Motion planning homework 4

Student name: Francisrk

Due date: February 13th, 2024

1 第 1 题

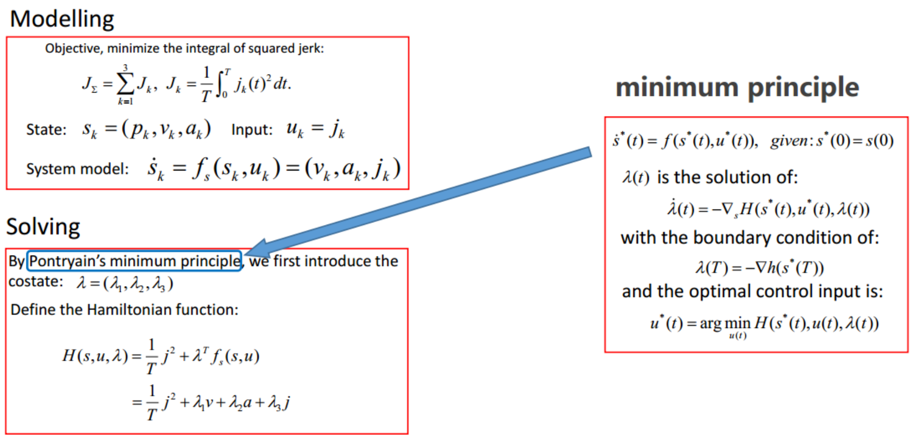
1.1 题目

- For the OBVP problem stated in slides p.25-p.29, please get the optimal solution (control, state, and time) for partially free final state case.

- Suppose the position is fixed, velocity and acceleration are free here.

1.2 求解

首先回顾课程中 OBVP 如何求解的:



Modelling

Objective, minimize the integral of squared jerk:

$$J_\Sigma = \sum_{k=1}^{3} J_k,\ J_k = \frac{1}{T}\int_0^T j_k(t)^2 dt.$$

State: $s_k = (p_k, v_k, a_k)$ Input: $u_k = j_k$

System model: $\dot{s}_k = f_s(s_k, u_k) = (v_k, a_k, j_k)$

Solving

By Pontryain's minimum principle, we first introduce the costate: $\lambda = (\lambda_1, \lambda_2, \lambda_3)$

Define the Hamiltonian function:

$$\begin{aligned} H(s,u,\lambda) &= \frac{1}{T} j^2 + \lambda^T f_s(s,u) \\ &= \frac{1}{T} j^2 + \lambda_1 v + \lambda_2 a + \lambda_3 j \end{aligned}$$

minimum principle

$\dot{s}^*(t) = f(s^*(t), u^*(t))$, $given: s^*(0) = s(0)$

$\lambda(t)$ is the solution of:

$$\dot{\lambda}(t) = -\nabla_s H(s^*(t), u^*(t), \lambda(t))$$

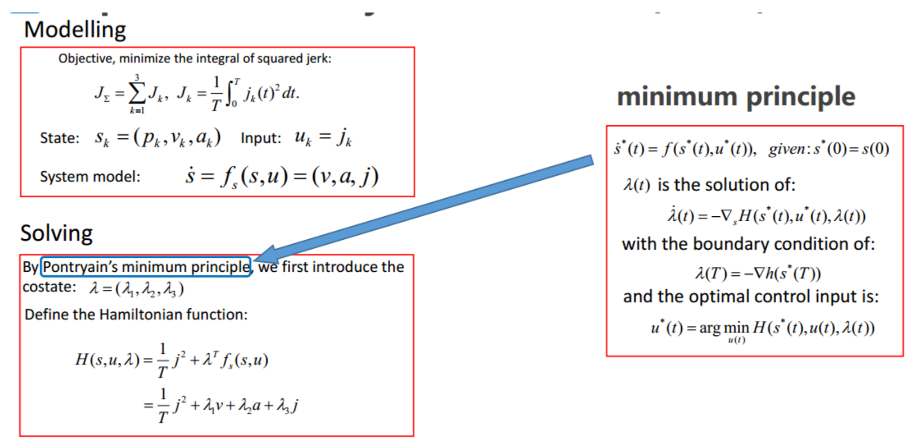
with the boundary condition of:

$$\lambda(T) = -\nabla h(s^*(T))$$

and the optimal control input is:

$$u^*(t) = \arg\min_{u(t)} H(s^*(t), u(t), \lambda(t))$$

图 1.1 四旋翼无人机建模


Modelling

Objective, minimize the integral of squared jerk:

$$J_\Sigma = \sum_{k=1}^{3} J_k,\quad J_k = \frac{1}{T}\int_0^T j_k(t)^2\,dt.$$

State: $s_k = (p_k, v_k, a_k)$ Input: $u_k = j_k$

System model: $\dot{s} = f_s(s,u) = (v, a, j)$

Solving

By Pontryain's minimum principle, we first introduce the costate: $\lambda = (\lambda_1, \lambda_2, \lambda_3)$

Define the Hamiltonian function:

$$\begin{aligned} H(s,u,\lambda) &= \frac{1}{T} j^2 + \lambda^T f_s(s,u) \\ &= \frac{1}{T} j^2 + \lambda_1 v + \lambda_2 a + \lambda_3 j \end{aligned}$$

minimum principle

$\dot{s}^*(t) = f(s^*(t), u^*(t))$, $given: s^*(0) = s(0)$

$\lambda(t)$ is the solution of:

$$\dot{\lambda}(t) = -\nabla_s H(s^*(t), u^*(t), \lambda(t))$$

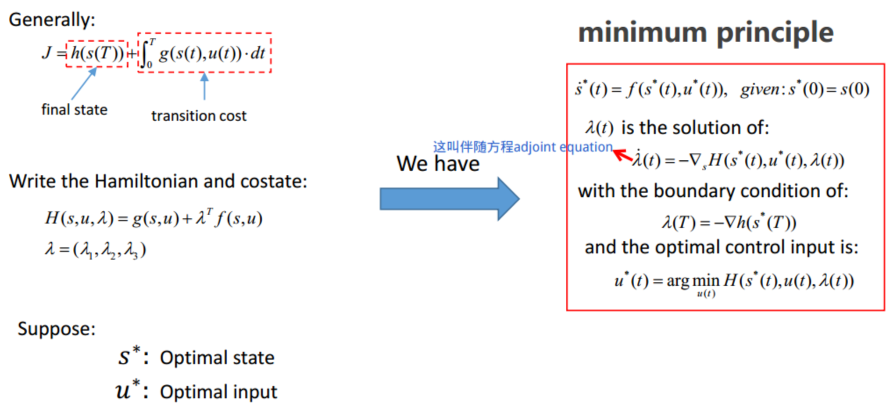
with the boundary condition of:

$$\lambda(T) = -\nabla h(s^*(T))$$

and the optimal control input is:

$$u^*(t) = \arg\min_{u(t)} H(s^*(t), u(t), \lambda(t))$$

图 1.2 对每个轴单独讨论，去掉 k


Generally:

$$J = \boxed{h(s(T))} + \boxed{\int_0^T g(s(t),u(t))\cdot dt}$$

final state
transition cost

Write the Hamiltonian and costate:

$$H(s,u,\lambda) = g(s,u) + \lambda^T f(s,u)$$
$$\lambda = (\lambda_1, \lambda_2, \lambda_3)$$

Suppose:

s^*: Optimal state

u^*: Optimal input

We have

minimum principle

$\dot{s}^*(t) = f(s^*(t), u^*(t))$, $given: s^*(0) = s(0)$

$\lambda(t)$ is the solution of:

这叫伴随方程adjoint equation

$$\dot{\lambda}(t) = -\nabla_s H(s^*(t), u^*(t), \lambda(t))$$

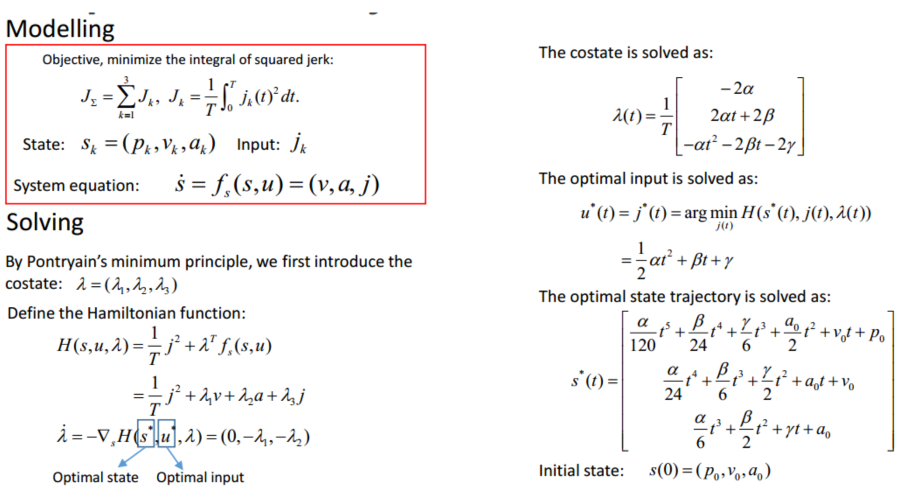
with the boundary condition of:

$$\lambda(T) = -\nabla h(s^*(T))$$

and the optimal control input is:

$$u^*(t) = \arg\min_{u(t)} H(s^*(t), u(t), \lambda(t))$$

图 1.3 引入 Pontryain 极小值原理，协态变量，Hamiltonian 方程



Modelling

Objective, minimize the integral of squared jerk:

$$J_\Sigma = \sum_{k=1}^{3} J_k,\ J_k = \frac{1}{T}\int_0^T j_k(t)^2 dt.$$

State: $s_k = (p_k, v_k, a_k)$ Input: j_k

System equation: $\dot{s} = f_s(s,u) = (v, a, j)$

Solving

By Pontryain's minimum principle, we first introduce the costate: $\lambda = (\lambda_1, \lambda_2, \lambda_3)$

Define the Hamiltonian function:

$$H(s,u,\lambda) = \frac{1}{T} j^2 + \lambda^T f_s(s,u)$$

$$= \frac{1}{T} j^2 + \lambda_1 v + \lambda_2 a + \lambda_3 j$$

$$\dot{\lambda} = -\nabla_s H(\boxed{s^*}, \boxed{u^*}, \lambda) = (0, -\lambda_1, -\lambda_2)$$

Optimal state Optimal input

The costate is solved as:

$$\lambda(t) = \frac{1}{T}\begin{bmatrix} -2\alpha \\ 2\alpha t + 2\beta \\ -\alpha t^2 - 2\beta t - 2\gamma \end{bmatrix}$$

The optimal input is solved as:

$$u^*(t) = j^*(t) = \arg\min_{j(t)} H(s^*(t), j(t), \lambda(t))$$

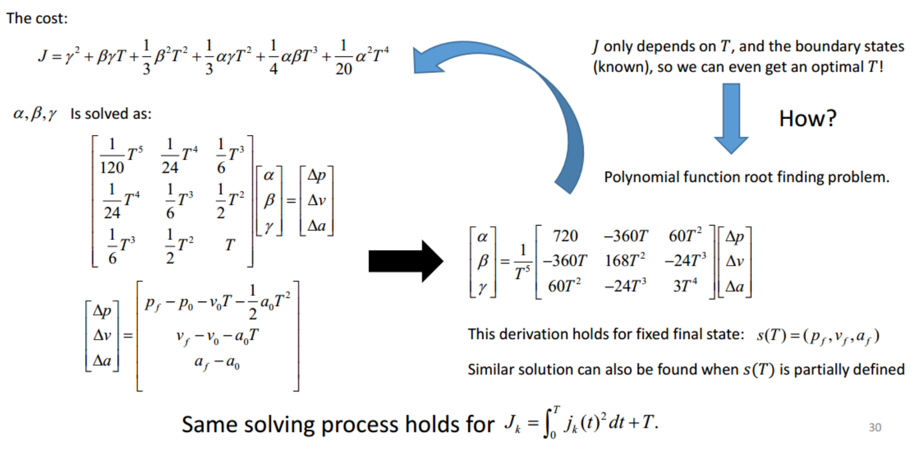
$$= \frac{1}{2}\alpha t^2 + \beta t + \gamma$$

The optimal state trajectory is solved as:

$$s^*(t) = \begin{bmatrix} \frac{\alpha}{120} t^5 + \frac{\beta}{24} t^4 + \frac{\gamma}{6} t^3 + \frac{a_0}{2} t^2 + v_0 t + p_0 \\ \frac{\alpha}{24} t^4 + \frac{\beta}{6} t^3 + \frac{\gamma}{2} t^2 + a_0 t + v_0 \\ \frac{\alpha}{6} t^3 + \frac{\beta}{2} t^2 + \gamma t + a_0 \end{bmatrix}$$

Initial state: $s(0) = (p_0, v_0, a_0)$

图 1.4 求解伴随方程、最优输入、最优状态



The cost:

$$J = \gamma^2 + \beta\gamma T + \frac{1}{3}\beta^2 T^2 + \frac{1}{3}\alpha\gamma T^2 + \frac{1}{4}\alpha\beta T^3 + \frac{1}{20}\alpha^2 T^4$$

J only depends on T, and the boundary states (known), so we can even get an optimal T!

How?

Polynomial function root finding problem.

α, β, γ Is solved as:

$$\begin{bmatrix} \frac{1}{120}T^5 & \frac{1}{24}T^4 & \frac{1}{6}T^3 \\ \frac{1}{24}T^4 & \frac{1}{6}T^3 & \frac{1}{2}T^2 \\ \frac{1}{6}T^3 & \frac{1}{2}T^2 & T \end{bmatrix}\begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} \Delta p \\ \Delta v \\ \Delta a \end{bmatrix}$$

$$\begin{bmatrix} \Delta p \\ \Delta v \\ \Delta a \end{bmatrix} = \begin{bmatrix} p_f - p_0 - v_0 T - \frac{1}{2}a_0 T^2 \\ v_f - v_0 - a_0 T \\ a_f - a_0 \end{bmatrix}$$

$$\begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \frac{1}{T^5}\begin{bmatrix} 720 & -360T & 60T^2 \\ -360T & 168T^2 & -24T^3 \\ 60T^2 & -24T^3 & 3T^4 \end{bmatrix}\begin{bmatrix} \Delta p \\ \Delta v \\ \Delta a \end{bmatrix}$$

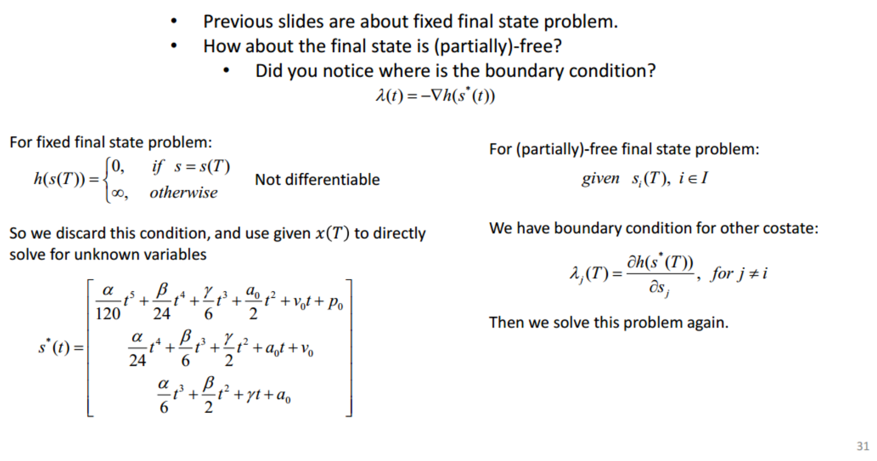
This derivation holds for fixed final state: $s(T) = (p_f, v_f, a_f)$

Similar solution can also be found when $s(T)$ is partially defined

Same solving process holds for $J_k = \int_0^T j_k(t)^2 dt + T.$

30

图 1.5 根据 final state 求出协态表达式



- Previous slides are about fixed final state problem.
- How about the final state is (partially)-free?
  - Did you notice where is the boundary condition?

$$\lambda(t) = -\nabla h(s^*(t))$$

For fixed final state problem:

$$h(s(T)) = \begin{cases} 0, & if \ \ s = s(T) \\ \infty, & otherwise \end{cases}$$ Not differentiable

So we discard this condition, and use given $x(T)$ to directly solve for unknown variables

$$s^*(t) = \begin{bmatrix} \frac{\alpha}{120}t^5 + \frac{\beta}{24}t^4 + \frac{\gamma}{6}t^3 + \frac{a_0}{2}t^2 + v_0 t + p_0 \\ \frac{\alpha}{24}t^4 + \frac{\beta}{6}t^3 + \frac{\gamma}{2}t^2 + a_0 t + v_0 \\ \frac{\alpha}{6}t^3 + \frac{\beta}{2}t^2 + \gamma t + a_0 \end{bmatrix}$$

For (partially)-free final state problem:

$$given \ \ s_i(T), \ i \in I$$

We have boundary condition for other costate:

$$\lambda_j(T) = \frac{\partial h(s^*(T))}{\partial s_j}, \ \ for \ j \neq i$$

Then we solve this problem again.

31

图 1.6 讨论 final state totally & partially fixed 的情况

题目要求最终状态 $s^*(T) = (p_f, ?, ?)$，即只 fix position，求解此时的 OBVP 问题，即求解 $u^*(t), s^*(t)$ 表达式。

为了完整地了解此问题的背景，需要参考该工作的文章 [1]。

对改文章的前 3 部分进行简单总结：本文提出一个用于生成四旋翼运动轨迹生成的一个框架，该框架特点是计算快，易实现。由两个 stage 组成：

1. trajectory generation

2. feasibilility check

同时给出了能够保证可行轨迹存在的条件。并在实际的机器上进行了测试。

stage1 时无需考虑可行性，所以可快速生成，stage2 时由于只需对多项式进行 check，所以 check 速度也很快。

1.Introduction 部分

现阶段（2015 年）对于四旋翼轨迹生成的研究有两派：

1. 将几何规划和时域规划解耦。第一步不考虑可行性，进行几何轨迹生成，第二步在时序上进行几何轨迹生成，以保证 feasibility。

2. 使用四旋翼动力学的微分平坦性推导出轨迹的约束，并且求解一系列轨迹的优化问题。

本文的方法显然是后者。

现有方法的问题：

1. 末状态的刚性约束 (必须 totally fixed)。

2. 末状态可能是时间依赖的（可能流派 1 的直接解耦不合适）

3. 流派 2 依赖于凸优化，为了使问题是凸的，可能会减少 feasible trajectories 的空间。

所以本文主要贡献是：不使用刚性的 final state 约束，在尽可能大的空间中生成轨迹，然后快速 check。

2. 系统动力学和问题描述部分

四旋翼被建模为一个 9-D 的 state：p, v, R，其中 R 是方向，由于 yaw 不可观，所以只有 8 自由度。系统有 3 项约束：动力学模型约束，输入约束，仿射平移约束。由于我们此处只讨论轨迹生成，且本文的算法框架在轨迹生成时不考虑后两种约束，所以对于 feasibility check 在此处不做讨论，仅考虑动力学约束下的轨迹生成问题。

3. 运动轨迹生成部分

同课程所讲相同，系统输入为 jerk。需要指出，cost function 原本为 $f^2||\omega||^2$，但经过推导，其上界为 $||j||^2$，所以被积函数为 $||j||^2$ 对四旋翼的三个轴进行解耦之后，单独讨论每个轴的优化问题，最终推导出 $u^*(t)$ 和 $s^*(t)$ 的 closed-form 形式：

$$\begin{aligned} j^*(t) &= arg\min_{j(t)} H(s^*(t), j(t), \lambda(t)) \\ &= \frac{1}{2}\alpha t^2 + \beta t + \gamma \end{aligned} \tag{1.1}$$

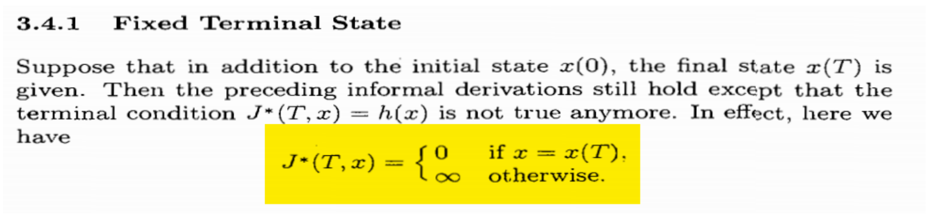
$$s^*(t) = \begin{bmatrix} \frac{\alpha}{120}t^5 + \frac{\beta}{24}t^4 + \frac{\gamma}{6}t^3 + \frac{a_0}{2}t^2 + v_0 t + p_0 \\ \frac{\alpha}{24}t^4 + \frac{\beta}{6}t^3 + \frac{\gamma}{2}t^2 + a_0 t + v_0 \\ \frac{\alpha}{6}t^3 + \frac{\beta}{2}t^2 + \gamma t + a_0 \end{bmatrix} \tag{1.2}$$

协态 costate

$$\lambda(t) = \frac{1}{T}\begin{bmatrix} -2\alpha \\ 2\alpha t + 2\beta \\ -\alpha t^2 - 2\beta t - 2\gamma \end{bmatrix} \tag{1.3}$$

其中 α, β, γ 是需要被确定的，根据 final state 的约束不同，求解的结果不同，下面将进行讨论。

1. 当 final state totally fixed 时, 通过带入 final state 即可求出。该 case 为 rigid constraints，终止 terminal condition 不再成立，[2] 中如下图所述：



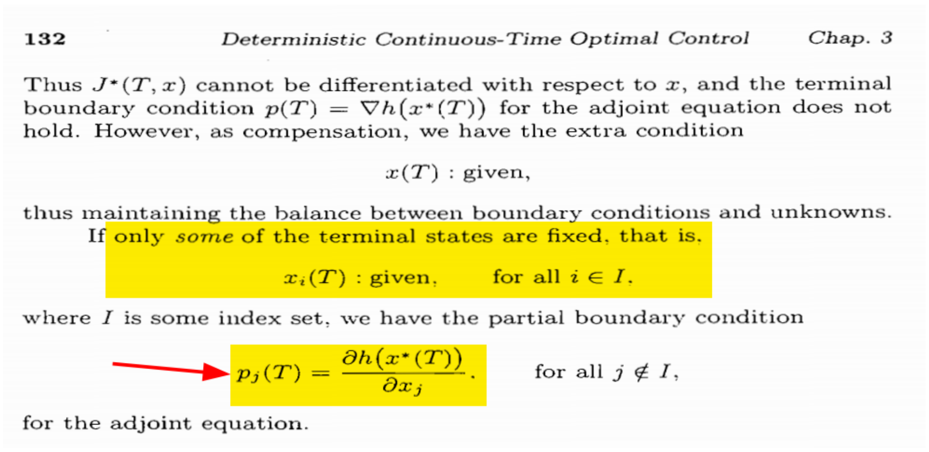
3.4.1 Fixed Terminal State

Suppose that in addition to the initial state $x(0)$, the final state $x(T)$ is given. Then the preceding informal derivations still hold except that the terminal condition $J^*(T,x) = h(x)$ is not true anymore. In effect, here we have

$$J^*(T,x) = \begin{cases} 0 & \text{if } x = x(T), \\ \infty & \text{otherwise.} \end{cases}$$

图 1.7 当 final state totally fixed 时，终止条件不成立

2. 当 final state partially fixed 时，仅使用 final state 的约束无法求解出所有的变量，需要利用 boundary condition 的约束进行求解。[1] 中作者提到 “the corresponding costates must equal zero at the end time”，结合 [2] 中的描述



132 *Deterministic Continuous-Time Optimal Control* Chap. 3

Thus $J^*(T,x)$ cannot be differentiated with respect to x, and the terminal boundary condition $p(T) = \nabla h\big(x^*(T)\big)$ for the adjoint equation does not hold. However, as compensation, we have the extra condition

$$x(T): \text{given},$$

thus maintaining the balance between boundary conditions and unknowns. If only *some* of the terminal states are fixed, that is,

$$x_i(T): \text{given}, \qquad \text{for all } i \in I,$$

where I is some index set, we have the partial boundary condition

$$p_j(T) = \frac{\partial h\big(x^*(T)\big)}{\partial x_j}, \qquad \text{for all } j \notin I,$$

for the adjoint equation.

图 1.8 final state partially fixed 时的 boundary condition

上图中 $p(t)$ 为 costate，也即课程中的 $\lambda(t)$，关键结论是

$$\lambda_j(T) = \frac{\partial h(s^*(T))}{\partial s_j}, forj \neq i \tag{1.4}$$

我们可以得出此时的边界条件，接下来将分别讨论 final state 中：

(a) only fix $\boldsymbol{p_f}$

(b) only fix $\boldsymbol{v_f}$

(c) fix $\boldsymbol{p_f}, \boldsymbol{v_f}$

三种情况时的 α, β, γ 的求解情况。

(a) only fix $\boldsymbol{p_f}$ 时

由 boundary condition

$$\begin{cases} \lambda_2(T) = 0 \\ \lambda_3(T) = 0 \end{cases} \tag{1.5}$$

将 (1.5) 带入 (1.3) 得

$$\begin{cases} 2\alpha T + 2\beta = 0 \\ -\alpha T^2 - 2\beta T - 2\gamma = 0 \end{cases} \tag{1.6}$$

整理得:

$$\begin{cases} \beta = -\alpha T \\ \gamma = \frac{1}{2}\alpha T^2 \end{cases} \tag{1.7}$$

将 (1.7) 带入 (1.2)，取第 1 行整理得

$$\frac{\alpha}{120}T^5 - \frac{\alpha}{24}T^5 + \frac{\alpha}{12}T^5 = \Delta p_f \tag{1.8}$$

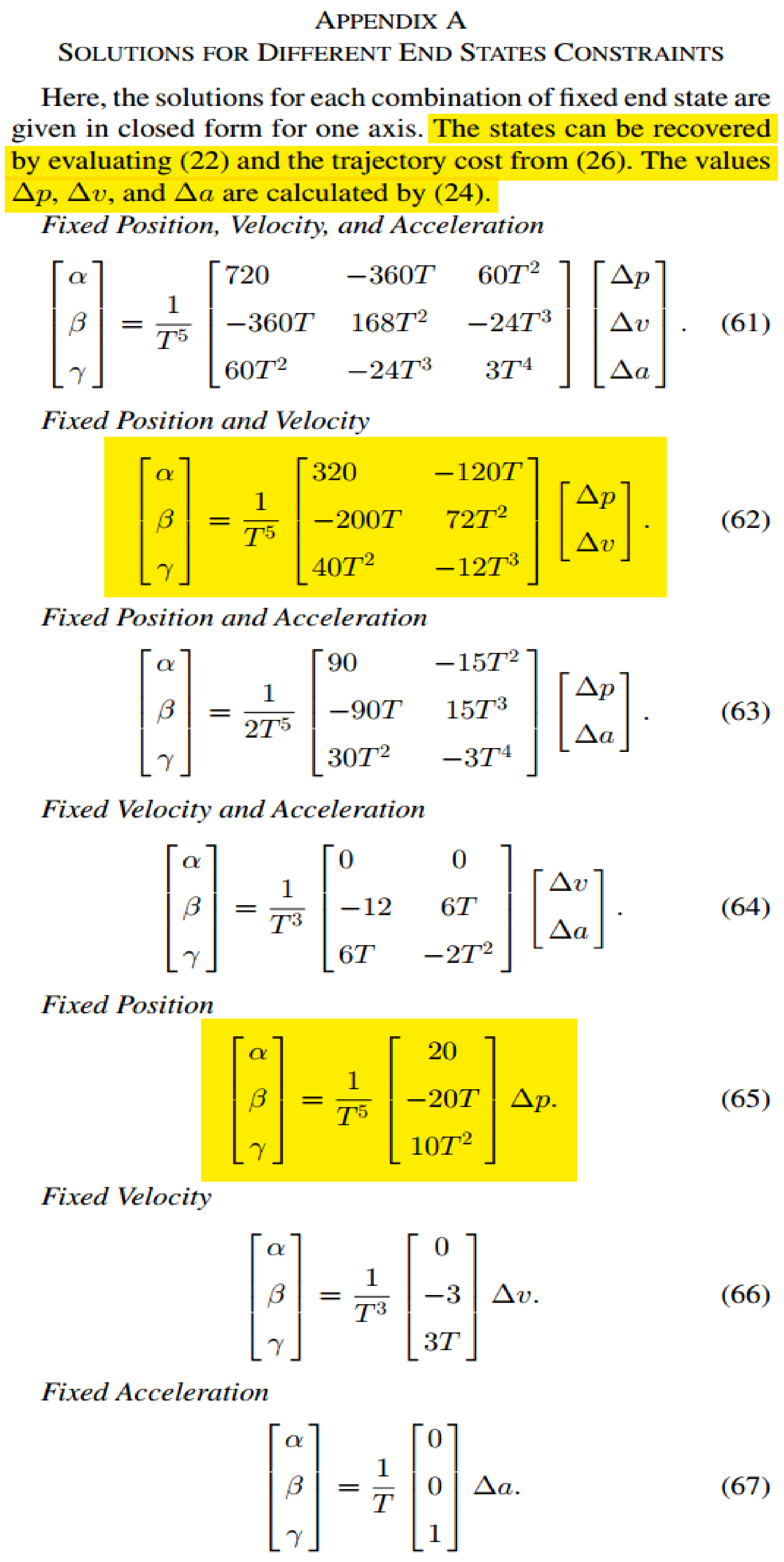
于是有

$$\alpha = \frac{20}{T^5}\Delta p_f \tag{1.9}$$

将 (1.9) 带入 (1.6)，整理得

$$\begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} \alpha \\ -\alpha T \\ \frac{1}{2}\alpha T^2 \end{bmatrix} = \begin{bmatrix} 1 \\ -T \\ \frac{1}{2}\alpha T^2 \end{bmatrix} \cdot \alpha = \begin{bmatrix} 1 \\ -T \\ \frac{1}{2}\alpha T^2 \end{bmatrix} \frac{20}{T^5}\Delta p_f = \frac{1}{T^5}\begin{bmatrix} 20 \\ -20T \\ 10T^2 \end{bmatrix} \Delta p_f \tag{1.10}$$

此处给出 [1] 附录 A 中的所有情况的结果



Appendix A
Solutions for Different End States Constraints

Here, the solutions for each combination of fixed end state are given in closed form for one axis. The states can be recovered by evaluating (22) and the trajectory cost from (26). The values Δp, Δv, and Δa are calculated by (24).

Fixed Position, Velocity, and Acceleration

$$\begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix}=\frac{1}{T^5}\begin{bmatrix}720 & -360T & 60T^2\\-360T & 168T^2 & -24T^3\\60T^2 & -24T^3 & 3T^4\end{bmatrix}\begin{bmatrix}\Delta p\\\Delta v\\\Delta a\end{bmatrix}. \tag{61}$$

Fixed Position and Velocity

$$\begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix}=\frac{1}{T^5}\begin{bmatrix}320 & -120T\\-200T & 72T^2\\40T^2 & -12T^3\end{bmatrix}\begin{bmatrix}\Delta p\\\Delta v\end{bmatrix}. \tag{62}$$

Fixed Position and Acceleration

$$\begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix}=\frac{1}{2T^5}\begin{bmatrix}90 & -15T^2\\-90T & 15T^3\\30T^2 & -3T^4\end{bmatrix}\begin{bmatrix}\Delta p\\\Delta a\end{bmatrix}. \tag{63}$$

Fixed Velocity and Acceleration

$$\begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix}=\frac{1}{T^3}\begin{bmatrix}0 & 0\\-12 & 6T\\6T & -2T^2\end{bmatrix}\begin{bmatrix}\Delta v\\\Delta a\end{bmatrix}. \tag{64}$$

Fixed Position

$$\begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix}=\frac{1}{T^5}\begin{bmatrix}20\\-20T\\10T^2\end{bmatrix}\Delta p. \tag{65}$$

Fixed Velocity

$$\begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix}=\frac{1}{T^3}\begin{bmatrix}0\\-3\\3T\end{bmatrix}\Delta v. \tag{66}$$

Fixed Acceleration

$$\begin{bmatrix}\alpha\\\beta\\\gamma\end{bmatrix}=\frac{1}{T}\begin{bmatrix}0\\0\\1\end{bmatrix}\Delta a. \tag{67}$$

图 4.9 [1]Appendix A 中当 fixed final state 不同时的 costate 结果

(b) only fix $\boldsymbol{v_f}$ 时

同理，先由 (1.4) 求得 costate 约束

$$\begin{cases} \alpha = 0 \\ \gamma = -\beta T \end{cases} \tag{1.11}$$

将 (1.11) 带入 (1.2)，取第 2 行得

$$T^3(\frac{-\beta}{3}) = \Delta v_f \tag{1.12}$$

则有

$$\beta = -\frac{3}{T^3}\Delta v_f \tag{1.13}$$

将 (1.13) 带入 (1.11)，整理得

$$\begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ -T \end{bmatrix} \beta = \begin{bmatrix} 0 \\ 1 \\ -T \end{bmatrix} (-\frac{3}{T^3})\Delta v_f = \frac{1}{T^3}\begin{bmatrix} 0 \\ -3 \\ -3T \end{bmatrix} \Delta v_f \tag{1.14}$$

(c) fix $\boldsymbol{p_f}, \boldsymbol{v_f}$ 时

只有 λ_3 可以使用 boundary condition 求得约束，由 (1.4) 得 costate 约束

$$\gamma = -\frac{1}{2}(\alpha T^2 + 2\beta T) = \begin{bmatrix} -\frac{1}{2}T^2 & -T \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{1.15}$$

将 (1.15) 带入 (1.2) 整理得

$$T^5 \begin{bmatrix} -\frac{3}{40} & -\frac{1}{8T} \\ -\frac{5}{24T} & \frac{1}{3T^2} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \Delta p_f \\ \Delta v_f \end{bmatrix} \tag{1.16}$$

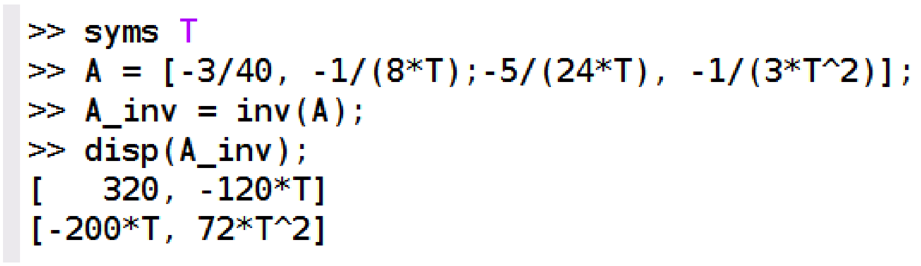
则有

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \frac{1}{T^5} \begin{bmatrix} 320 & -120T \\ -200T & 72T^2 \end{bmatrix} \begin{bmatrix} \Delta p_f \\ \Delta v_f \end{bmatrix} \tag{1.17}$$

将 (1.17) 带入 (1.15) 并与 (1.17) 联立，整理得

$$\begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \frac{1}{T^5} \begin{bmatrix} 320 & -120T \\ -200T & 72T^2 \\ 40T^2 & -12T^3 \end{bmatrix} \begin{bmatrix} \Delta p_f \\ \Delta v_f \end{bmatrix} \tag{1.18}$$

其中 (1.16) 矩阵的求逆可使用 Matlab 的符号运算完成，如图 1.10 所示



```
>> syms T
>> A = [-3/40, -1/(8*T);-5/(24*T), -1/(3*T^2)];
>> A_inv = inv(A);
>> disp(A_inv);
[   320, -120*T]
[-200*T, 72*T^2]
```

图 1.10 使用 Matlab 完成符号矩阵求逆

(1.1) 求出 $u^*(t) = j^*(t)$ 之后，带入 cost function 即可求出 cost 表达式，

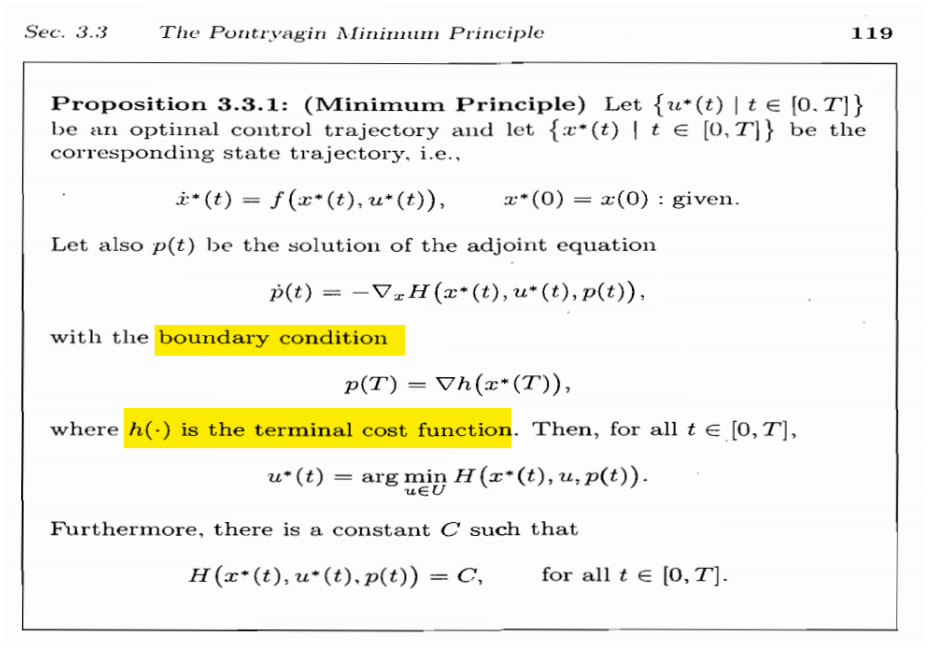
$$\begin{aligned} J &= \frac{1}{T}\int_0^T j_k(t)^2 dt \\ &= \frac{1}{T}\int_0^T (\frac{1}{2}\alpha t^2 + \beta t + \gamma) dt \\ &= \frac{1}{20}\alpha^2 T^4 + \frac{1}{4}\alpha\beta T^3 + \frac{1}{3}(\alpha\gamma + \beta^2)T^2 + \beta\gamma T + \gamma^2 \end{aligned} \tag{1.19}$$

由于前面求出的 α, β, γ 均只与 T 相关，所以 cost 也仅与 T 相关。

需要指出，由于 $u^*(t)$ 是使用 Pontryagin 极小值原理求得的通用形式，所以 “this cost holds for all combinations of end translational variables” [1]

第 1 题完成。

1.3 Pontryagin Minimum Principle 的拓展



Sec. 3.3 *The Pontryagin Minimum Principle* **119**

Proposition 3.3.1: (Minimum Principle) Let $\{u^*(t) \mid t \in [0, T]\}$ be an optimal control trajectory and let $\{x^*(t) \mid t \in [0, T]\}$ be the corresponding state trajectory, i.e.,

$$\dot{x}^*(t) = f\big(x^*(t), u^*(t)\big), \qquad x^*(0) = x(0) : \text{given}.$$

Let also $p(t)$ be the solution of the adjoint equation

$$\dot{p}(t) = -\nabla_x H\big(x^*(t), u^*(t), p(t)\big),$$

with the boundary condition

$$p(T) = \nabla h\big(x^*(T)\big),$$

where $h(\cdot)$ is the terminal cost function. Then, for all $t \in [0, T]$,

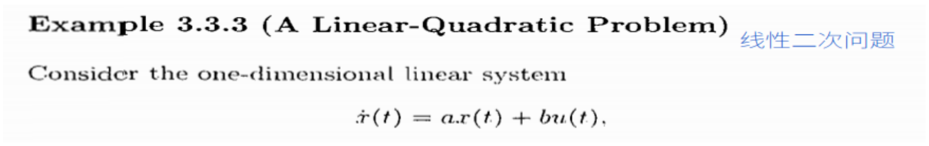
$$u^*(t) = \arg\min_{u \in U} H\big(x^*(t), u, p(t)\big).$$

Furthermore, there is a constant C such that

$$H\big(x^*(t), u^*(t), p(t)\big) = C, \qquad \text{for all } t \in [0, T].$$

图 1.11 Pontryagin Minimum Principle[1]

以 [1] 中的一个例子来强调 adjoint equation 和一阶微分方程的求解。关于一个线性二次问题，

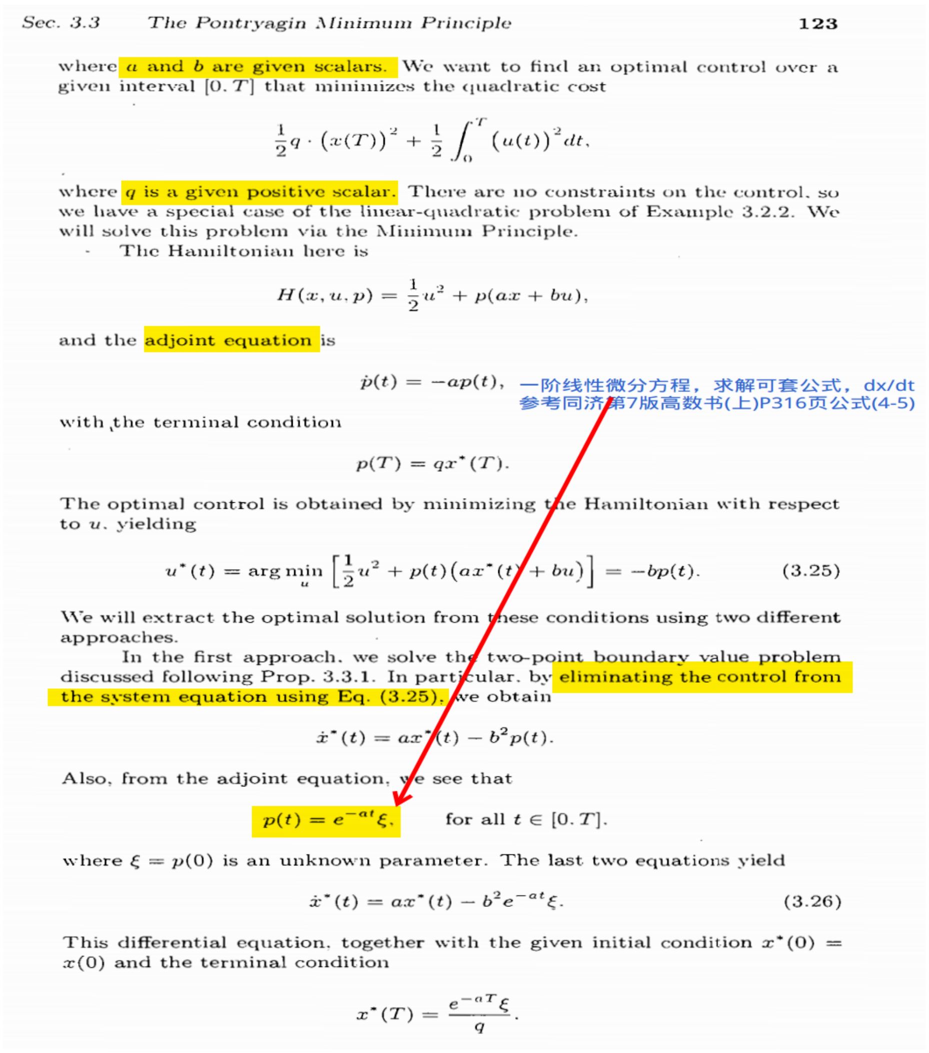


Example 3.3.3 (A Linear-Quadratic Problem) 线性二次问题

Consider the one-dimensional linear system

$$\dot{x}(t) = ax(t) + bu(t),$$

图 1.12 linear-quadratic problem_1[1]



Sec. 3.3 The Pontryagin Minimum Principle 123

where a and b are given scalars. We want to find an optimal control over a given interval $[0, T]$ that minimizes the quadratic cost

$$\frac{1}{2}q \cdot (x(T))^2 + \frac{1}{2}\int_0^T (u(t))^2 dt,$$

where q is a given positive scalar. There are no constraints on the control, so we have a special case of the linear-quadratic problem of Example 3.2.2. We will solve this problem via the Minimum Principle.

The Hamiltonian here is

$$H(x, u, p) = \frac{1}{2}u^2 + p(ax + bu),$$

and the adjoint equation is

$$\dot{p}(t) = -ap(t),$$

一阶线性微分方程，求解可套公式，dx/dt 参考同济第7版高数书(上)P316页公式(4-5)

with the terminal condition

$$p(T) = qx^*(T).$$

The optimal control is obtained by minimizing the Hamiltonian with respect to u, yielding

$$u^*(t) = \arg\min_u \left[\frac{1}{2}u^2 + p(t)\big(ax^*(t) + bu\big)\right] = -bp(t). \tag{3.25}$$

We will extract the optimal solution from these conditions using two different approaches.

In the first approach, we solve the two-point boundary value problem discussed following Prop. 3.3.1. In particular, by eliminating the control from the system equation using Eq. (3.25), we obtain

$$\dot{x}^*(t) = ax^*(t) - b^2 p(t).$$

Also, from the adjoint equation, we see that

$$p(t) = e^{-at}\xi, \qquad \text{for all } t \in [0, T],$$

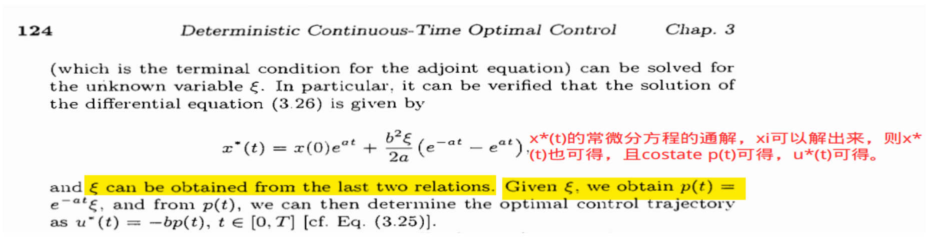
where $\xi = p(0)$ is an unknown parameter. The last two equations yield

$$\dot{x}^*(t) = ax^*(t) - b^2 e^{-at}\xi. \tag{3.26}$$

This differential equation, together with the given initial condition $x^*(0) = x(0)$ and the terminal condition

$$x^*(T) = \frac{e^{-aT}\xi}{q}.$$

图 1.13 linear-quadratic problem_2[1]



124 Deterministic Continuous-Time Optimal Control Chap. 3

(which is the terminal condition for the adjoint equation) can be solved for the unknown variable ξ. In particular, it can be verified that the solution of the differential equation (3.26) is given by

$$x^*(t) = x(0)e^{at} + \frac{b^2\xi}{2a}\left(e^{-at} - e^{at}\right),$$

x*(t)的常微分方程的通解，xi可以解出来，则x*(t)也可得，且costate p(t)可得，u*(t)可得。

and ξ can be obtained from the last two relations. Given ξ, we obtain $p(t) = e^{-at}\xi$, and from $p(t)$, we can then determine the optimal control trajectory as $u^*(t) = -bp(t)$, $t \in [0, T]$ [cf. Eq. (3.25)].

图 1.14 linear-quadratic problem_3[1]

图 1.13 中，伴随方程实际上就是课程中解出 λ 的微分方程，一阶微分方程求解方式可直接套用公式。对于形如 (1.19) 类型的微分方程，可以使

用式 (1.20) 的公式求通解，借助边界条件来求特解。

$$\frac{dy}{dx}+P(x)y=Q(x) \tag{1.20}$$

$$y=e^{-\int P(x)dx}(\int Q(x)e^{\int P(x)dx}dx+C) \tag{1.21}$$

由图 1.12 中 (3.25) 的微分方程整理得：

$$\frac{d(p(t))}{dt}+ap(t)=0 \tag{1.22}$$

对应 (1.19) 得 $P(x)=a, Q(x)=0$，使用 (1.20) 得

$$p(t)=e^{-\int adt}(\int 0dt+C)=e^{-at}C \tag{1.23}$$

令 $C=\xi$，则有

$$p(t)=e^{-at}\xi \tag{1.24}$$

由图 1.12 中 (3.26) 的微分方程整理得：

$$\frac{dx}{dt}-ax=-b^2e^{-at}\xi \tag{1.25}$$

其中 $P(x)=-a, Q(x)=-b^2\xi e^{-at}$ 使用公式 (1.20) 求得 costate 约束

$$\begin{aligned}x^*(t)&=e^{-\int(-a)dt}(\int -b^2\xi e^{-at}e^{\int -adt}dt+C)\\&=e^{at}(\frac{b^2\xi}{2a}e^{-2at}+C)\end{aligned} \tag{1.26}$$

有初始条件

$$x^*(0)=x(0) \tag{1.27}$$

带入 (1.25) 得

$$x(0)=\frac{b^2\xi}{2a}+C \tag{1.28}$$

则

$$C=x(0)-\frac{b^2\xi}{2a} \tag{1.29}$$

$$\begin{aligned}x^*(t)&=e^{at}(\frac{b^2\xi}{2a}e^{-2at}+x(0)-\frac{b^2\xi}{2a})\\&=x(0)e^{at}+\frac{b^2\xi}{2a}(e^{-at}-e^{at})\end{aligned} \tag{1.30}$$

有终止条件

$$x^*(T)=\frac{e^{-aT}\xi}{q} \tag{1.31}$$

带入 (1.29) 得

$$\frac{e^{-aT}\xi}{q} = e^{aT}(\frac{b^2\xi}{2a}e^{-2aT} + x(0) - \frac{b^2\xi}{2a}) \tag{1.32}$$

由 (1.32) 可解出 ξ，可得 $x^*(t)$ 关于 T 的通解，且 costate $p(t)$ 可得，$u^*(t)$ 可得，整个系统都可求解。

以上即为 Pontryagin Minimum Principle 的拓展，主要强调 adjoint equation 和一阶微分方程的求解方法，更多细节参考 [2]。

2 第 2 题

2.1 题目

Local Lattice Planner

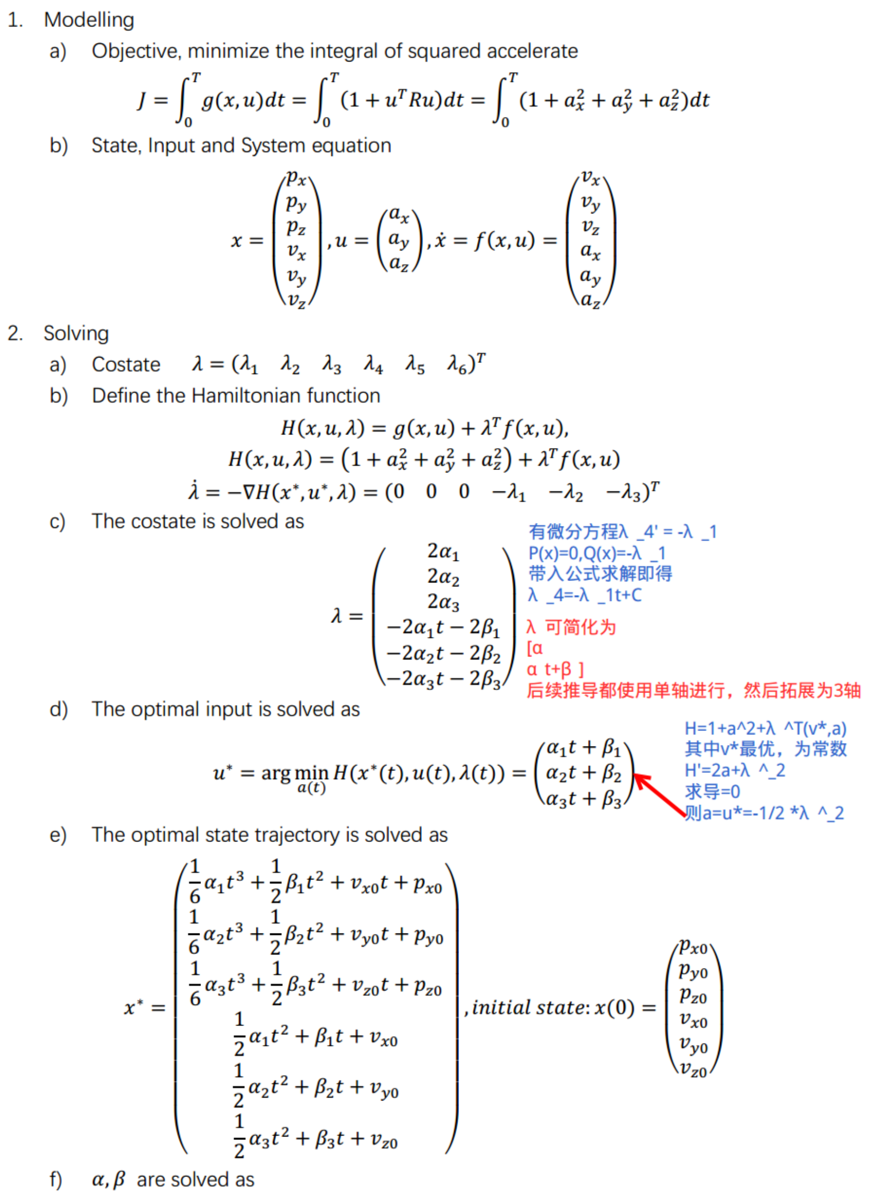
- Build an ego-graph of the linear modeled robot.

- Select the best trajectory closest to the planning target.

2.2 求解

理论推导部分

已经给出的建模和求解如图 2.1 和图 2.2 所示



1. Modelling
   a) Objective, minimize the integral of squared accelerate

$$J = \int_0^T g(x,u)dt = \int_0^T (1 + u^T R u)dt = \int_0^T (1 + a_x^2 + a_y^2 + a_z^2)dt$$

   b) State, Input and System equation

$$x = \begin{pmatrix} p_x \\ p_y \\ p_z \\ v_x \\ v_y \\ v_z \end{pmatrix}, u = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}, \dot{x} = f(x,u) = \begin{pmatrix} v_x \\ v_y \\ v_z \\ a_x \\ a_y \\ a_z \end{pmatrix}$$

2. Solving
   a) Costate $\lambda = (\lambda_1 \quad \lambda_2 \quad \lambda_3 \quad \lambda_4 \quad \lambda_5 \quad \lambda_6)^T$
   b) Define the Hamiltonian function

$$H(x,u,\lambda) = g(x,u) + \lambda^T f(x,u),$$
$$H(x,u,\lambda) = \left(1 + a_x^2 + a_y^2 + a_z^2\right) + \lambda^T f(x,u)$$
$$\dot{\lambda} = -\nabla H(x^*, u^*, \lambda) = (0 \quad 0 \quad 0 \quad -\lambda_1 \quad -\lambda_2 \quad -\lambda_3)^T$$

   c) The costate is solved as

$$\lambda = \begin{pmatrix} 2\alpha_1 \\ 2\alpha_2 \\ 2\alpha_3 \\ -2\alpha_1 t - 2\beta_1 \\ -2\alpha_2 t - 2\beta_2 \\ -2\alpha_3 t - 2\beta_3 \end{pmatrix}$$

有微分方程λ _4' = -λ _1
P(x)=0,Q(x)=-λ _1
带入公式求解即得
λ _4=-λ _1t+C

λ 可简化为
[α
α t+β]
后续推导都使用单轴进行，然后拓展为3轴

   d) The optimal input is solved as

$$u^* = \arg\min_{a(t)} H(x^*(t), u(t), \lambda(t)) = \begin{pmatrix} \alpha_1 t + \beta_1 \\ \alpha_2 t + \beta_2 \\ \alpha_3 t + \beta_3 \end{pmatrix}$$

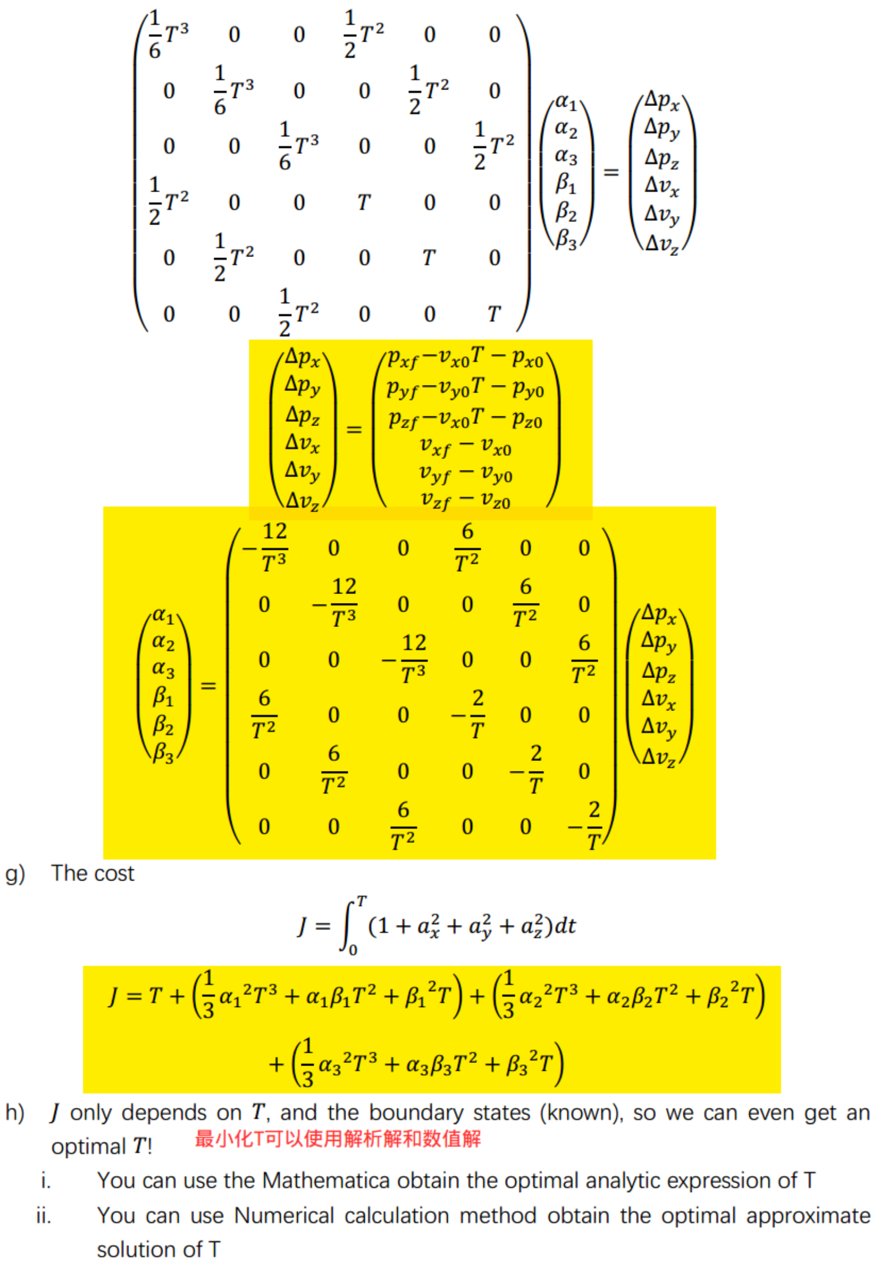
H=1+a^2+λ ^T(v*,a)
其中v*最优，为常数
H'=2a+λ ^_2
求导=0
则a=u*=-1/2 *λ ^_2

   e) The optimal state trajectory is solved as

$$x^* = \begin{pmatrix} \frac{1}{6}\alpha_1 t^3 + \frac{1}{2}\beta_1 t^2 + v_{x0} t + p_{x0} \\ \frac{1}{6}\alpha_2 t^3 + \frac{1}{2}\beta_2 t^2 + v_{y0} t + p_{y0} \\ \frac{1}{6}\alpha_3 t^3 + \frac{1}{2}\beta_3 t^2 + v_{z0} t + p_{z0} \\ \frac{1}{2}\alpha_1 t^2 + \beta_1 t + v_{x0} \\ \frac{1}{2}\alpha_2 t^2 + \beta_2 t + v_{y0} \\ \frac{1}{2}\alpha_3 t^2 + \beta_3 t + v_{z0} \end{pmatrix}, initial\ state: x(0) = \begin{pmatrix} p_{x0} \\ p_{y0} \\ p_{z0} \\ v_{x0} \\ v_{y0} \\ v_{z0} \end{pmatrix}$$

   f) α, β are solved as

图 2.1 建模 & 求解 _1



$$\begin{pmatrix} \frac{1}{6}T^3 & 0 & 0 & \frac{1}{2}T^2 & 0 & 0 \\ 0 & \frac{1}{6}T^3 & 0 & 0 & \frac{1}{2}T^2 & 0 \\ 0 & 0 & \frac{1}{6}T^3 & 0 & 0 & \frac{1}{2}T^2 \\ \frac{1}{2}T^2 & 0 & 0 & T & 0 & 0 \\ 0 & \frac{1}{2}T^2 & 0 & 0 & T & 0 \\ 0 & 0 & \frac{1}{2}T^2 & 0 & 0 & T \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix} = \begin{pmatrix} \Delta p_x \\ \Delta p_y \\ \Delta p_z \\ \Delta v_x \\ \Delta v_y \\ \Delta v_z \end{pmatrix}$$

$$\begin{pmatrix} \Delta p_x \\ \Delta p_y \\ \Delta p_z \\ \Delta v_x \\ \Delta v_y \\ \Delta v_z \end{pmatrix} = \begin{pmatrix} p_{xf} - v_{x0}T - p_{x0} \\ p_{yf} - v_{y0}T - p_{y0} \\ p_{zf} - v_{x0}T - p_{z0} \\ v_{xf} - v_{x0} \\ v_{yf} - v_{y0} \\ v_{zf} - v_{z0} \end{pmatrix}$$

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix} = \begin{pmatrix} -\frac{12}{T^3} & 0 & 0 & \frac{6}{T^2} & 0 & 0 \\ 0 & -\frac{12}{T^3} & 0 & 0 & \frac{6}{T^2} & 0 \\ 0 & 0 & -\frac{12}{T^3} & 0 & 0 & \frac{6}{T^2} \\ \frac{6}{T^2} & 0 & 0 & -\frac{2}{T} & 0 & 0 \\ 0 & \frac{6}{T^2} & 0 & 0 & -\frac{2}{T} & 0 \\ 0 & 0 & \frac{6}{T^2} & 0 & 0 & -\frac{2}{T} \end{pmatrix} \begin{pmatrix} \Delta p_x \\ \Delta p_y \\ \Delta p_z \\ \Delta v_x \\ \Delta v_y \\ \Delta v_z \end{pmatrix}$$

g) The cost

$$J = \int_0^T (1 + a_x^2 + a_y^2 + a_z^2)dt$$

$$J = T + \left(\frac{1}{3}{\alpha_1}^2T^3 + \alpha_1\beta_1T^2 + {\beta_1}^2T\right) + \left(\frac{1}{3}{\alpha_2}^2T^3 + \alpha_2\beta_2T^2 + {\beta_2}^2T\right) + \left(\frac{1}{3}{\alpha_3}^2T^3 + \alpha_3\beta_3T^2 + {\beta_3}^2T\right)$$

h) J only depends on T, and the boundary states (known), so we can even get an optimal T! 最小化T可以使用解析解和数值解

i. You can use the Mathematica obtain the optimal analytic expression of T

ii. You can use Numerical calculation method obtain the optimal approximate solution of T

图 2.2 建模 & 求解 _2

系统输入为 $u(t) = a(t)$

根据助教的讲解，这里可以把 final state 中的速度看做常量，在推导过程中直接假设 final state 中速度为 0，即 $v_{xf} = v_{yf} = v_{zf} = 0$（这里还不知道为什么可以看做常量）。

所给推导中考虑了三个轴的情况，为了简化，可以只推导单轴情况再拓

展为 3 轴，则对于 costate 有

$$\begin{cases} \alpha = -\frac{12}{T^3}\Delta p + \frac{6}{T^2}\Delta v \\ \beta = \frac{6}{T^2}\Delta p - \frac{2}{T}\Delta v \end{cases} \tag{2.1}$$

其中

$$\begin{cases} \Delta p = p_f - p_0 - v_0 T = d_x - v_0 T \\ \Delta v = v_f - v_0 = -v_0 \end{cases} \tag{2.2}$$

根据前面推导

$$J = T + \frac{1}{3}\alpha^2 T^3 + \alpha\beta T^2 + \beta^2 T \tag{2.3}$$

将 (2.2) 带入 (2.1) 得

$$\begin{cases} \alpha = -\frac{12}{T^3}d_x + \frac{6}{T^2}v_0 \\ \beta = \frac{6}{T^2}d_x - \frac{4}{T}v_0 \end{cases} \tag{2.4}$$

整理 (2.3) 中各项可得

$$\begin{cases} \frac{1}{3}\alpha^2 T^3 = \frac{48}{T^3}d_x^2 - \frac{48}{T^2}v_0 d_x + \frac{12}{T}v_0^2 \\ \alpha\beta T^2 = -\frac{72}{T^3}d_x^2 + \frac{84}{T}v_0 d_x - \frac{24}{T}v_0^2 \\ \beta^2 T = \frac{36}{T^3}d_x^2 - \frac{48}{T^2}v_0 d_x + \frac{16}{T}v_0^2 \end{cases} \tag{2.5}$$

将 (2.5) 带入 (2.3) 得

$$J = T + \frac{12}{T^3}d_x^2 - \frac{12}{T^2}v_0 d_x + \frac{4}{T}v_0^2 \tag{2.6}$$

此时 cost J 只跟 T 有关，根据 Pontryagin Minimum Principle，要对 J 进行最小化，对 J 求导等于 0

$$\frac{\partial J}{\partial T} = 0 \tag{2.7}$$

最终整理为 (2.8) 所示的一元四次方程：

$$T^4 - 4v_0^2 T^2 + 24 v_0 d_x T - 36 d_x^2 = 0 \tag{2.8}$$

方程拓展为三维为：

$$\begin{aligned} T^4 - 4(v_{x0}^2 + v_{y0}^2 + v_{z0}^2)T^2 + 24(v_{x0}d_x + v_{y0}d_y + v_{z0}d_z)T \\ -36(d_x^2 + d_y^2 + d_z^2) = 0 \end{aligned} \tag{2.9}$$

式 (2.6)cost 拓展为三维为

$$\begin{aligned} J = T + \frac{4(v_{x0}^2 + v_{y0}^2 + v_{z0}^2)}{T} - \frac{12(v_{x0}d_x + v_{y0}d_y + v_{z0}d_z)}{T^2} \\ + \frac{12(d_x^2 + d_y^2 + d_z^2)}{T^3} \end{aligned} \tag{2.10}$$

求解 (2.9) 可得 T，带入 (2.10) 求得 cost，取 cost 最小的 T，且 check collision-free 即可完成求解。其中，求解一元四次方程可参考 [3]

这里采用了方法二，使用多项式的伴随矩阵进行求解：对于多项式

$$P(x) = x^n + c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \cdots + c_1 x + c_0 \tag{2.11}$$

其伴随矩阵为：

$$M_x = \begin{bmatrix} 0 & 0 & \cdots & 0 & -c_0 \\ 1 & 0 & \cdots & 0 & -c_1 \\ 0 & 1 & \cdots & 0 & -c_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 & -c_{n-1} \end{bmatrix} \tag{2.12}$$

其中，M_x 的特征值就是 $P(x)$ 的根。所以这里构造的伴随矩阵为

$$\begin{bmatrix} 0 & 0 & 0 & 36(d_x^2 + d_y^2 + d_z^2) \\ 1 & 0 & 0 & -24(v_{x0}d_x + v_{y0}d_y + v_{z0}d_z) \\ 0 & 1 & 0 & 4(v_{x0}^2 + v_{y0}^2 + v_{z0}^2) \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.13}$$

代码部分

STEP1 主要使用运动学公式

$$\begin{cases} p = p_0 + v_0 t + \frac{1}{2}at^2 \\ v = v_0 + at \end{cases} \tag{2.14}$$

代码如 Listing1 所示。

```
for(int step=0 ; step<=_time_step ; step ++){
    /*
    STEP 1: finish the forward integration, the modelling has been given in
     the document
    the parameter of forward integration: _max_input_acc|_discretize_step|
     _time_interval|_time_step   all have been given
    use the pos and vel to recored the steps in the trakectory
    */
    //使用运动学公式: x=x0+v0t+1/2at^2,v=v0+at
    //要体现出积分，是accumulate的过程
    pos(0) = pos(0) + vel(0) * delta_time + 0.5 * acc_input(0) * std::pow(
     delta_time, 2);
    pos(1) = pos(1) + vel(1) * delta_time + 0.5 * acc_input(1) * std::pow(
     delta_time, 2);
    pos(2) = pos(2) + vel(2) * delta_time + 0.5 * acc_input(2) * std::pow(
     delta_time, 2);
    vel(0) = vel(0) + acc_input(0) * delta_time;
    vel(1) = start_velocity(1) + acc_input(1) * delta_time;
    vel(2) = start_velocity(2) + acc_input(2) * delta_time;

    Position.push_back(pos);
    Velocity.push_back(vel);
    double coord_x = pos(0);
    double coord_y = pos(1);
    double coord_z = pos(2);
    //check if if the trajectory face the obstacle
    if(_homework_tool->isObsFree(coord_x,coord_y,coord_z) != 1){
        collision = true;
    }
}
```

Listing 1: demo_node.cpp/trajectoryLibrary()

STEP2 主要构造 (2.13) 所述的伴随矩阵求根 T，取出实部为正的最小 cost，代码如 Listing2 所示。

```
double Homeworktool::OptimalBVP(Eigen::Vector3d _start_position,Eigen::
    Vector3d _start_velocity,Eigen::Vector3d _target_position)
{
    double optimal_cost = 1000000; // this just to initial the optimal_cost,
     you can delete it
    /*
    STEP 2: go to the hw_tool.cpp and finish the function Homeworktool::
    OptimalBVP
    the solving process has been given in the document

    because the final point of trajectory is the start point of OBVP, so we
    input the pos,vel to the OBVP

    after finish Homeworktool::OptimalBVP, the Trajctory_Cost will record
    the optimal cost of this trajectory

    */
    //直接求解一元四次方程

    Eigen::Matrix<double,4,4> mat44;
    Eigen::Matrix<complex<double>, Eigen::Dynamic, Eigen::Dynamic>
    matrix_eigenvalues;
    //求出一元四次方程的五个系数abcde
    double v_x0 = _start_velocity(0), v_y0 = _start_velocity(1), v_z0 =
    _start_velocity(2);

    double dx = _target_position(0) - _start_position(0);
    double dy = _target_position(1) - _start_position(1);
    double dz = _target_position(2) - _start_position(2);
    double v0_square_sum = v_x0*v_x0 + v_y0*v_y0 + v_z0*v_z0;
    double v0_dp_sum = v_x0*dx+v_y0*dy+v_z0*dz;
    double dp_square_sum = dx*dx + dy*dy + dz*dz;

    double a = 1.0;
```

```cpp
    double b = 0.0;
    double c = -4 * v0_square_sum;
    double d = 24 * v0_dp_sum;
    double e = -36 * dp_square_sum;

    mat44 <<
            0, 0, 0, -e,
            1, 0, 0, -d,
            0, 1, 0, -c,
            0, 0, 1, -b;
//    ROS_INFO_STREAM("\nmatrix_44: \n" << mat44);
    matrix_eigenvalues = mat44.eigenvalues();
//    ROS_INFO_STREAM("\nmatrix_eigenvalues: \n"<<matrix_eigenvalues);

    for(int i=0; i<4; ++i) {
        if(matrix_eigenvalues(i).real() < 0)
            continue;
        double T = matrix_eigenvalues(i).real();
        double tmp_cost = T + 4.0/T * v0_square_sum - 12.0/(std::pow(T,2)) * v0_dp_sum + 12.0/std::pow(T,3) * dp_square_sum;
        ROS_INFO_STREAM("\nnow optimal_cost=" << optimal_cost <<", tmp_cost=" << tmp_cost);
        if(tmp_cost < optimal_cost) {
            ROS_INFO_STREAM("\n======now optimal_cost=" << optimal_cost <<", found lower cost= " << tmp_cost);
            optimal_cost = tmp_cost;
        }
    }
    ROS_INFO_STREAM("\nreturn optimal_cost=" << optimal_cost);
    return optimal_cost;
}
```

Listing 2: hw_tool.cpp/OptimalBVP()

最终运行结果如图 2.3 所示，其中 green for optimal trajectory,red for not collision-free trajectory, blue for collision-free trajectory.

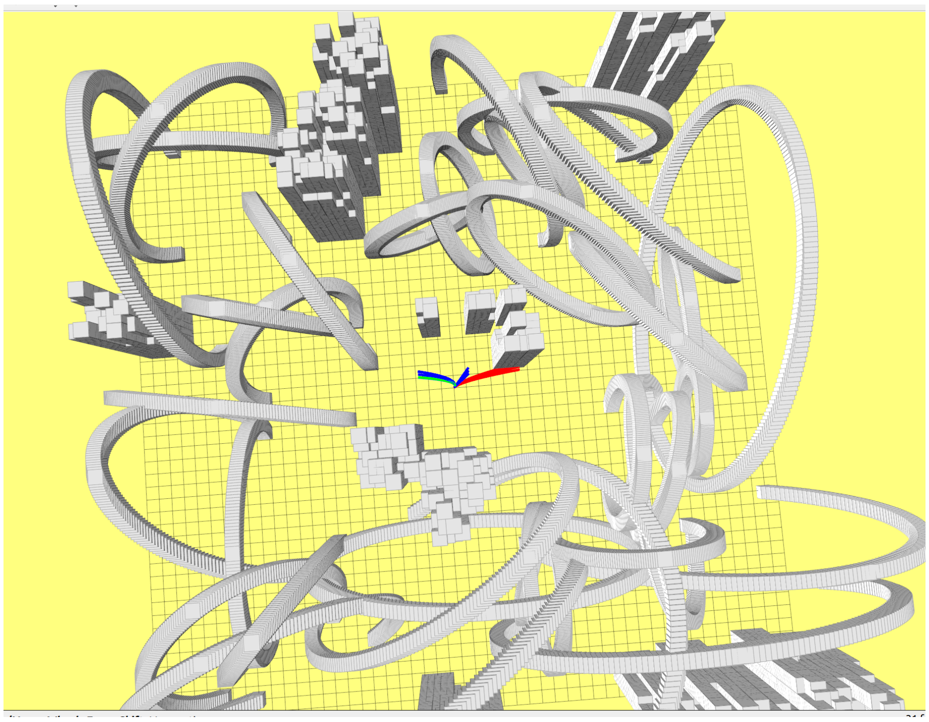


图 2.3 运行结果

参考文献

[1] A Computationally Efficient Motion Primitive for Quadrocopter Trajectory Generation, Mark W. Mueller, Markus Hehn, and Raffaello D’Andrea.

[2] Dynamic Programming and Optimal Control, D. P. Bertsekas.

[3] https://blog.csdn.net/fb__941219/article/details/102984587