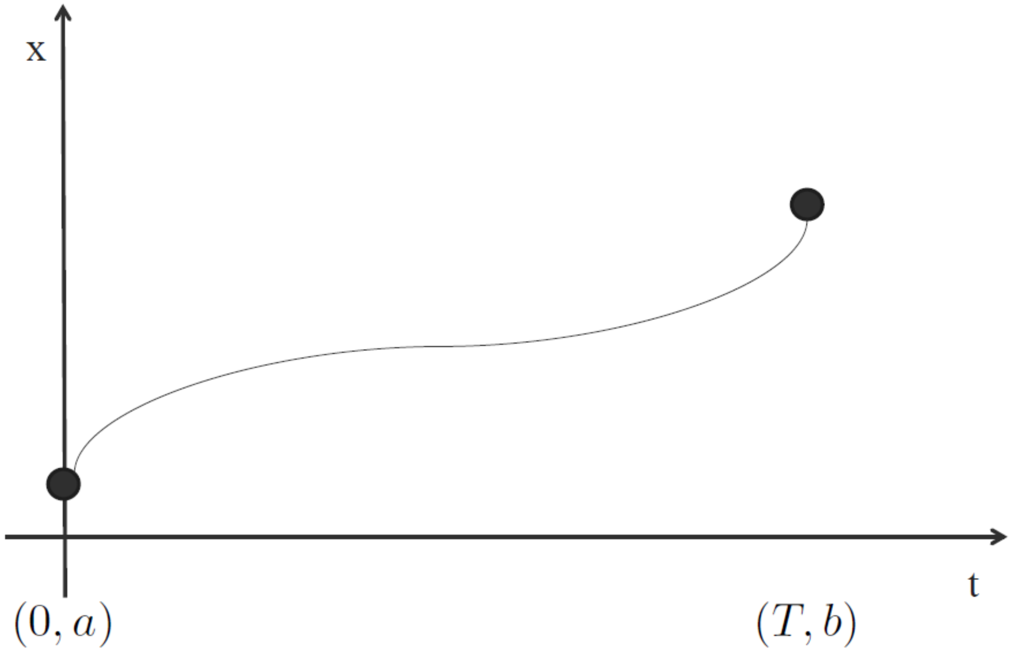
- 边界条件：起始位置（朝向）
- 中间条件：航点位置（朝向）
  - 航点由路径规划得到（A*,RRT*等）
  - 之前3节课的内容
- 光滑性的标准
  - 通常通过最小化“输入”的变化率

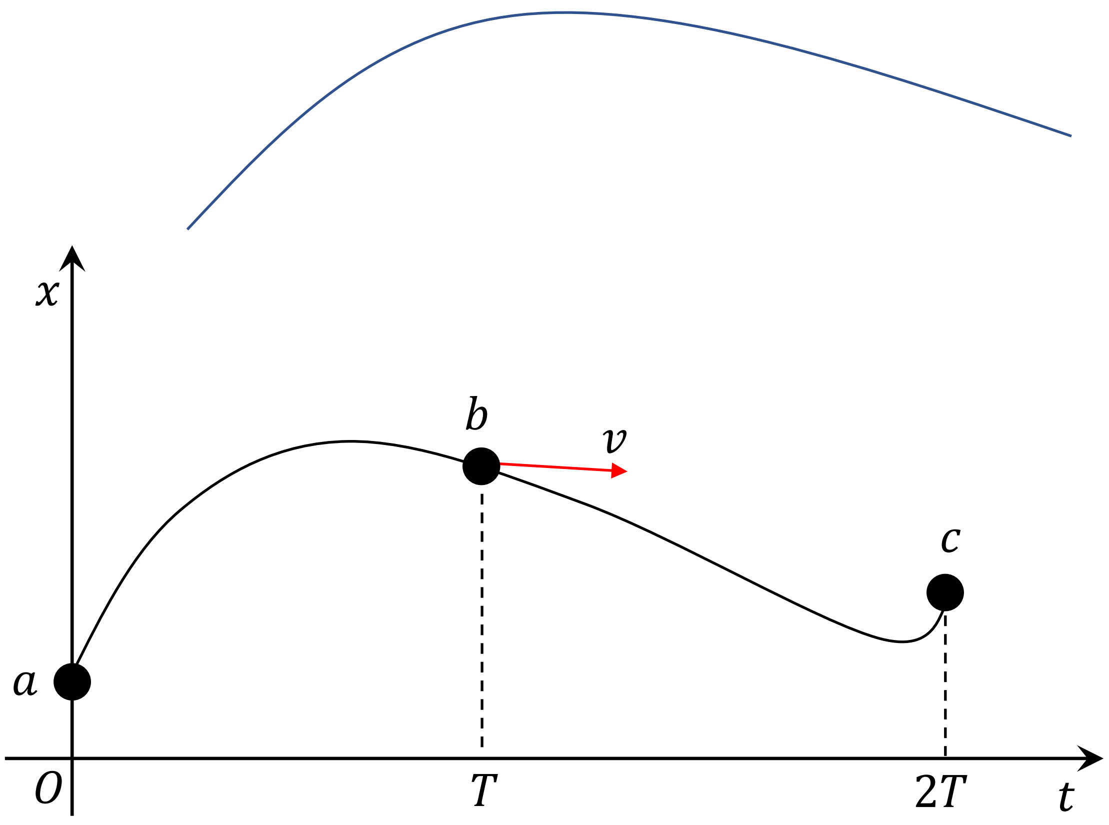


光滑一维轨迹

- 设计一条轨迹$x(t)$ 使得：
  - $x(0) = a$
  - $x(T) = b$

  边界条件
- 轨迹光滑性通过轨迹参数化形式保证





光滑一维轨迹

- 5^{th} 多项式轨迹：
  - $x(t) = p_5 t^5 + p_4 t^4 + p_3 t^3 + p_2 t^2 + p_1 t + p_0$

轨迹参数化

- 边界条件

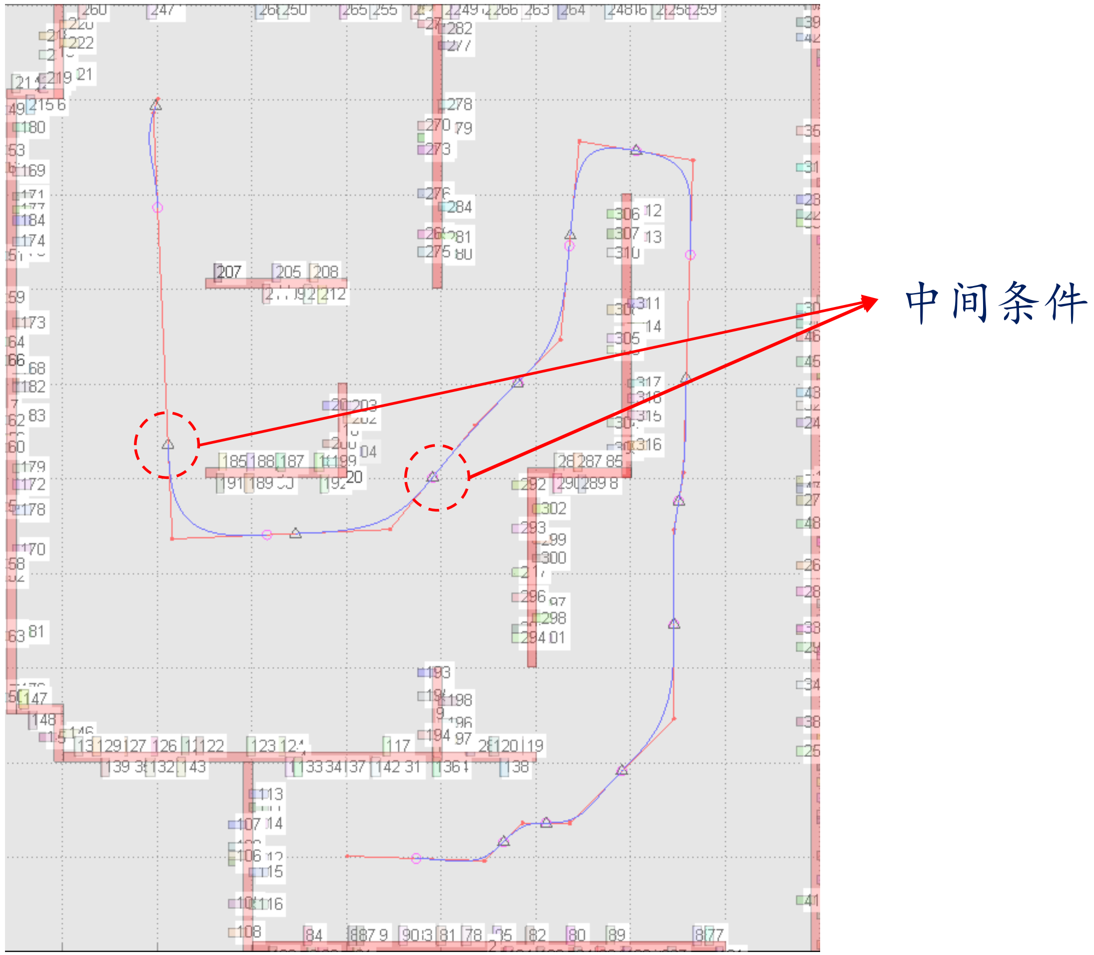
	位置	速度	加速度
t = 0	a	0	0
t = T	b	0	0

- 求解：

$$\begin{bmatrix} a \\ b \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ T^5 & T^4 & T^3 & T^2 & T & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 5T^4 & 4T^3 & 3T^2 & 2T & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 20T^3 & 12T^2 & 6T & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_5 \\ p_4 \\ p_3 \\ p_2 \\ p_1 \\ p_0 \end{bmatrix}$$

光滑多段一维轨迹

- 光滑直线段的转角处
- 倾向于以v作匀速运动
- 倾向于零加速度
- 短的直线段需要特殊处理



光滑多段一维轨迹

- 在每个维度上仍使用相同的参数化形式：
  - $x(t) = p_5 t^5 + p_4 t^4 + p_3 t^3 + p_2 t^2 + p_1 t + p_0$
- 边界条件

	位置	速度	加速度
t = 0	a	v_0	0
t = T	b	v_T	0

- 求解：

$$\begin{bmatrix} a \\ b \\ v_0 \\ v_T \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ T^5 & T^4 & T^3 & T^2 & T & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 5T^4 & 4T^3 & 3T^2 & 2T & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 20T^3 & 12T^2 & 6T & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_5 \\ p_4 \\ p_3 \\ p_2 \\ p_1 \\ p_0 \end{bmatrix}$$

微分平坦

微分平坦

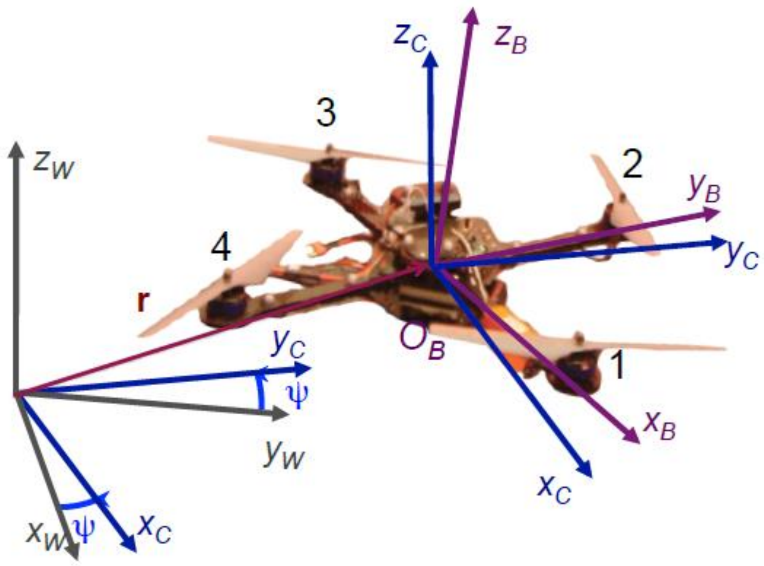
- 四旋翼的状态和输入可以写作四个平坦（flat）的输出变量和它们导数的代数方程
  - 使得轨迹能够自动生成
  - 任何平坦的输出变量空间中的光滑轨迹（其导数在合理范围内）都能够被欠驱动的四旋翼系统跟随
  - 一种可行的选择
    - $\boldsymbol{\sigma} = [x, y, z, \psi]^T$
  - 平坦输出空间中的轨迹
    - $\boldsymbol{\sigma}(t) = [T_0, T_M] \to \mathbb{R}^3 \times SO(2)$

Minimum Snap Trajectory Generation and Control for Quadrotors, Daniel Mellinger and Vijay Kumar

四旋翼动力特性

- 四旋翼状态
  - 位置，朝向，线速度，角速度

$$\mathbf{X} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z]^T$$



- 非线性动态方程

牛顿方程：
$$m\ddot{\boldsymbol{p}} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \boldsymbol{R} \begin{bmatrix} 0 \\ 0 \\ \boxed{F_1 + F_2 + F_3 + F_4} \end{bmatrix} \quad \boldsymbol{u_1}$$

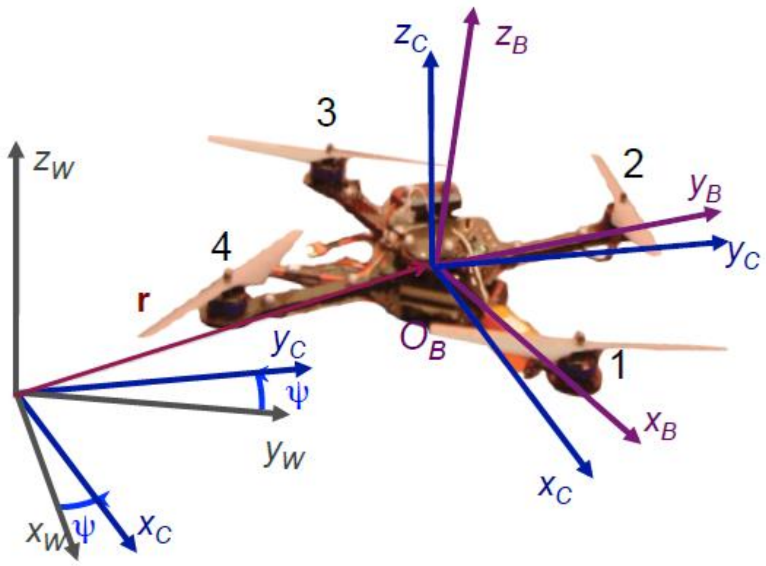
欧拉方程：
$$\boldsymbol{I} \cdot \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} + \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \times \boldsymbol{I} \cdot \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{array}{c} \left[\boxed{\begin{array}{c} l(F_2 - F_4) \\ l(F_3 - F_1) \\ M_1 - M_2 + M_3 - M_4 \end{array}}\right] \end{array} \begin{array}{l} \boldsymbol{u}_2 \\ \boldsymbol{u}_3 \\ \boldsymbol{u}_4 \end{array}$$

微分平坦

- 四旋翼状态
  - 位置，朝向，线速度，角速度

$$\mathbf{X} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \boxed{\omega_x, \omega_y, \omega_z}]^T$$

体坐标系中的体角加速度



  - 运动方程

$$m\ddot{\mathbf{p}} = -mg\mathbf{z}_W + u_1\mathbf{z}_B.$$

$$\boldsymbol{\omega}_B = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \quad , \qquad \dot{\boldsymbol{\omega}}_B = \boldsymbol{I}^{-1}\left[-\boldsymbol{\omega}_B \times \boldsymbol{I} \cdot \boldsymbol{\omega}_B + \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix}\right]$$

  - 位置，速度和加速度是平坦输出的微分

Minimum Snap Trajectory Generation and Control for Quadrotors, Daniel Mellinger and Vijay Kumar

微分平坦

- 朝向
  - 四旋翼状态：

$$\mathbf{X} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z]^T$$

  - 从运动方程可得：

$$\mathbf{z}_B = \frac{\mathbf{t}}{\|\mathbf{t}\|}, \mathbf{t} = [\ddot{\boldsymbol{\sigma}}_1, \ddot{\boldsymbol{\sigma}}_2, \ddot{\boldsymbol{\sigma}}_3 + g]^T$$

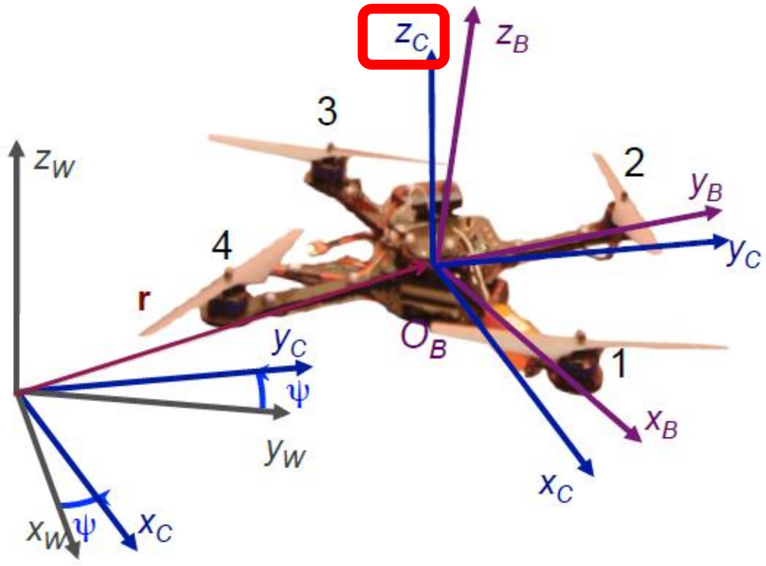
  - 定义偏航角矢量（ Z-X-Y 欧拉角 ）：

$$\mathbf{x}_C = [cos\boldsymbol{\sigma}_4, sin\boldsymbol{\sigma}_4, 0]^T$$

  - 姿态（body坐标系朝向）可以由平坦输出表示

$$\mathbf{y}_B = \frac{\mathbf{z}_B \times \mathbf{x}_C}{\|\mathbf{z}_B \times \mathbf{x}_C\|}, \quad \mathbf{x}_B = \mathbf{y}_B \times \mathbf{z}_B \qquad \boldsymbol{R}_B = [\mathbf{x}_B \ \ \mathbf{y}_B \ \ \mathbf{z}_B]$$

“不倾斜”的坐标系



$$\boldsymbol{\sigma} = [x, y, z, \psi]^T$$

微分平坦

- 角速度
  - 四旋翼状态：

$$\mathbf{X} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \omega_z, \omega_y, \omega_z]^T$$

  - 取运动方程的微分

$$m\ddot{\boldsymbol{p}} = -mg\mathbf{z}_W + u_1\mathbf{z}_B. \longrightarrow m\dot{\boldsymbol{a}} = \dot{u}_1\mathbf{z}_B + \boxed{\boldsymbol{\omega}_{BW}} \times u_1\,\mathbf{z}_B$$

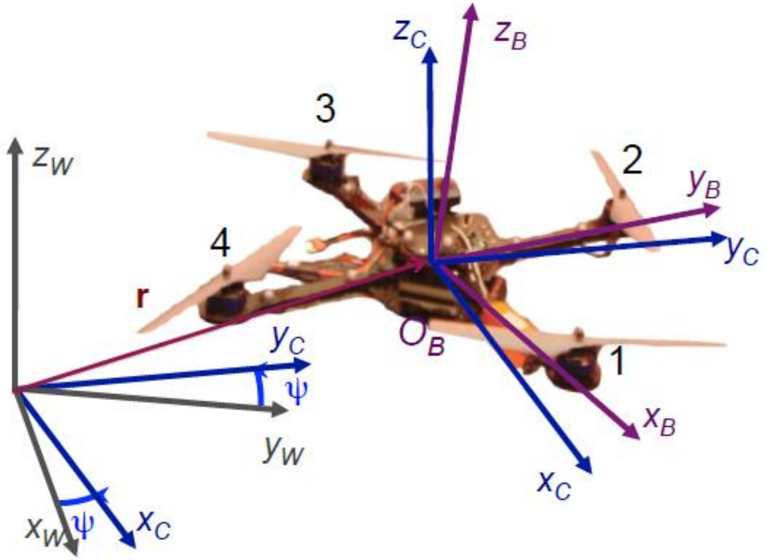
世界系中的体角速度

  - 四旋翼仅有垂直的升力：

$$\dot{u}_1 = \mathbf{z}_B \cdot m\dot{\boldsymbol{a}}$$

  - 可得到：

$$\mathbf{h}_\omega = \boldsymbol{\omega}_{BW} \times \mathbf{z}_B = \frac{m}{u_1}(\dot{\boldsymbol{a}} - (\mathbf{z}_B \cdot \dot{\boldsymbol{a}})\mathbf{z}_B).$$



微分平坦

- 角速度
  - 四旋翼状态：

$$\mathbf{X} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z]^T$$

  - 已得到：

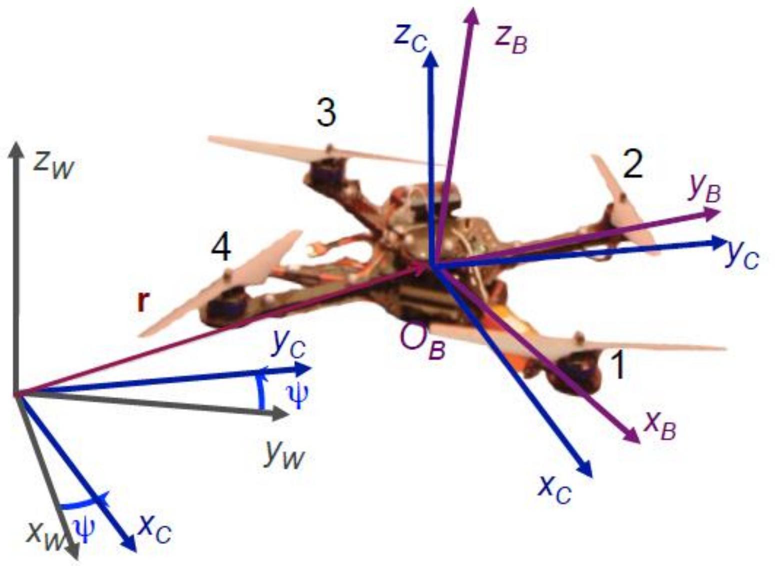
$$\mathbf{h}_\omega = \boldsymbol{\omega}_{BW} \times \mathbf{z}_B = \frac{m}{u_1}(\dot{\boldsymbol{a}} - (\mathbf{z}_B \cdot \dot{\boldsymbol{a}})\mathbf{z}_B).$$

  - 又已知：

$$\boldsymbol{\omega}_{BW} = \omega_x \mathbf{x}_B + \omega_y \mathbf{y}_B + \omega_z \mathbf{z}_B$$ ，代入上式：

  - x_B和 y_B方向的角速度可以得到：

$$\omega_x = -\mathbf{h}_\omega \cdot \mathbf{y}_B, \quad \omega_y = \mathbf{h}_\omega \cdot \mathbf{x}_B$$



微分平坦

- 角速度
  - 四旋翼状态：

$$\mathbf{X} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z]^T$$

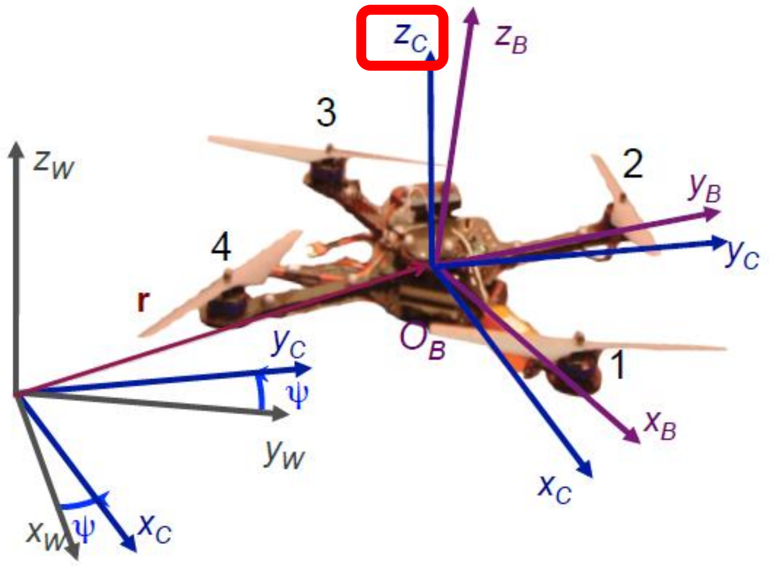
  - 已得到：

$$\mathbf{h}_\omega = \boldsymbol{\omega}_{BW} \times \mathbf{z}_B = \frac{m}{u_1}(\dot{\boldsymbol{a}} - (\mathbf{z}_B \cdot \dot{\boldsymbol{a}})\mathbf{z}_B).$$

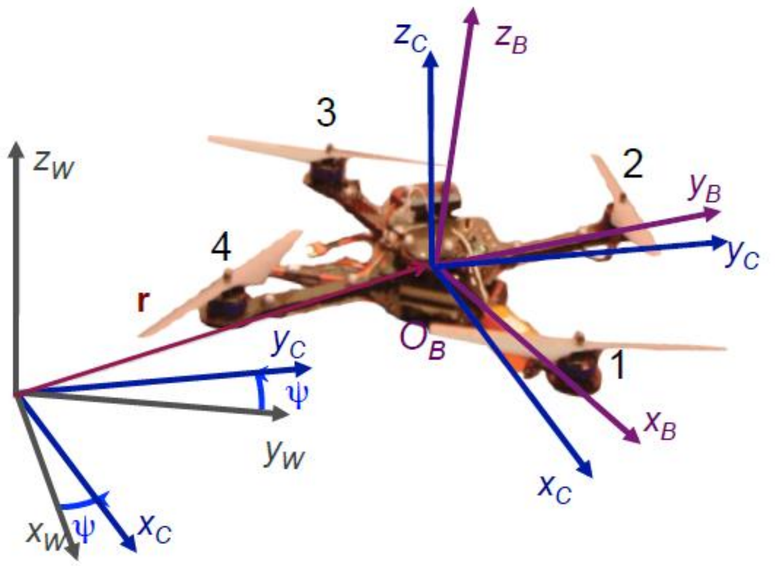
  - 因为 $\boldsymbol{\omega}_{BW} = \boldsymbol{\omega}_{BC} + \boldsymbol{\omega}_{CW}$，且 $\boldsymbol{\omega}_{BC}$ 没有$\mathbf{z}_B$的成分

$$\omega_z = \boldsymbol{\omega}_{BW} \cdot \mathbf{z}_B = \boldsymbol{\omega}_{CW} \cdot \mathbf{z}_B = \dot{\psi}\mathbf{z}_W \cdot \mathbf{z}_B.$$

“不倾斜”的坐标系



微分平坦



- 总结
  - 四旋翼状态:

$$\mathbf{X} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \omega_x, \omega_y, \omega_z]^T$$

  - 平坦输出:
    - $\boldsymbol{\sigma} = [x, y, z, \psi]^T$
  - 位置，速度，加速度:
    - 平坦输出的微分
  - 姿态:

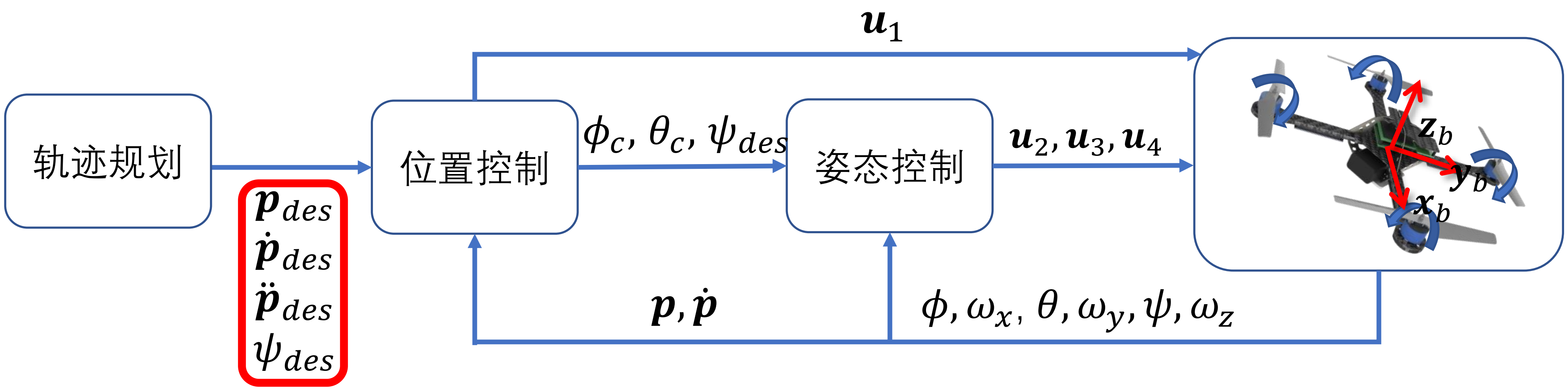
$$\mathbf{x}_C = [cos\sigma_4, sin\sigma_4, 0]^T \qquad \boldsymbol{R}_B = [\mathbf{x}_B \ \mathbf{y}_B \ \mathbf{z}_B]$$

→ 你只需要记住无人机的规划可以在x, y, z, ψ中进行。

  - 角速度:

$$\omega_x = -\mathbf{h}_\omega \cdot \mathbf{y}_B, \ \ \omega_y = \mathbf{h}_\omega \cdot \mathbf{x}_B, \ \ \omega_z = \dot{\psi}\mathbf{z}_W \cdot \mathbf{z}_B$$

规划-控制闭环



非线性动态方程

牛顿方程：
$$m\ddot{\boldsymbol{p}} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \boldsymbol{R} \begin{bmatrix} 0 \\ 0 \\ F_1 + F_2 + F_3 + F_4 \end{bmatrix}$$

欧拉方程：
$$\boldsymbol{I} \cdot \begin{bmatrix} \dot{\omega_x} \\ \dot{\omega_y} \\ \dot{\omega_z} \end{bmatrix} + \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \times \boldsymbol{I} \cdot \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} l(F_2 - F_4) \\ l(F_3 - F_1) \\ M_1 - M_2 + M_3 - M_4 \end{bmatrix}$$

多项式轨迹

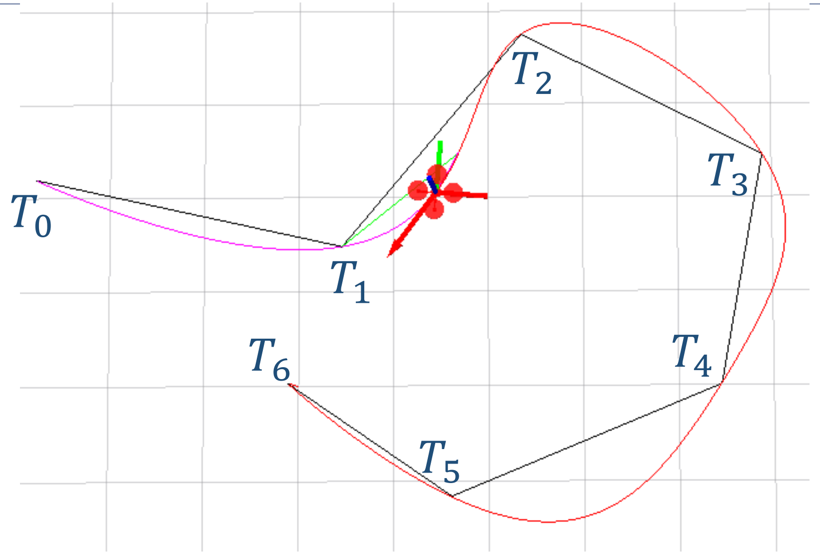
- 平坦输出：
  - $\boldsymbol{\sigma} = [x, y, z, \psi]^T$

- 平坦输出空间中的轨迹：
  - $\boldsymbol{\sigma}(t) = [T_0, T_M] \rightarrow \mathbb{R}^3 \times SO(2)$

- 多项式方程可用来指定平坦输出空间中的轨迹
  - 利用多项式的阶数判定平滑的标准
  - 简单且闭式地计算微分
  - 在三维中解耦轨迹生成

最小化Snap轨迹生成

最小化Snap的轨迹生成



问题定义：

- 轨迹的**初始（或初末）状态给定**；
- 轨迹在**预定时刻**需经过**预设航点**；
- 轨迹的**4阶导数处处存在**
- 最小化轨迹**4阶导数平方幅值积分**。

$$f(t)=\begin{cases} f_1(t) \doteq \sum_{i=0}^{N} p_{1,i}t^i & T_0 \le t \le T_1 \\ f_2(t) \doteq \sum_{i=0}^{N} p_{2,i}t^i & T_1 \le t \le T_2 \\ \quad\vdots & \quad\vdots \\ f_M(t) \doteq \sum_{i=0}^{N} p_{M,i}t^i & T_{M-1} \le t \le T_M \end{cases}$$

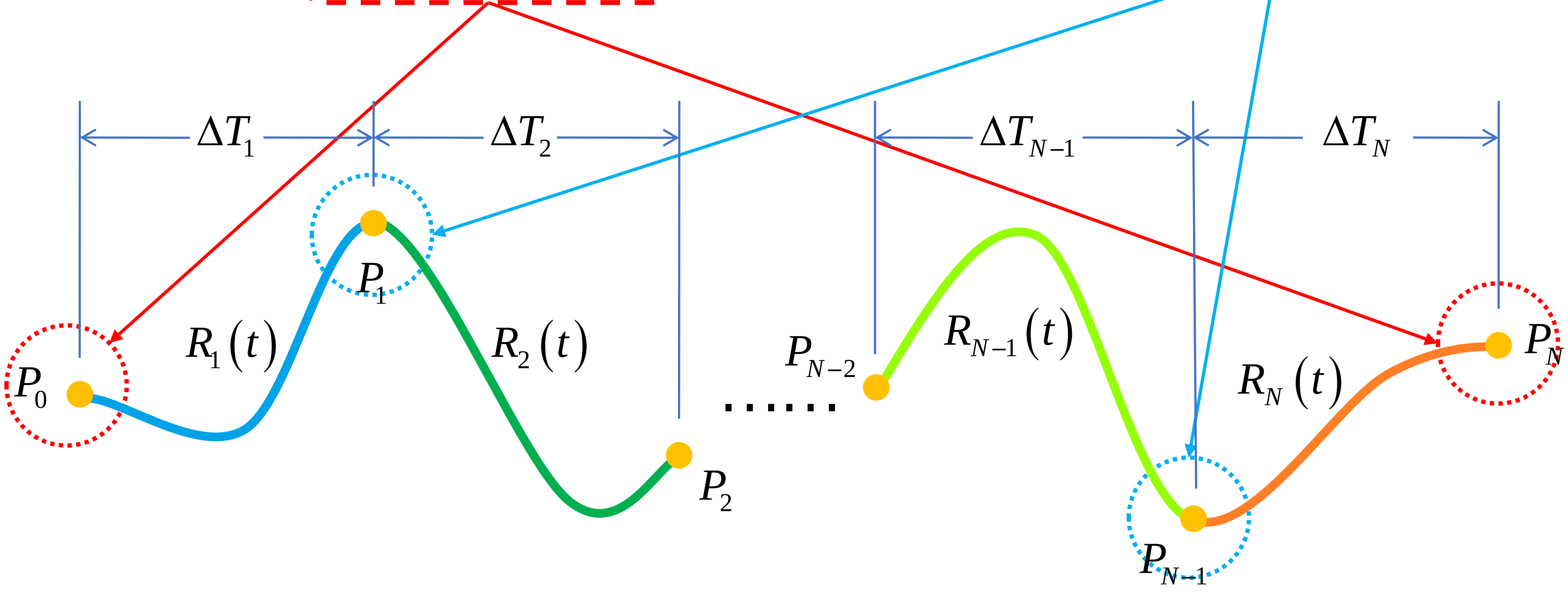
重要性质：

- 最优轨迹的参数化形式为**分段多项式**；
- 每段多项式的次数不超过$2\times 4-1=7$，即**每段均可被7次多项式表征**。

最小化Snap的轨迹生成

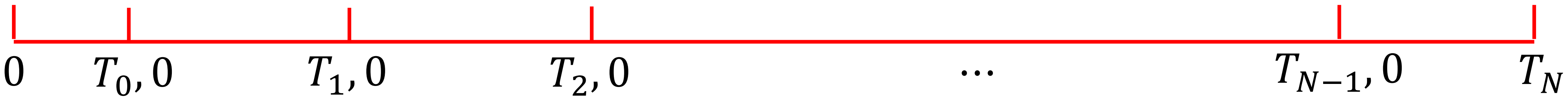
约束：

- 导数给定值约束：$\begin{cases} f_j^{(k)}(T_{j-1}) & = x_{0,j}^{(k)} \\ f_j^{(k)}(T_j) & = x_{T,j}^{(k)} \end{cases}$
- 高阶连续性约束：$f_j^{(k)}(T_j) = f_{j+1}^{(k)}(T_j)$



最小化Snap的轨迹生成

两种时间参数方案



方案1：相对每一段初始的时间 优点：数值稳定性高

方案2：相对第一段初始的时间 优点：数学表达清楚

最小化Snap的轨迹生成

目标函数的解析表达

半正定二次型

$$\boxed{J_j(T) = \mathbf{p}_j^T \mathbf{Q}_j \mathbf{p}_j}$$

$$f(t) = \sum_i p_i t^i$$

$$\Longrightarrow f^{(4)}(t) = \sum_{i \geq 4} i(i-1)(i-2)(i-3)t^{i-4} p_i$$

$$\Longrightarrow \left(f^{(4)}(t)\right)^2 = \sum_{i \geq 4, l \geq 4} i(i-1)(i-2)(i-3)l(l-1)(l-2)(l-3)t^{i+l-8} p_i \, p_l$$

$$\Longrightarrow J(T) = \int_{T_{j-1}}^{T_j} \left(f^4(t)\right)^2 dt = \sum_{i \geq 4, l \geq 4} \frac{i(i-1)(i-2)(i-3)j(l-1)(l-2)(l-3)}{i+l-7} \left(T_j^{i+l-7} - T_{j-1}^{i+l-7}\right) p_i \, p_l$$

$$\Longrightarrow J(T) = \int_{T_{j-1}}^{T_j} \left(f^4(t)\right)^2 dt = \begin{bmatrix} \vdots \\ p_i \\ \vdots \end{bmatrix}^T \begin{bmatrix} & & \vdots & & \\ \cdots & & \dfrac{i(i-1)(i-2)(i-3)l(l-1)(l-2)(l-3)}{i+l-7} T^{i+l-7} & & \cdots \\ & & \vdots & & \end{bmatrix} \begin{bmatrix} \vdots \\ p_l \\ \vdots \end{bmatrix}$$

最小化Snap的轨迹生成

指定导数值约束：0阶导给定即航点坐标

- $f_j^{(k)}(T_j) = x_j^{(k)}$

$$\Rightarrow \sum_{i\geq k} \frac{i!}{(i-k)!} T_j^{i-k} p_{j,i} = x_{T,j}^{(k)}$$

$$\Rightarrow \begin{bmatrix} \cdots & \frac{i!}{(i-k)!} T_j^{i-k} & \cdots \end{bmatrix} \begin{bmatrix} \vdots \\ p_{j,i} \\ \vdots \end{bmatrix} = x_{T,j}^{(k)}$$

$$\Rightarrow \begin{bmatrix} \cdots & \frac{i!}{(i-k)!} T_{j-1}^{i-k} & \cdots \\ \cdots & \frac{i!}{(i-k)!} T_j^{i-k} & \cdots \end{bmatrix} \begin{bmatrix} \vdots \\ p_{j,i} \\ \vdots \end{bmatrix} = \begin{bmatrix} x_{0,j}^{(k)} \\ x_{T,j}^{(k)} \end{bmatrix}$$

$$\Rightarrow \mathbf{A}_j \mathbf{p}_j = \mathbf{d}_j$$

$$x(t) = p_5 t^5 + p_4 t^4 + p_3 t^3 + p_2 t^2 + p_1 t + p_0$$

$$\begin{aligned} x(0) &= \cdots, x(T) = \cdots \\ \dot{x}(0) &= \cdots, \dot{x}(T) = \cdots \\ \ddot{x}(0) &= \cdots, \ddot{x}(T) = \cdots \\ &\vdots \end{aligned}$$

$$p_0 = \cdots,$$
$$p_5 T^5 + p_4 T^4 + p_3 T^3 + p_2 T^2 + p_1 T + p_0 = \cdots$$

$$[T^5, T^4, T^3, T^2, T, 1] \begin{bmatrix} p_5 \\ p_4 \\ p_3 \\ p_2 \\ p_1 \\ p_0 \end{bmatrix} = \cdots$$

最小化Snap的轨迹生成

高阶连续性约束：保证两段之间的4阶导连续

$$f_j^{(k)}(T_j) = f_{j+1}^{(k)}(T_j)$$

$$\Rightarrow \sum_{i\geq k} \frac{i!}{(i-k)!} T_j^{i-k} p_{j,i} - \sum_{l\geq k} \frac{l!}{(l-k)!} T_j^{l-k} p_{j+1,l} = 0$$

$$\Rightarrow \begin{bmatrix} \cdots & \frac{i!}{(i-k)!} T_j^{i-k} & \cdots & -\frac{l!}{(l-k)!} T_j^{l-k} & \cdots \end{bmatrix} \begin{bmatrix} \vdots \\ p_{j,i} \\ \vdots \\ p_{j+1,l} \\ \vdots \end{bmatrix} = 0$$

$$\Rightarrow \begin{bmatrix} \mathbf{A}_j & -\mathbf{A}_{j+1} \end{bmatrix} \begin{bmatrix} \mathbf{p}_j \\ \mathbf{p}_{j+1} \end{bmatrix} = 0$$

最小化Snap的轨迹生成

线性等式约束的二次规划

$$
\begin{aligned}
\min \quad & \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}^T \begin{bmatrix} \mathbf{Q}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Q}_M \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix} \\
\text{s.t.} \quad & \mathbf{A}_{\mathrm{eq}} \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix} = \mathbf{d}_{eq}
\end{aligned}
$$

最小化Snap的轨迹生成是一个典型的**凸优化**问题。

轨迹优化的实验验证


Aggressive Quadrotor Part II
Daniel Mellinger and Vijay Kumar
GRASP Lab, University of Pennsylvania

解析解法

决策变量映射

- 直接优化多项式轨迹在数值上不稳定
- 更倾向于替代变量，使得优化每段端点的微分
- 我们有$\boldsymbol{M}_{\mathrm{j}}\mathbf{p}_j = \mathbf{d}_j$，$\boldsymbol{M}_{\mathrm{j}}$是一个映射矩阵，将多项式系数映射至微分

$$J = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}^T \begin{bmatrix} \mathbf{Q}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Q}_M \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix} \qquad J = \begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_M \end{bmatrix}^T \begin{bmatrix} \boldsymbol{M}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{M}_M \end{bmatrix}^{-T} \begin{bmatrix} \mathbf{Q}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Q}_M \end{bmatrix} \begin{bmatrix} \boldsymbol{M}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{M}_M \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_M \end{bmatrix}$$

Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments, Charles Richter, Adam Bry, and Nicholas Roy

决策变量映射

$$J=\begin{bmatrix}\mathbf{p}_1\\ \vdots\\ \mathbf{p}_M\end{bmatrix}^T\begin{bmatrix}\mathbf{Q}_1 & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \ddots & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{Q}_M\end{bmatrix}\begin{bmatrix}\mathbf{p}_1\\ \vdots\\ \mathbf{p}_M\end{bmatrix} \quad + \quad \boldsymbol{M}_{\mathrm{j}}\mathbf{p}_j=\mathbf{d}_j$$

$$J=\begin{bmatrix}\mathbf{d}_1\\ \vdots\\ \mathbf{d}_M\end{bmatrix}^T\begin{bmatrix}\boldsymbol{M}_1 & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \ddots & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \boldsymbol{M}_M\end{bmatrix}^{-T}\begin{bmatrix}\mathbf{Q}_1 & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \ddots & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{Q}_M\end{bmatrix}\begin{bmatrix}\boldsymbol{M}_1 & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \ddots & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \boldsymbol{M}_M\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{d}_1\\ \vdots\\ \mathbf{d}_M\end{bmatrix}$$

$$x(t)=p_5t^5+p_4t^4+p_3t^3+p_2t^2+p_1t+p_0$$
$$x'(t)=5p_5t^4+4p_4t^3+3p_3t^2+2p_2t+p_1$$
$$x''(t)=20p_5t^3+12p_4t^2+6p_3t+2p_2$$

$$M=\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 2 & 0 & 0\\ T^5 & T^4 & T^3 & T^2 & T & 1\\ 5T^4 & 4T^3 & 3T^2 & 2T & 1 & 0\\ 20T^3 & 12T^2 & 6T & 2 & 0 & 0\end{bmatrix}$$

Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments, Charles Richter, Adam Bry, and Nicholas Roy

分离固定变量和自由变量

- 使用选择矩阵C来分离自由变量($\mathbf{d}_P$）和受约束变量($\mathbf{d}_F$)
  - 自由变量：微分不受指定，只受连续性约束

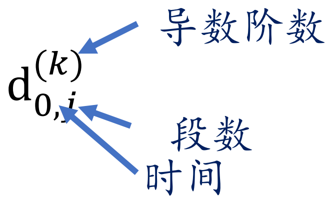
$$\mathbf{C}^T \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix} = \begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_M \end{bmatrix} \Rightarrow J = \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix}^T \underbrace{\mathbf{C}\boldsymbol{M}^{-T}\mathbf{Q}\boldsymbol{M}^{-1}\mathbf{C}^T}_{\mathbf{R}} \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix} = \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix}^T \begin{bmatrix} \mathbf{R}_{FF} & \mathbf{R}_{FP} \\ \mathbf{R}_{PF} & \mathbf{R}_{PP} \end{bmatrix} \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix}$$

- 转变为一个不受约束的二次规划问题，可直接闭式求解

$$J = \mathbf{d}_F^T \mathbf{R}_{FF} \mathbf{d}_F + \mathbf{d}_F^T \mathbf{R}_{FP} \mathbf{d}_P + \mathbf{d}_P^T \mathbf{R}_{PF} \mathbf{d}_F + \mathbf{d}_P^T \mathbf{R}_{PP} \mathbf{d}_P$$

$$\mathbf{d}_P^* = -\mathbf{R}_{PP}^{-1} \mathbf{R}_{FP}^T \mathbf{d}_F$$

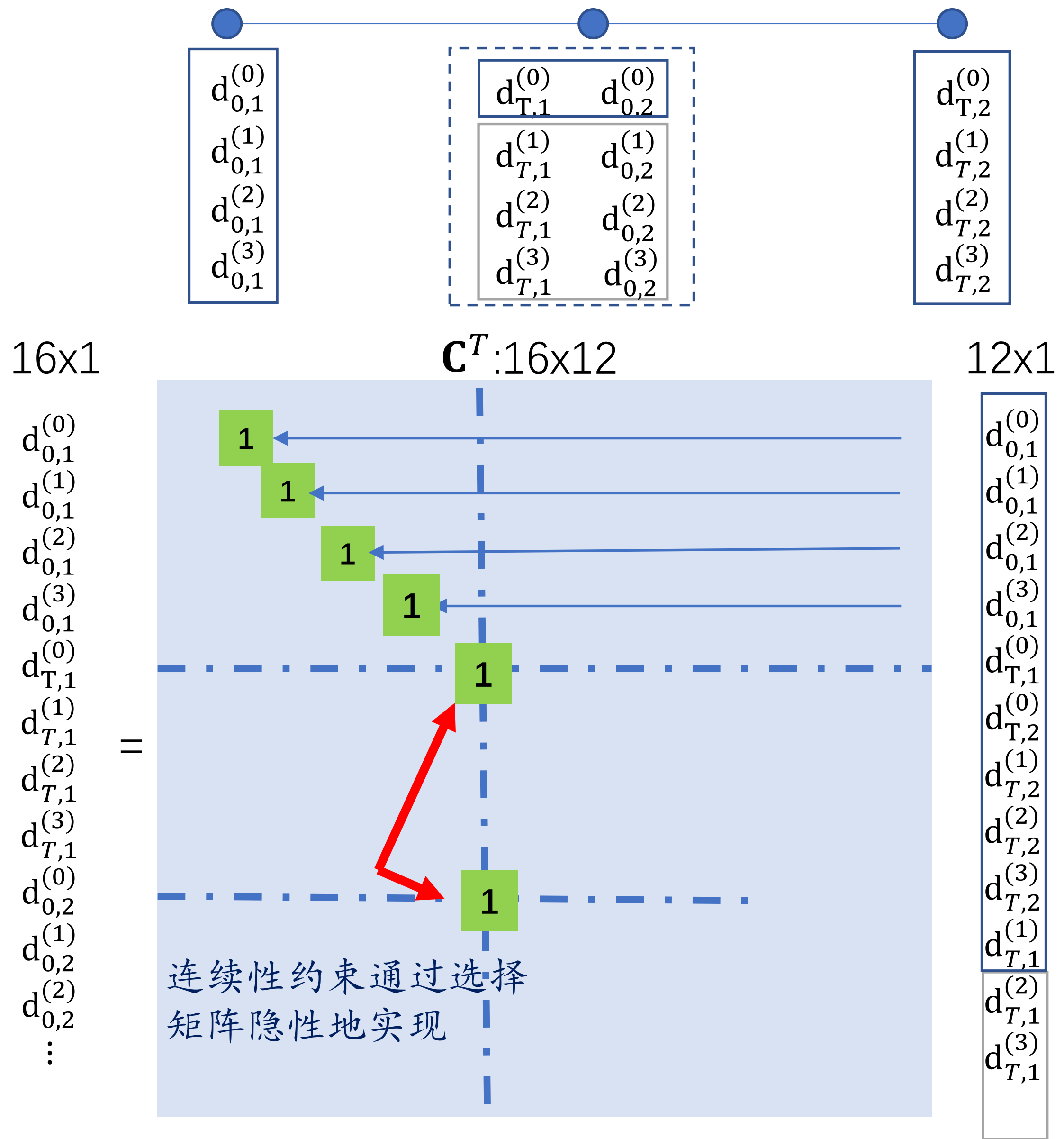
构建选择矩阵



固定的微分项：固定的起点，目标状态以及中间位置

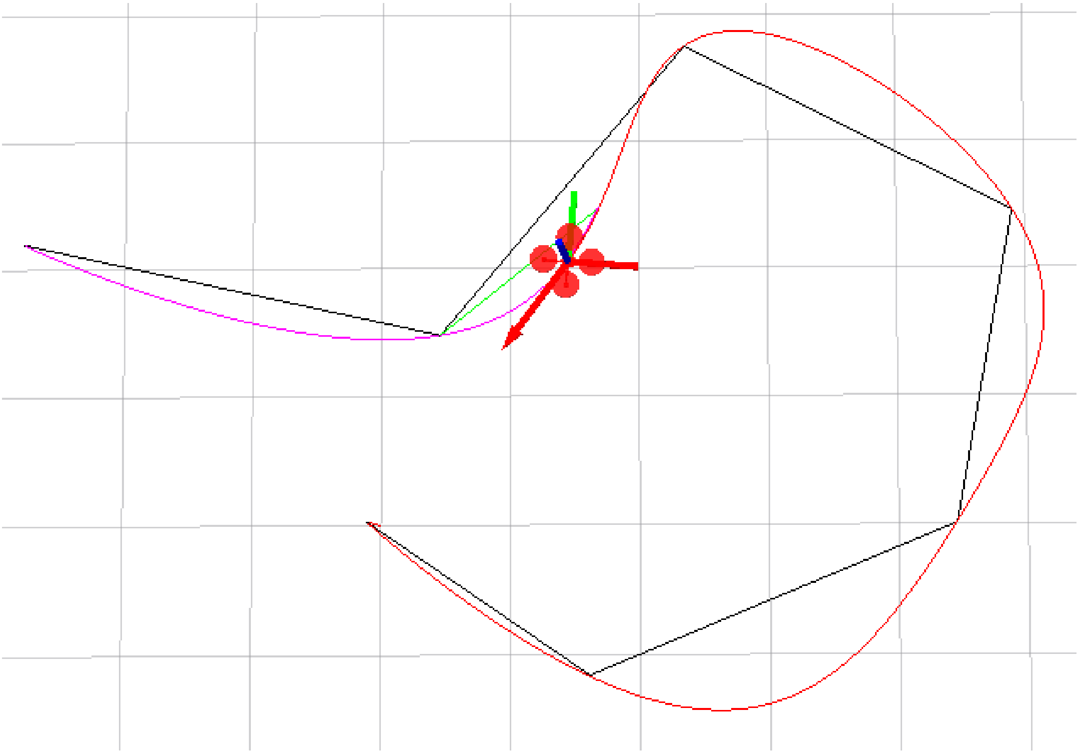
自由的微分项：在中间连接处的所有微分项

$$\begin{bmatrix} \mathbf{d}_1 \\ \vdots \\ \mathbf{d}_M \end{bmatrix} = \mathbf{C}^T \begin{bmatrix} \mathbf{d}_F \\ \mathbf{d}_P \end{bmatrix}$$



和凸优化结果相同

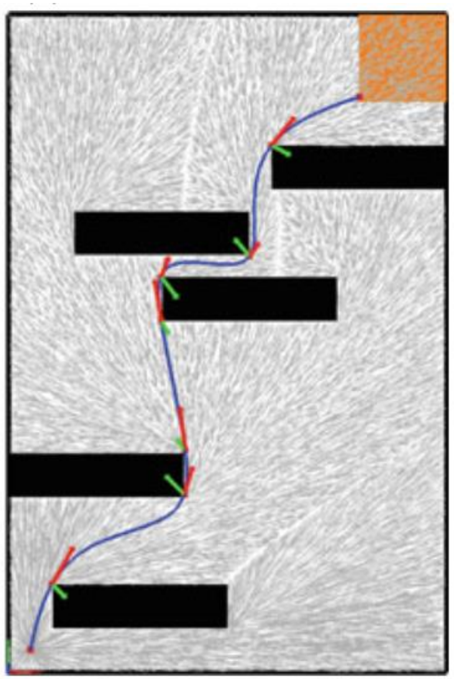
- 最终轨迹



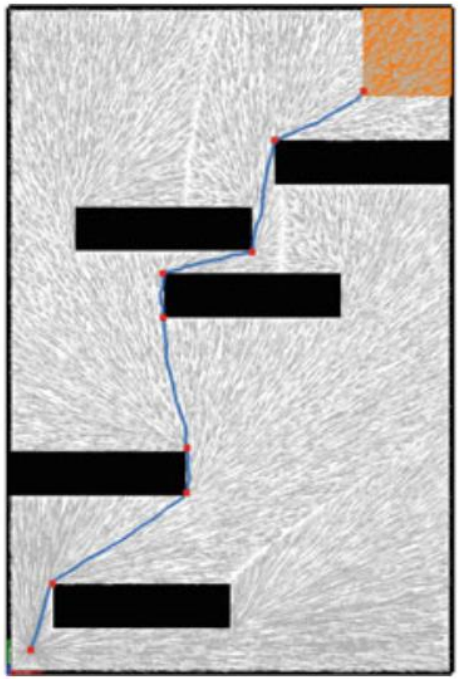
问：怎样获得航点？

层次性规划

- 路径规划+ 轨迹生成
  - 低复杂度：路径规划的问题维度低



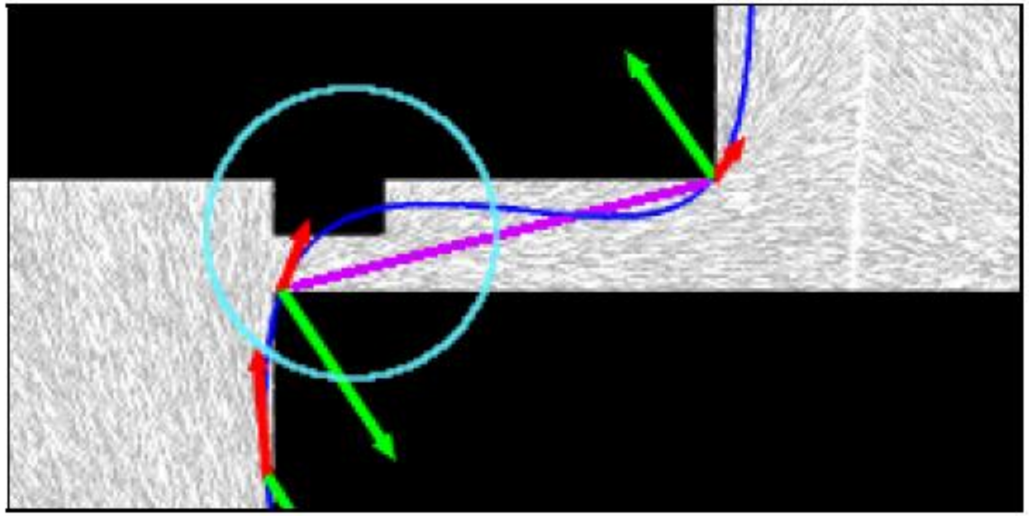
给定航点的轨迹生成具有闭式解

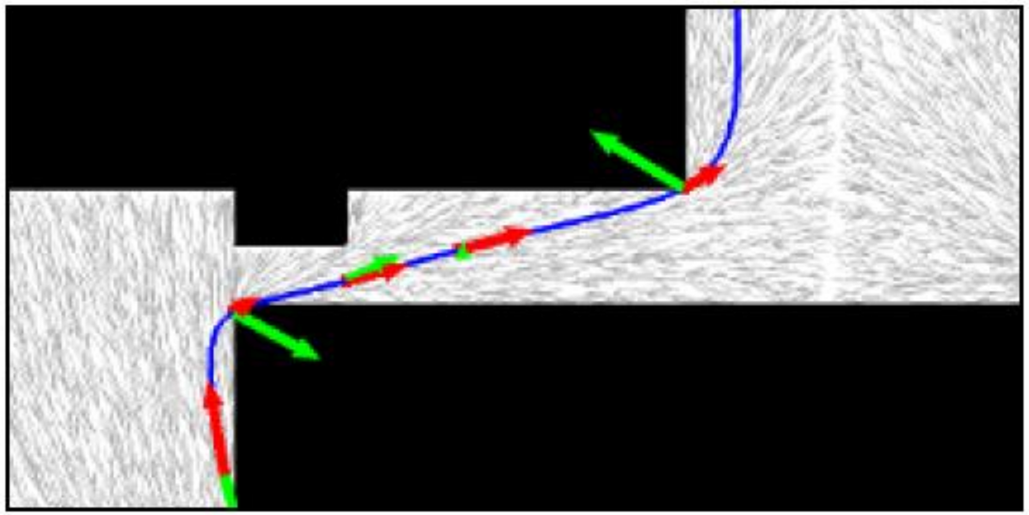


如何获取无碰撞的航点：路径规划

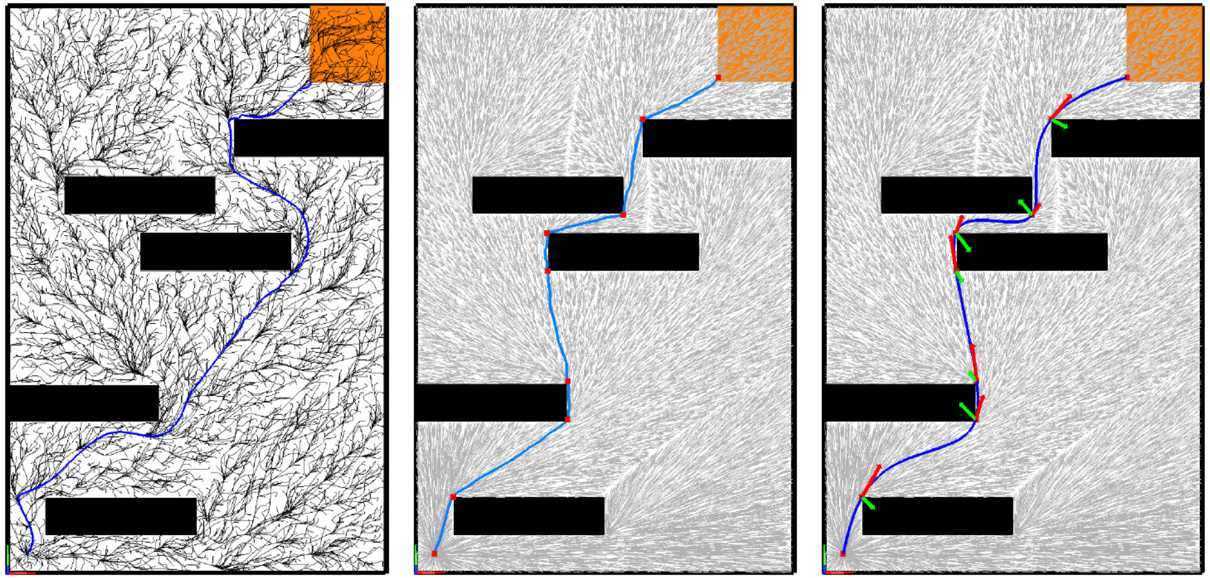
基于闭式解的层次化规划

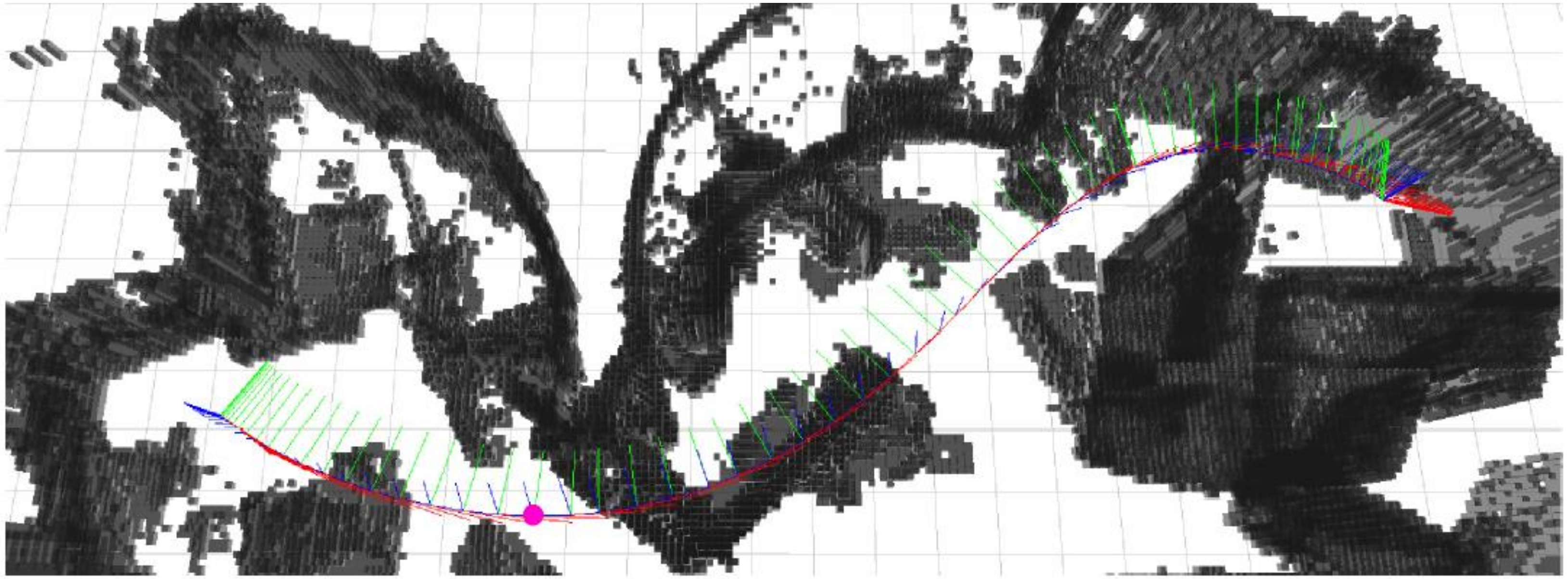
- 初始路径无碰撞
- 在初始路径的航点上利用闭式解生成轨迹
- 轨迹的碰撞部分则迭代地加入无碰撞的中间航点



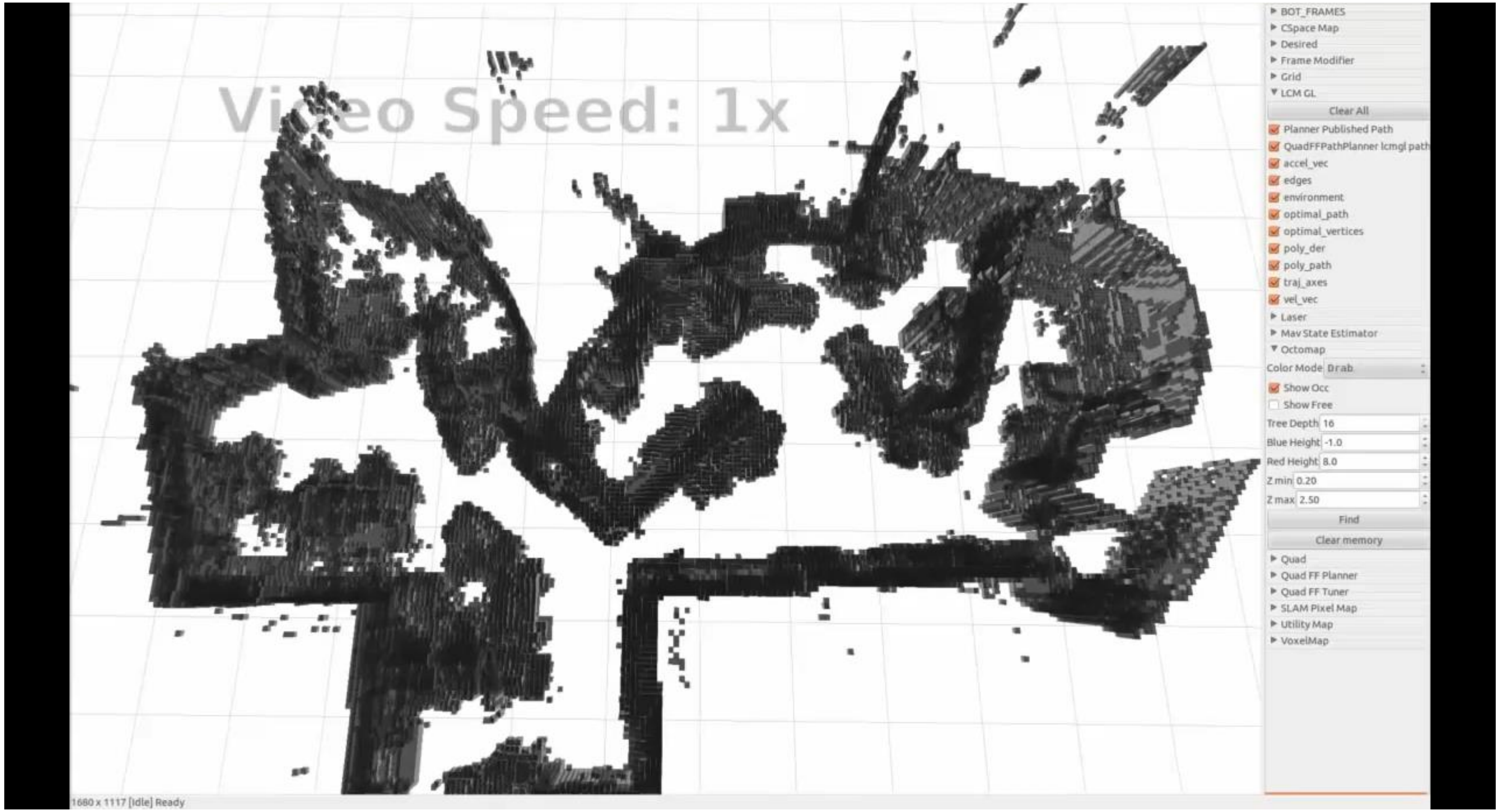


RRT*+最小化Snap轨迹生成





实验验证


Speed: 1x
BOT_FRAMES
CSpace Map
Desired
Frame Modifier
Grid
LCM GL
Clear All
Planner Published Path
QuadFFPathPlanner lcmgl path
accel_vec
edges
environment
optimal_path
optimal_vertices
poly_der
poly_path
traj_axes
vel_vec
Laser
Mav State Estimator
Octomap
Color Mode Drab
Show Occ
Show Free
Tree Depth 16
Blue Height -1.0
Red Height 8.0
Z min 0.20
Z max 2.50
Find
Clear memory
Quad
Quad FF Planner
Quad FF Tuner
SLAM Pixel Map
Utility Map
VoxelMap
1680 x 1117 [Idle] Ready

拓展

归一化

- 时间归一化
  - 很小的时间间隔可能破坏整段轨迹
  - 将很小的时间区间改成合理的数字(1.0).
  - 或者向所有曲线增加比例因子.

$$f(t)=\begin{cases}\sum_{i=0}^{N} p_{1,i}\left(\frac{t-T_0}{T_1-T_0}\right)^i & T_0 \le t \le T_1 \\ \qquad\vdots & \qquad\vdots \\ \sum_{i=0}^{N} p_{2,i}\left(\frac{t-T_1}{T_2-T_1}\right)^i & T_1 \le t \le T_2 \\ \qquad\vdots & \qquad\vdots \\ \sum_{i=0}^{N} p_{M,i}\left(\frac{t-T_{M-1}}{T_M-T_{M-1}}\right)^i & T_{M-1} \le t \le T_M \end{cases}$$

使用相对时间!

- 问题尺度（空间） 归一化
  - 如果问题在大尺度场景下提出.
  - 例如航点的 $x = 100.0\,m$
  - 考虑解决一个小尺度问题（沙盒），然后将结果变换回来

在实践中这两步操作大大提高了数值稳定性

其他工程注意事项

1. 独立还是一起求解3个坐标轴?

$$\min \quad \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}^T \begin{bmatrix} \mathbf{Q}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Q}_M \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}$$

$$\text{s.t.} \quad \mathbf{A}_{eq} \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix} = \mathbf{d}_{eq}$$

- 通常，求解3个小尺度的问题比1个大尺度的问题更好（稳定，快捷）
- 耦合的轨迹生成可能会在3个坐标轴中产生不同的权重

2. 闭式求解一直更优?

- 当矩阵操作耗费资源，数值求解更鲁棒
- 通用求解器(如Mosek)具有良好的稳定性

Efficiency comparison

Benchmark Problem: 3-Segment Joint Optimization	
Method	Solution Time (ms)
MATLAB `quadprog.m`	9.5
MATLAB Constrained	1.7
MATLAB Unconstrained (Dense)	2.7
C++/Eigen Constrained	0.18
C++/Eigen Unconstrained (Dense)	0.34

3. 多项式可以做任何事?

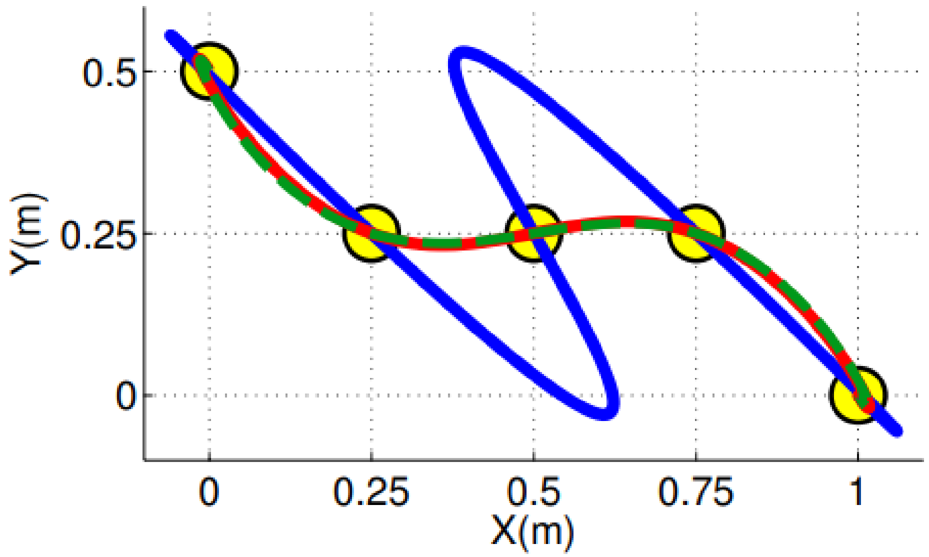
- 几乎是，但不是所有
- 可以证明多项式方程可以求得最小化单项平方控制输入的最优解。
- 但是如果是

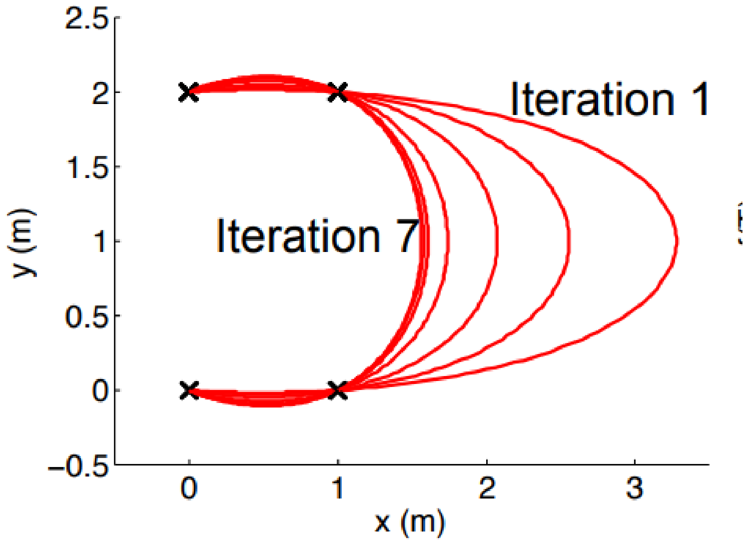
$$J = \int_0^T \rho_1 \cdot jerk^2(t) + \rho_1 \cdot snap^2(t)dt.$$

多项式将不再是最优解（超出课程范围）

时间分配

- 分段轨迹需要分段时间分配
- 时间分配显著影响最终轨迹
- 如何获得合理的时间分配？





朴素的解决方法

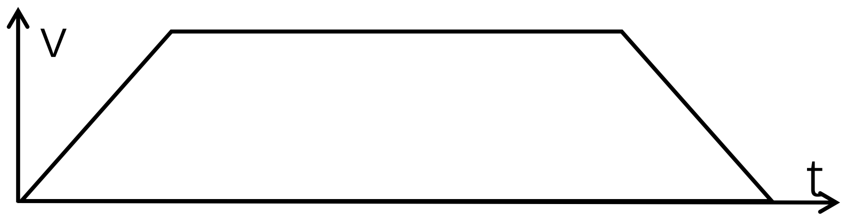
- 使用“梯形速度”时间轮廓来获得每段的时间区间
  - 假设每段轨迹，加速到最大速度->减速到0
  - 加速 + 最大速度 + 减速

- 使用期待的平均速度来获得每段轨迹的时间

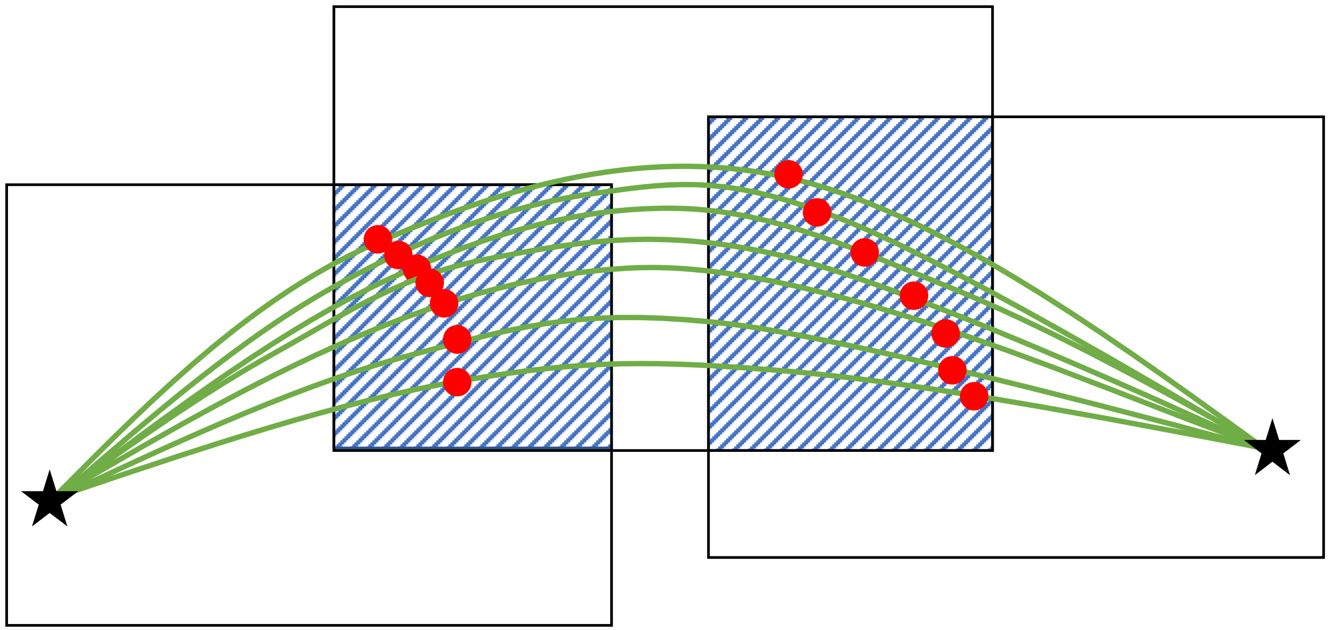
- 远未达到最优
- 只能获得保守的时间轮廓
- 对不同的环境情况没有响应

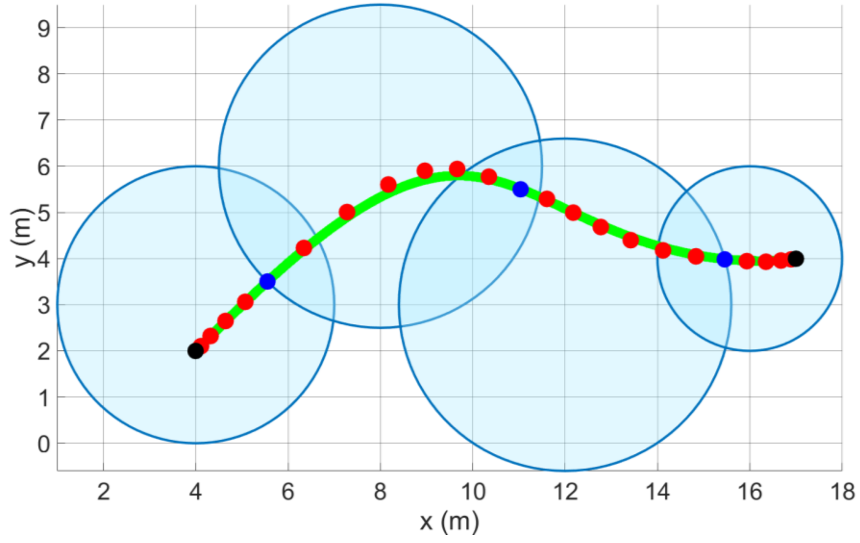
Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments, J. Chen, **ICRA** 2016
Fast, dynamic trajectory planning for a dynamically stable mobile robot, M. Shomin, **IROS** 2014

朴素的解决方法



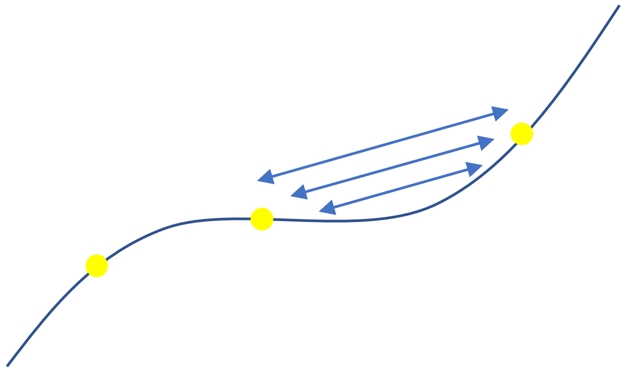
- 作用于一段轨迹非常粗糙
- 在基于飞行走廊的轨迹生成取得不错的效果
- 飞行走廊的重叠区域提供了大片自动调节的空间





直观的解决方法

- 时间区间后处理
  - 假设相对激进的时间分配
  - 生成轨迹
  - 检查每段轨迹的可行性



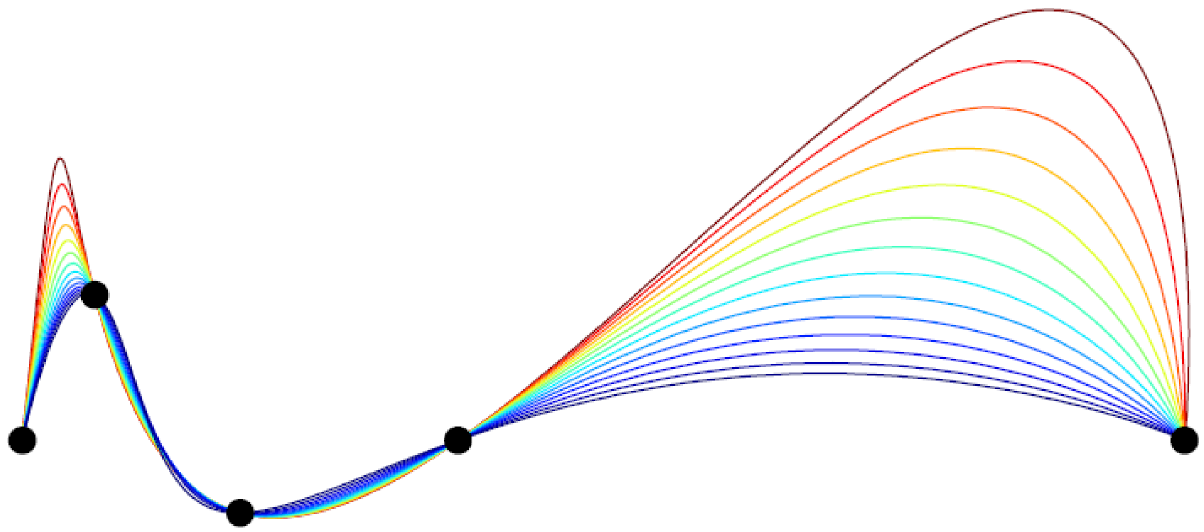
- 增大 T 直到所有可行性条件满足.
- 可能无解

迭代数值求解

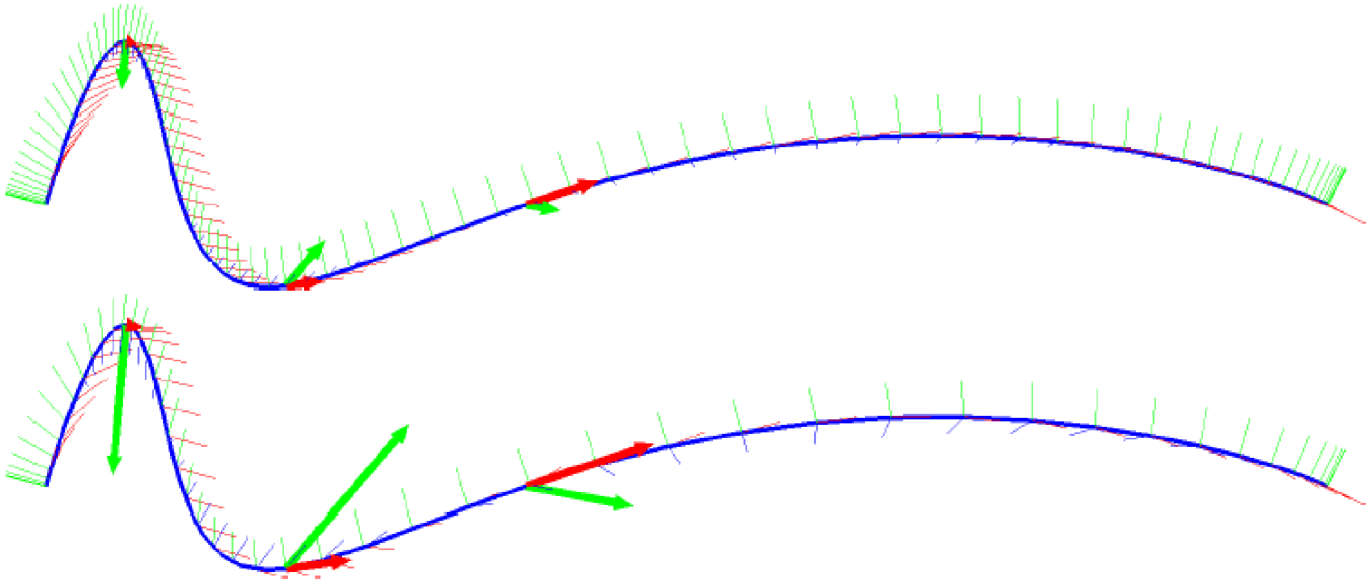
$$\boxed{J_T} = \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}^T \begin{bmatrix} Q_1(T_1) & & \\ & \ddots & \\ & & Q_M(T_M) \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_M \end{bmatrix}^T + \boxed{k_T} \sum_{i=1}^{M} T_i$$

在总代价函数中惩罚时间区间

- 最小化目标函数 **J**
- 数值方法获得 T 的梯度



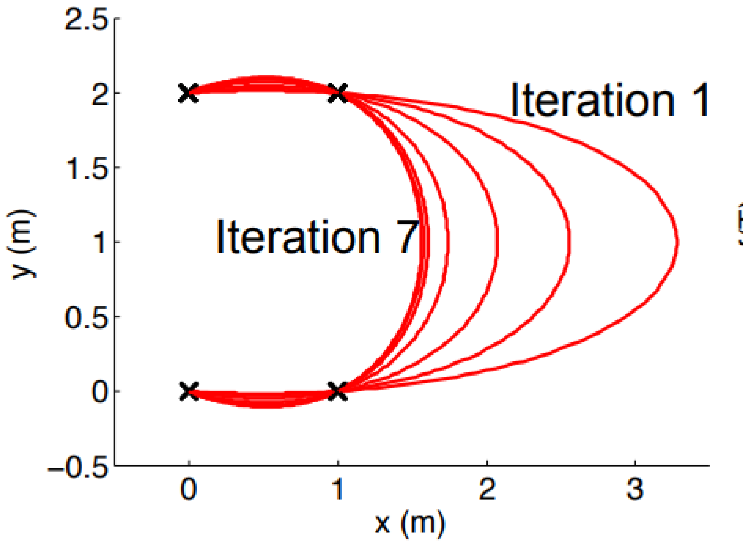
迭代数值求解



- 可调分段时间比率
- 可调总时间

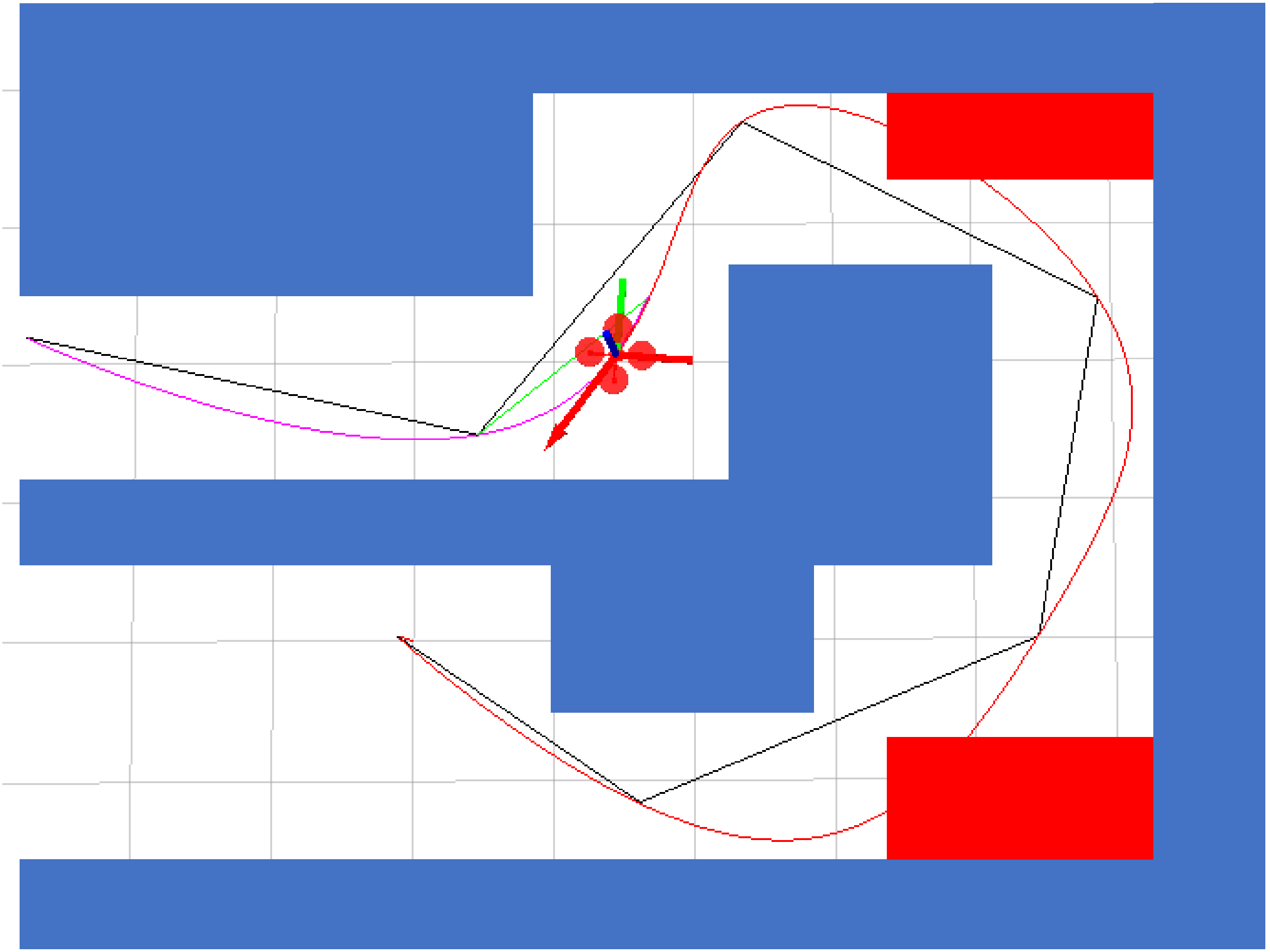
动态约束？

- 无约束，找到最佳时间分配，固定比率
- 保留最佳比率调整总轨迹时间
- 直到约束得到满足

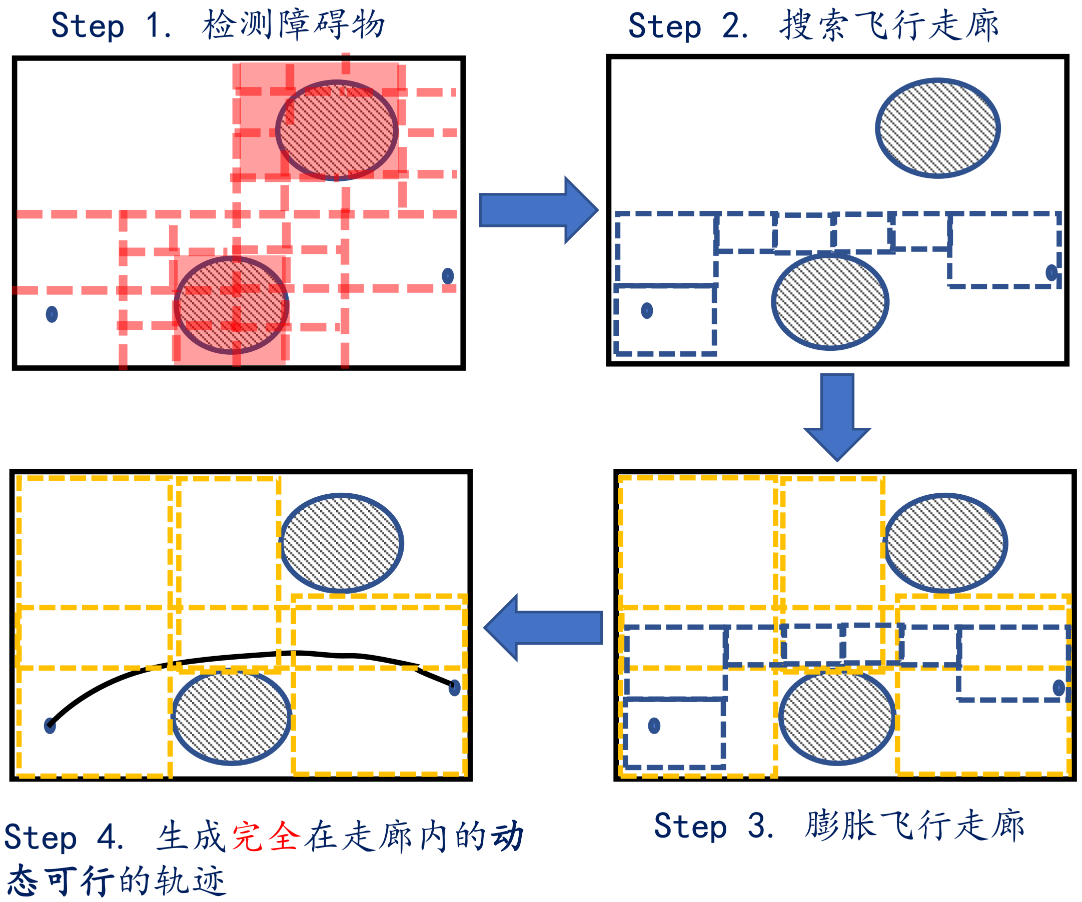


- 可调分段时间比率
- 固定总时间

更好的解决办法?

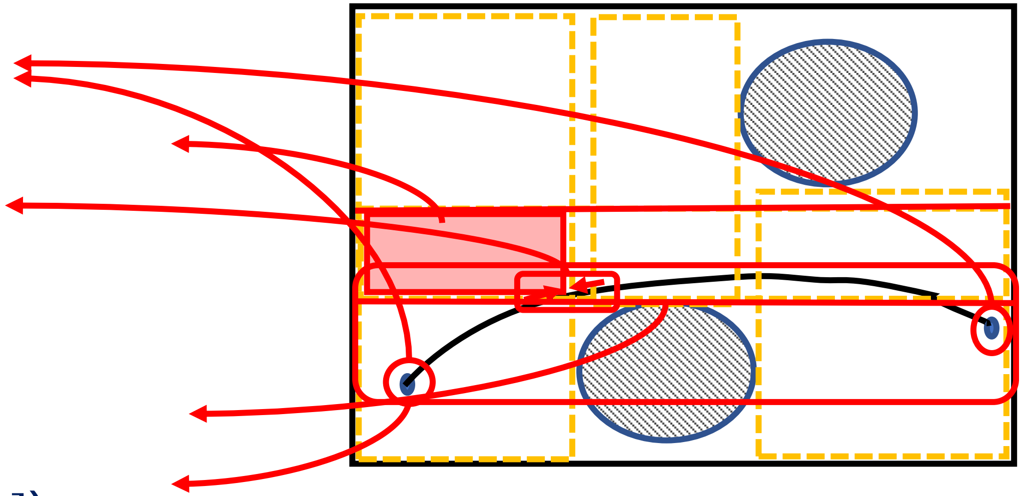


确保避障下生成光滑轨迹



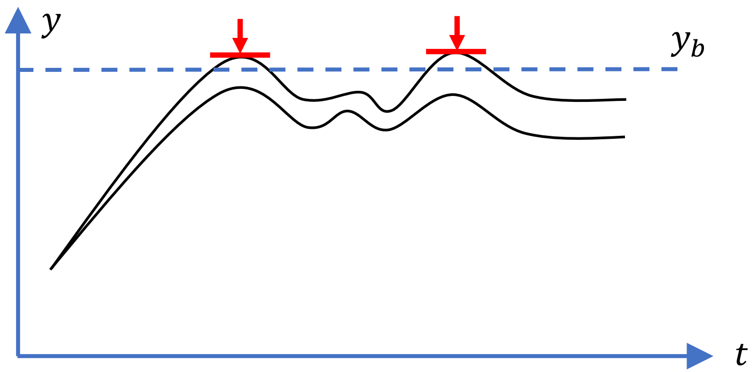
确保避障下生成光滑轨迹：构造约束

- 直接线性约束：
  - 起始位置状态约束 ($\mathbf{Ap} = \mathbf{b}$)
  - 中间点约束 ($\mathbf{Ap} = \mathbf{b}, \mathbf{Ap} \leq \mathbf{b}$)
  - 连续性约束($\mathbf{Ap_i} = \mathbf{Ap_{i+1}}$)
- 中间线性约束：
  - 边界条件 ($\mathbf{A}(t)\mathbf{p} \leq \mathbf{b}, \forall t \in [t_l, t_r]$)
  - 动态约束 ($\mathbf{A}(t)\mathbf{p} \leq \mathbf{b}, \forall t \in [t_l, t_r]$)
    - 速度约束
    - 加速度约束



约束在全局条件下满足

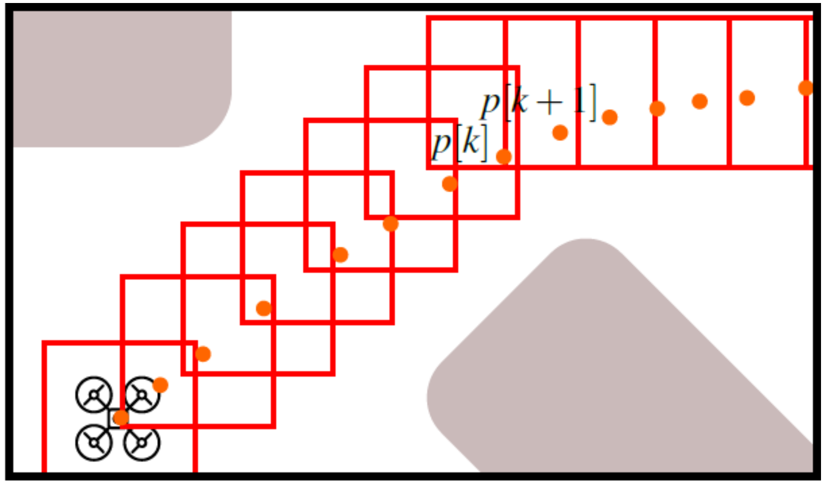
- 迭代检查机制，并增加额外约束



Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments, J. Chen, **ICRA** 2016

- 迭代求解耗费时间
- 如果没有严格满足所有约束的解。必须迭代10次检查解的存在？

- 在离散时间点上增加数值约束



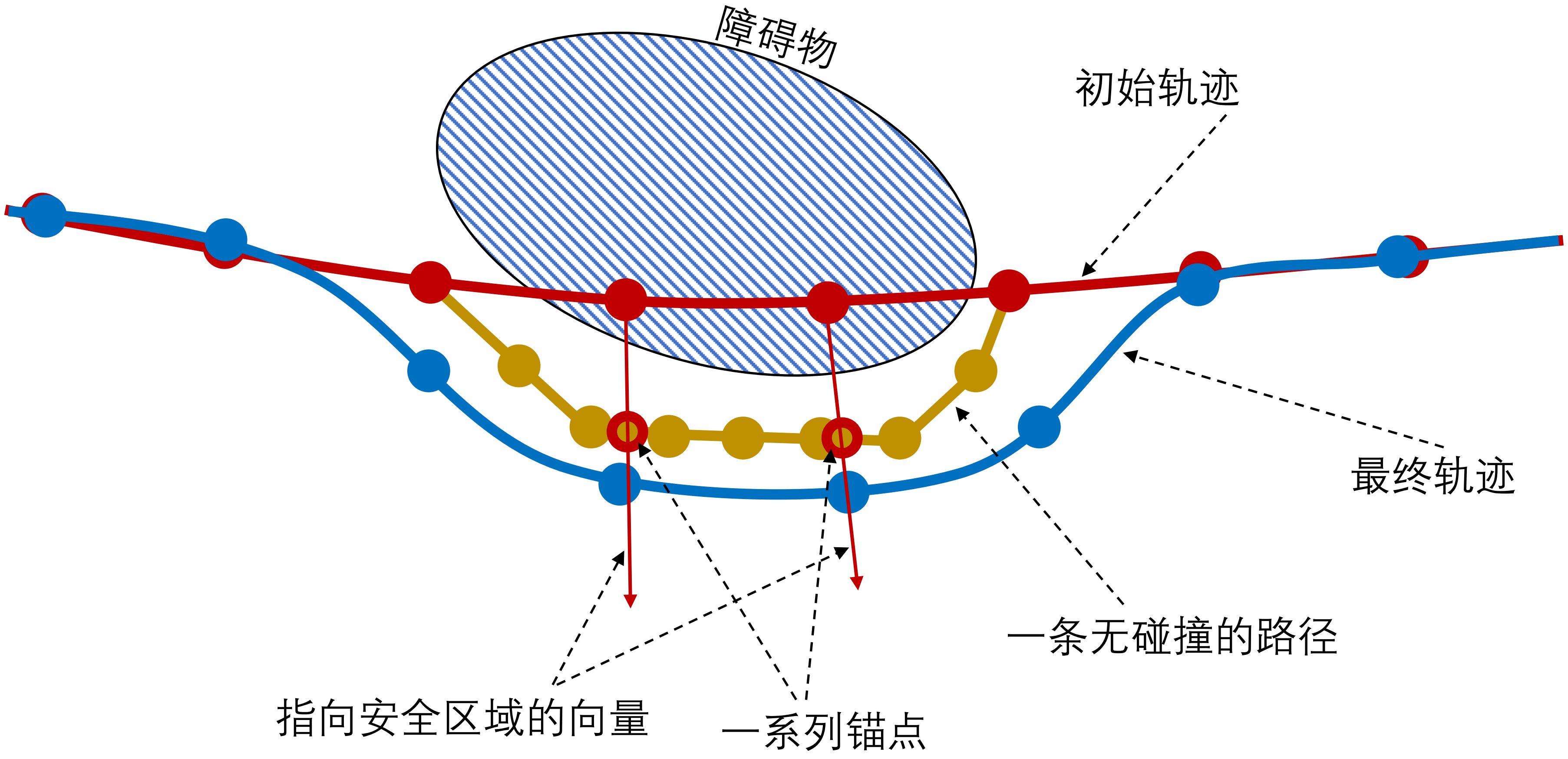
A hybrid method for online trajectory planning of mobile robots in cluttered environments, L Campos-Macías, **RAL** 2017

- 产生过于保守的轨迹
- 太多约束，计算耗费太大

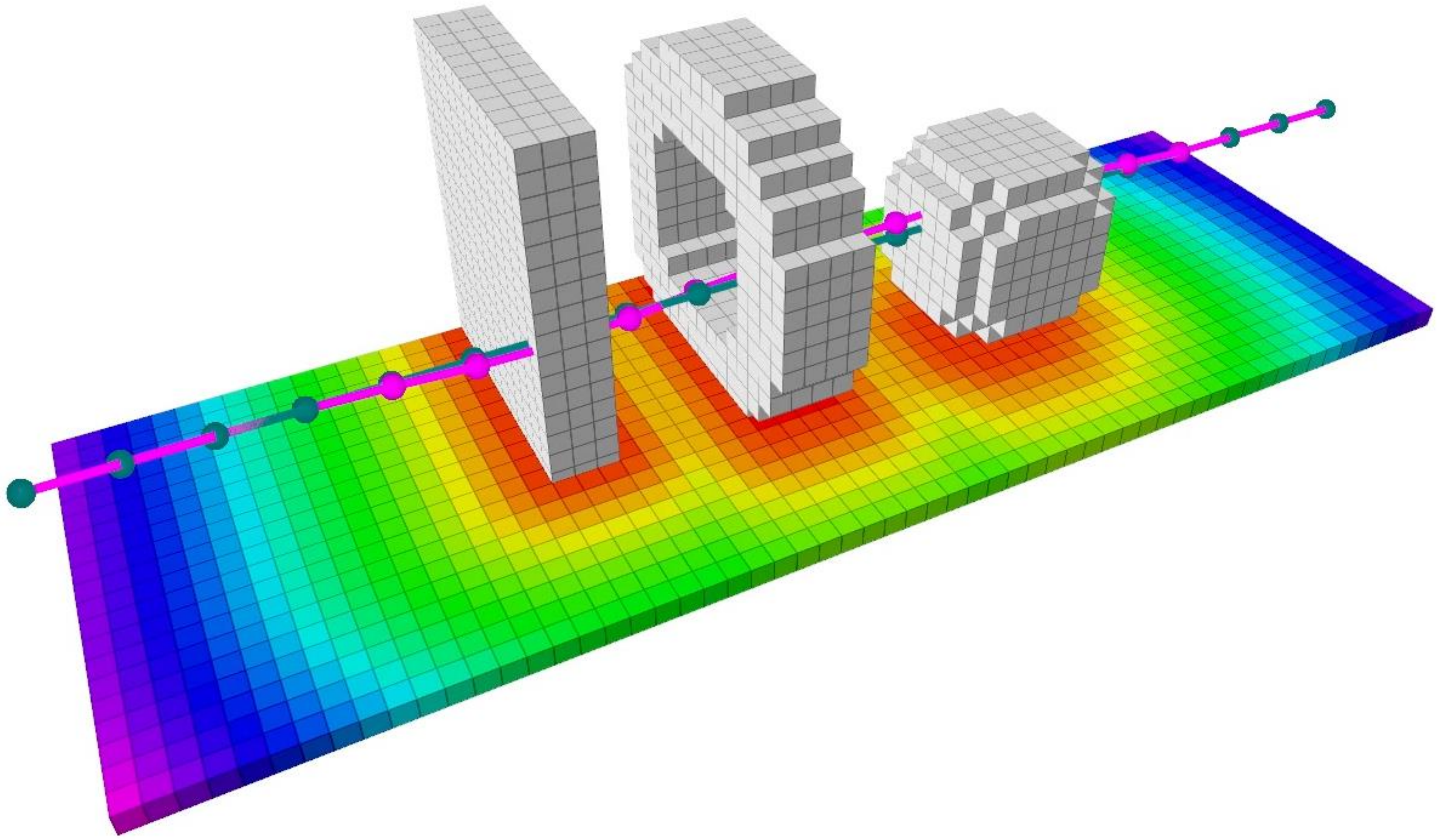
解决方案1：迭代约束

解决方案2：保守约束

解决方案3：软约束



解决方案3：软约束



解决方案3：软约束


EGO-Planner: An ESDF-free Gradient-based
Local Planner for Quadrotors
Xin Zhou, Zhepei Wang, Chao Xu and Fei Gao
CSE
ZJU FAST Lab

谢谢观看