深蓝学院
shenlanxueyuan.com

第六章作业思路分享

主讲人 丁成科

https://pomax.github.io/bezierinfo/zh-CN/index.html#extended
https://en.wikipedia.org/wiki/Bernstein_polynomial

思路分享

深蓝学院 shenlanxueyuan.com

- Monomial 多项式和Bezeir曲线是可以互相转化的

$$P_j(t) = p_j^0 + p_j^1 t + p_j^2 t^2 + \cdots + p_j^n t^n \quad \rightarrow \quad \begin{aligned} B_j(t) &= c_j^0 b_n^0(t) + c_j^1 b_n^1(t) + \cdots + c_j^n b_n^n(t) = \sum_{i=0}^{n} c_j^i b_n^i(t) \\ b_n^i(t) &= \binom{n}{i} \cdot t^i \cdot (1-t)^{n-i} \end{aligned}$$

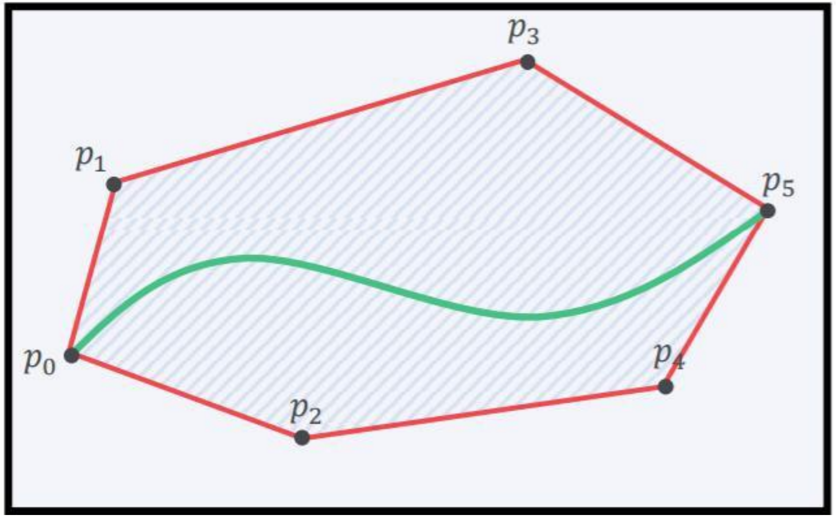
- 两者的系数可以通过M矩阵转换，有p=Mc

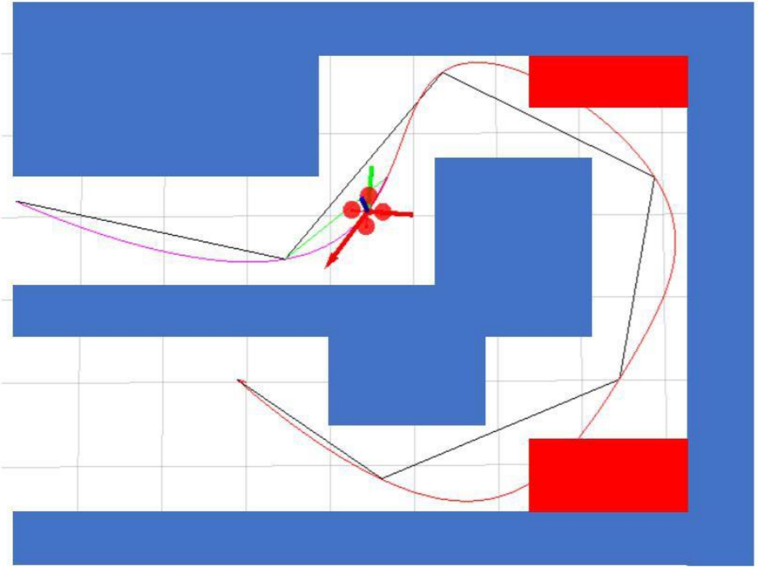
$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -6 & 6 & 0 & 0 & 0 & 0 & 0 \\ 15 & -30 & 15 & 0 & 0 & 0 & 0 \\ -20 & 60 & -60 & 20 & 0 & 0 & 0 \\ 15 & -60 & 90 & -60 & 15 & 0 & 0 \\ -6 & 30 & -60 & 60 & -30 & 6 & 0 \\ 1 & -6 & 15 & -20 & 15 & -6 & 1 \end{bmatrix}$$

https://pomax.github.io/bezierinfo/zh-CN/index.html#extended
https://en.wikipedia.org/wiki/Bernstein_polynomial

思路分享

- **Convex hull.** The Bezier curve $B(t)$ consists of a set of control points ci are entirely confined within the convex hull defined by all these control points.





- 控制控制点在安全区域内+Convex hull性质 = 保证轨迹在安全区内，不会出现右图的情况

思路分享

- 注意多项式的时间取值是在[0,1]内，所以对于实际问题需要进行归一化，多项式组可以写成

$$f_\mu(t)=\begin{cases} s_1\cdot\sum_{i=0}^{n} c_{\mu1}^{i} b_n^i\left(\frac{t}{s_1}\right), & t\in[0,T_1] \\ s_2\cdot\sum_{i=0}^{n} c_{\mu2}^{i} b_n^i\left(\frac{t}{s_2}\right), & t\in[0,T_2] \\ \vdots & \vdots \\ s_m\cdot\sum_{i=0}^{n} c_{\mu m}^{i} b_n^i\left(\frac{t}{s_m}\right), & t\in[0,T_m] \end{cases}$$

s1,s2,….为归一化系数

https://ieeexplore.ieee.org/document/8462878

思路分享

深蓝学院
shenlanxueyuan.com

- 而目标函数同样使用minimum snap，其结论基本和第五章的类似
- 但注意若使用Bezier多项式，其Q函数会有些变化， 每一项需要乘上s^-(2k-3)，其中使用minimum snap时k=4

$$J=\int_0^T\left(\frac{\mathrm{d}^k\left(s\cdot\sum c_i b^i\left(\frac{t}{s}\right)\right)}{dt^4}\right)^2 dt, t\in[0,T]$$

$$=\int_0^1 s^3\left(\frac{\mathrm{d}^k\left(\sum c_i b^i(\tau)\right)}{s^k d\tau^4}\right)^2 d\tau, \tau\in[0,1]$$

$$=s^{-(2k-3)}\int_0^1\left(\frac{\mathrm{d}^k\left(\sum p_i\tau^i\right)}{d\tau^4}\right)^2 d\tau, \tau\in[0,1]$$

$$=s^{-(2k-3)}\int_0^1 (f^4(\tau))^2 d\tau, \tau\in[0,1]$$

$$=\begin{bmatrix}\cdots\\ p_i\\ \cdots\end{bmatrix}^T\left[\cdots\frac{i(i-1)(i-2)(i-3)l(l-1)(l-2)(l-3)s^{-(2k-3)}}{i+l-7}\cdots\right]\begin{bmatrix}\cdots\\ p_i\\ \cdots\end{bmatrix}$$

思路分享

- 因此Q函数的写法会有少许不同，需要/s^(2*4-3)，积分上限是1。其余部分和第五章的一致
- 注意这里的s直接取该时间段的时间值

```
Q = [];
M = [];
M_k = getM(n_order);
for k = 1:n_seg
    %#####################################################
    % STEP 2.1 calculate Q_k of the k-th segment, minimize snap
    Q_k = [];
    t_k = 1;
    s_k = ts(k);
    for i = 4:n_order
        for l = 4:n_order
            den = i + l - 7;
            Q_k(i+1,l+1) = i*(i-1)*(i-2)*(i-3)*l*(l-1)*(l-2)*(l-3)/den*(t_k^den)/s_k^(2*4-3);
        end
    end

    Q = blkdiag(Q, Q_k);
    M = blkdiag(M, M_k);
end
```

思路分享

- 等式约束条件: Boundary constraints (1) 开始点的P,V,A (2) 终点的P,V,A

```
n_all_poly = n_seg*(n_order+1);
%#####################################################
% STEP 2.1 p,v,a constraint in start
Aeq_start = zeros(3, n_all_poly);
beq_start = start_cond';
S = ts(1);
k = 0; Aeq_start(k+1, 1:3) = [1,0,0]*S^(1-k);
k = 1; Aeq_start(k+1, 1:3) = [-1,1,0]*n_order*S^(1-k);
k = 2; Aeq_start(k+1, 1:3) = [1,-2,1]*n_order*(n_order-1)*S^(1-k);

%#####################################################
% STEP 2.2 p,v,a constraint in end
Aeq_end = zeros(3, n_all_poly);
beq_end = end_cond';
S = ts(end);
idx = (n_seg-1)*(n_order+1) + n_order+1 - 3;
k = 0; Aeq_end(k+1, idx+(1:3)) = [0,0,1]*S^(1-k);
k = 1; Aeq_end(k+1, idx+(1:3)) = [0,-1,1]*n_order*S^(1-k);
k = 2; Aeq_end(k+1, idx+(1:3)) = [1,-2,1]*n_order*(n_order-1)*S^(1-k);
```

- 注意在使用控制点时，需要乘上归一化系数

- Boundary Constraints:

$$\boxed{a^{l,0}_{\mu j} \cdot s_j^{(1-l)} = d^{(l)}_{\mu j}}$$

- Continuity Constraints:

$$\boxed{a^{\phi,n}_{\mu j} \cdot s_j^{(1-\phi)} = a^{\phi,0}_{\mu,j+1} \cdot s_{j+1}^{(1-\phi)}, \quad a^{0,i}_{\mu j} = c^i_{\mu j}.}$$

Stack

- Safety Constraints:

$$\boxed{\beta^-_{\mu j} \le c^i_{\mu j} S \le \beta^+_{\mu j}, \quad \mu \in \{x, y, z\}, \quad i = 0, 1, 2, ..., n,}$$

- Dynamical Feasibility Constraints:

$$\boxed{\begin{aligned} &v^-_m \le n \cdot (c^i_{\mu j} - c^{i-1}_{\mu j}) \le v^+_m, \\ &a^-_m \le n \cdot (n-1) \cdot (c^i_{\mu j} - 2c^{i-1}_{\mu j} + c^{i-2}_{\mu j})/s_j \le a^+_m \end{aligned}}$$

https://github.com/HKUST-Aerial-Robotics/Btraj

深蓝学院 shenlanxueyuan.com

思路分享

- 等式约束条件: Continuity constraints (3,4,5) 段的P,V,A连续

```
% STEP 2.3 position continuity constrain between 2 segments
Aeq_con_p = zeros(n_seg - 1, n_all_poly);
beq_con_p = zeros(n_seg - 1, 1);

for k = 1:n_seg - 1
    s1 = ts(k);
    s2 = ts(k+1);
    Aeq_con_p(k,8*k:8*k+1) = [1*s1,-1*s2];
end
```

- 注意控制点的归一化系数是不一样的

- Boundary Constraints:

$$\boxed{a_{\mu j}^{l,0} \cdot s_j^{(1-l)} = d_{\mu j}^{(l)}}$$

- Continuity Constraints:

$$\boxed{a_{\mu j}^{\phi,n} \cdot s_j^{(1-\phi)} = a_{\mu,j+1}^{\phi,0} \cdot s_{j+1}^{(1-\phi)}, \quad a_{\mu j}^{0,i} = c_{\mu j}^{i}.}$$

Stack

```
% STEP 2.4 velocity continuity constrain between 2 segments
Aeq_con_v = zeros(n_seg - 1, n_all_poly);
beq_con_v = zeros(n_seg - 1, 1);

for k = 1:n_seg - 1
    Aeq_con_v(k,8*k-1:8*k+2) = n_order*[-1,1,1,-1];
end
```

```
% STEP 2.5 acceleration continuity constrain between 2 segments
Aeq_con_a = zeros(n_seg - 1, n_all_poly);
beq_con_a = zeros(n_seg - 1, 1);

for k = 1:n_seg - 1
    s1 = ts(k);
    s2 = ts(k+1);
    Aeq_con_a(k,8*k-2:8*k+3) = n_order*(n_order-1)*[1/s1,-2/s1,1/s1,-1/s2,2/s2,-1/s2];
end
```

- 位置要乘s，速度不用变，加速度要除s

思路分享

深蓝学院
shenlanxueyuan.com

- 不等式约束条件: (1)Safety constraints :控制点被corridor_range包围

```
% STEP 3.2.1 p constraint
Aieq_p = zeros(n_seg*(n_order+1)*2,n_all_poly);
bieq_p = zeros(n_seg*(n_order+1)*2,1);

for k = 1:n_seg
    s = ts(k);
    for i = 1: (n_order+1)
        Aieq_p(i+16*(k-1),i+8*(k-1)) = 1*s;
        Aieq_p(8+i+16*(k-1),i+8*(k-1)) = -1*s;
        bieq_p(i+16*(k-1)) = corridor_range(k,2);
        bieq_p(8+i+16*(k-1)) = -corridor_range(k,1);
    end
end
```

- Safety Constraints:

$$\boxed{\beta^-_{\mu j} \le c^i_{\mu j} s \le \beta^+_{\mu j}, \quad \mu \in \{x,y,z\}, \quad i = 0,1,2,...,n,}$$

- 大于端的处理方法是两边同时乘-1变成小于
- 位置要乘s

思路分享

- 不等式约束条件: (2,3) Dynamical Feasibility Constraints: 以控制点形式的v,a受到限制

```
% STEP 3.2.2 v constraint
Aieq_v = zeros(n_seg*(n_order)*2,n_all_poly);
bieq_v = zeros(n_seg*(n_order)*2,1);

for k = 1:n_seg
    for i = 1: (n_order)
        Aieq_v(i+14*(k-1),i+8*(k-1):i+8*(k-1)+1) = n_order*[-1,1];
        Aieq_v(7+i+14*(k-1),i+8*(k-1):i+8*(k-1)+1) = -n_order*[-1,1];
        bieq_v(i+14*(k-1)) = v_max;
        bieq_v(7+i+14*(k-1)) = v_max;
    end
end
```

```
% STEP 3.2.3 a constraint
Aieq_a = zeros(n_seg*(n_order-1)*2,n_all_poly);
bieq_a = zeros(n_seg*(n_order-1)*2,1);

for k = 1:n_seg
    s = ts(k);
    for i = 1: (n_order-1)
        Aieq_a(i+12*(k-1),i+8*(k-1):i+8*(k-1)+2) = n_order*(n_order-1)/s*[1,-2,1];
        Aieq_a(6+i+12*(k-1),i+8*(k-1):i+8*(k-1)+2) = n_order*(n_order-1)/s*[-1,2,-1];
        bieq_a(i+12*(k-1)) = a_max;
        bieq_a(6+i+12*(k-1)) = a_max;
    end
end
```

- Dynamical Feasibility Constraints:

$$\boxed{\begin{aligned} &v_m^- \le n \cdot (c_{\mu j}^i - c_{\mu j}^{i-1}) \le v_m^+, \\ &a_m^- \le n \cdot (n-1) \cdot (c_{\mu j}^i - 2c_{\mu j}^{i-1} + c_{\mu j}^{i-2})/s_j \le a_m^+ \end{aligned}}$$

- >的处理方法是两边同时乘-1变成<
- 速度不用变，加速度要除s

深蓝学院
shenlanxueyuan.com

感谢各位聆听!

Thanks for Listening