B样条Clamp路径规划模型

概述

本文档主要针对APS自动泊车路径规划部分；针对目前的模型设计情况，做此文档说明。

通过前文分析针对泊车起始点，计算单次泊车（圆弧直线圆弧）最小车位轨迹，取计算的轨迹点，利用B样条生成连续曲率的轨迹线，在定义的世界坐标系中，实时仿真车辆位置。

规划算法：B样条轨迹规划算法已完成，根据截取的规划点，可生成光滑连续的5次曲线轨迹。

问题及不足：1、现阶段只针对平行泊车轨迹规划部分，模型中找车位的模块用自定义的四点车位坐标代替。后续需能适应变车位坐标模型仿真；

2、现阶段只仿真B样条轨迹算法，基于几何规划，对泊车相关的其他算法（模糊逻辑、神经网络、搜索采样、强化学习）没能对比仿真验证。后续需设计基于库位跟踪的轨迹；

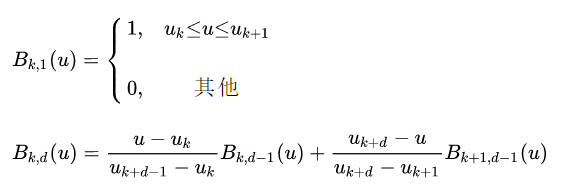
3、在车速和车辆控制方面，采用MPC预测跟踪算法，车速和模型参数局限性很大，只能实现定速前向跟踪控制。后续需继续低速后向跟踪控制；

B样条曲线算法

Clamped B 样条曲线

通过 B 样条理论对圆弧路径曲线进行多阶平滑处理，使平滑处理后的路径满足避障约束和车辆行驶性能参数约束，且曲线始、末点曲率尽量小，甚至为 0。实现车辆泊车结束时车轮回正；

B样条算法详解，根据如下公式，计算样条阶数基函数，再由已知控制点数k，得到轨迹上n个样条离散点；



实现代码如下，

function [Control\_Point,phi,Dest\_row\_x,Dest\_row\_y] = fcn(P)

% n,线条控制点数；k，阶次数

n =15; k =5;

P\_dest = P(1:n,1:2)';

NodeVector = [0,0,0,0,0,0,1,2,3,4,5,6,7,8,9,10,10,10,10,10,10];

% 当起始点和终止点的重复度为最高次数加1是，开B样条变为Clamped B样条，当起始点和终止点重合且重复度为p+1时为闭B样条曲线。

Nik = zeros(n+k, k+1);

umin =0;umax = 10;delta\_u = 0.02;

Num = floor((umax-umin)/delta\_u)+1;

Dest = zeros(2,Num);

Control\_Point = zeros(Num,3);

Dest\_row\_x = zeros(Num,1);

Dest\_row\_y = zeros(Num,1);

m=1;

phi = zeros(Num,1);

N\_i\_5 = zeros(n,1);

for u = umin : delta\_u : umax

for i5 = 1:n

for i4 = i5:i5+1

for i3 = i5:i5+2

for i2 = i5:i5+3

for i1 = i5:i5+4

for i = i5:i5+5

if (u >= NodeVector(i))&&(u<NodeVector(i+1))

Nik(i,1) = 1.0;

else

Nik(i,1) = 0.0;

end

end

Length1 = NodeVector(i1+1) - NodeVector(i1);

Length2 = NodeVector(i1+2) - NodeVector(i1+1);

if Length1 == 0.0

Length1 = 1.0;

end

if Length2 == 0.0

Length2 = 1.0;

end

Nik(i1,2)=(u-NodeVector(i1))/Length1\*Nik(i1,1)...

+(NodeVector(i1+2) - u)/Length2\*Nik(i1+1,1);

end

Length1 = NodeVector(i2+2) - NodeVector(i2);

Length2 = NodeVector(i2+3) - NodeVector(i2+1);

if Length1 == 0.0

Length1 = 1.0;

end

if Length2 == 0.0

Length2 = 1.0;

end

Nik(i2,3)=(u-NodeVector(i2))/Length1\* Nik(i2,2) ...

+ (NodeVector(i2+3) - u) / Length2 \* Nik(i2+1,2);

end

Length1 = NodeVector(i3+3) - NodeVector(i3);

Length2 = NodeVector(i3+4) - NodeVector(i3+1);

if Length1 == 0.0

Length1 = 1.0;

end

if Length2 == 0.0

Length2 = 1.0;

end

Nik(i3,4) = (u-NodeVector(i3))/Length1\* Nik(i3,3) ...

+ (NodeVector(i3+4) - u) / Length2 \* Nik(i3+1,3);

end

Length1 = NodeVector(i4+4) - NodeVector(i4);

Length2 = NodeVector(i4+5) - NodeVector(i4+1);

if Length1 == 0.0

Length1 = 1.0;

end

if Length2 == 0.0

Length2 = 1.0;

end

Nik(i4,5) = (u - NodeVector(i4)) / Length1 \* Nik(i4,4) ...

+ (NodeVector(i4+5) - u) / Length2 \* Nik(i4+1,4);

end

Length1 = NodeVector(i5+5) - NodeVector(i5);

Length2 = NodeVector(i5+6) - NodeVector(i5+1);

if Length1 == 0.0

Length1 = 1.0;

end

if Length2 == 0.0

Length2 = 1.0;

end

Nik(i5,6) = (u - NodeVector(i5)) / Length1 \* Nik(i5,5) ...

+ (NodeVector(i5+6) - u) / Length2 \* Nik(i5+1,5);

N\_i\_5(i5,1) = Nik(i5,6);

end

p\_u = P\_dest \* N\_i\_5;

Dest(:,m) = p\_u;

Dest\_row\_x(m,1) = p\_u(1,1);

Dest\_row\_y(m,1) = p\_u(2,1);

if m>1

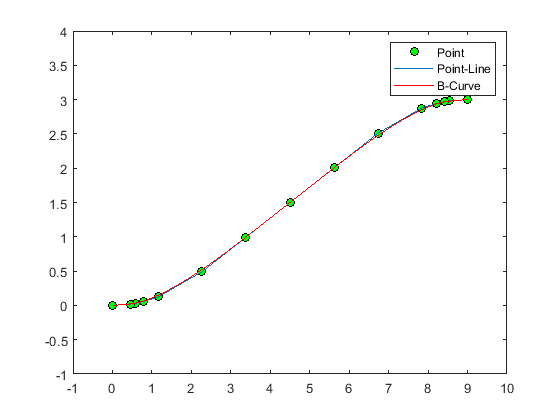
phi(m,1) = (Dest(2,m-1)-Dest(2,m))/(Dest(1,m-1)- Dest(1,m));

end

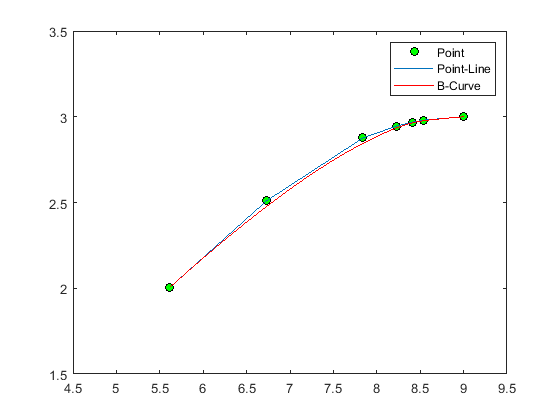
Control\_Point(m,:) = [Dest(:,m)' phi(m,1) ];

m = m+1;

end



15个控制点，5次曲线



7个点，5次曲线

规划过程中，要经过起始点，采用Clamped闭区间B样条，控制节点首尾两处分别设重复度为6的0和1，中间均匀分布；算法可通过设置控制点n的数量和阶次k的大小，生成不同形式的曲线；案例过程分别采用全局15个点和局部7个点做5次B样条轨迹曲线，如图所示。

规则路径约束

据前文中所述，根据车辆倒车运动模型，根据车位检测结果和车辆运动运动约束，可初步得到车辆倒车规则路线，两段圆弧轨迹，圆弧直线圆弧轨迹，最小车位路径，如下图所示；B样条曲线的控制点是从规则路径约束中选取的控制点，目的是生成多阶连续可导的光滑轨迹。以圆弧直线圆弧最小车位轨迹路径规划为例，圆弧直线代码如下所示，







function Route1 = fcn(H,S,Rmin,theta)

R1 = Rmin;

R2 = Rmin;

thetaR0 = 0.01;xR0 = 9;yR0 = 3;

thetaR1 = theta/5;

xR1 = S - R1\*sin(thetaR1);

yR1 = H - R1\*(1-cos(thetaR1));

thetaR2 = theta/4;

xR2 = S - R1\*sin(thetaR2);

yR2 = H - R1\*(1-cos(thetaR2));

thetaR3 = theta/3;

xR3 = S - R1\*sin(thetaR3);

yR3 = H - R1\*(1-cos(thetaR3));

thetaR4 = theta/2;

xR4 = S - R1\*sin(thetaR4);

yR4 = H - R1\*(1-cos(thetaR4));

xLine1 = S - R1\*sin(theta);

yLine1 = H - R1\*(1-cos(theta));

xLine5 = R2 \*sin(theta);

yLine5 = R2 \*(1 - cos(theta));

xLine3 = (xLine1 + xLine5)/2;

yLine3 = tan(theta)\*(xLine3 - R2 \*sin(theta)) + R2 \*(1 - cos(theta));

xLine2 = (xLine3 + xLine1)/2;

yLine2 = tan(theta)\*(xLine2 - R2 \*sin(theta)) + R2 \*(1 - cos(theta));

xLine4 = (xLine5 + xLine3)/2;

yLine4 = tan(theta)\*(xLine4 - R2 \*sin(theta)) + R2 \*(1 - cos(theta));

thetaR5 = theta/5;

xR5 = R2 \*sin(thetaR5);

yR5 = R2 \*(1 - cos(thetaR5));

thetaR6 = theta/4;

xR6 = R2 \*sin(thetaR6);

yR6 = R2 \*(1 - cos(thetaR6));

thetaR7 = theta/3;

xR7 = R2 \*sin(thetaR7);

yR7 = R2 \*(1 - cos(thetaR7));

thetaR8 = theta/2;

xR8 = R2 \*sin(thetaR8);

yR8 = R2 \*(1 - cos(thetaR8));

thetaR9 = 0.01;xR9 = 0;yR9 = 0;

Route1 = [xR0,yR0,thetaR0;...

xR1,yR1,thetaR1;...

xR2,yR2,thetaR2;...

xR3,yR3,thetaR3;...

xR4,yR4,thetaR4;...

xLine1,yLine1,theta;...

xLine2,yLine2,theta;...

xLine3,yLine3,theta;...

xLine4,yLine4,theta;...

xLine5,yLine5,theta;...

xR8,yR8,thetaR8;...

xR7,yR7,thetaR7;...

xR6,yR6,thetaR6;...

xR5,yR5,thetaR5;...

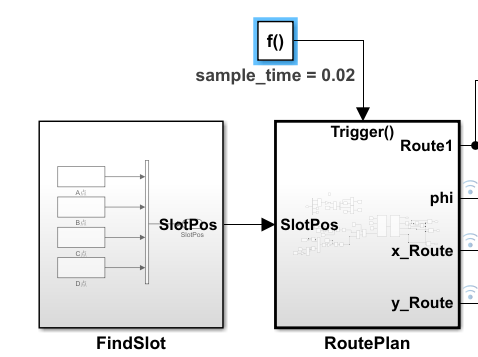
xR9,yR9,thetaR9;

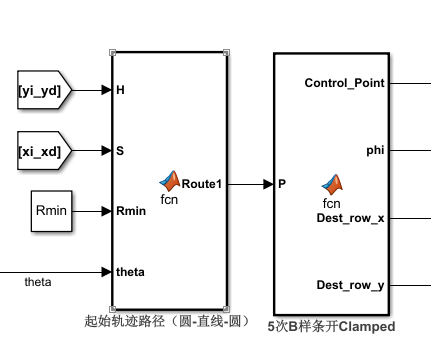
];

end

路径规划模块

初步简化找车位模块，将车位模块用四个已知点代替，定泊车起始点（9，3），（0，0），将上述两路径模块连接起来，生成离散的可跟踪路径点(x,y,phi)。

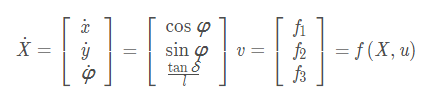




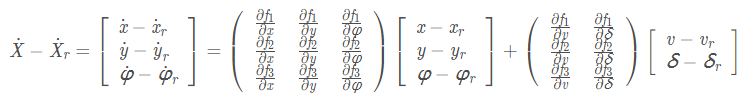
决策控制模块

从已有路径规划模块，设计MPC控制模型。初步设置2个模型控制量速度和前轮偏角，设置3个状态量，分别为车辆坐标和航向角，设置模型预测步长为30步，权重矩阵为0.1倍的单位阵，用二次规划求优化。低速情况下，简化车辆运动学模型，根据运动学公式求解优化问题。

1、运动学模型

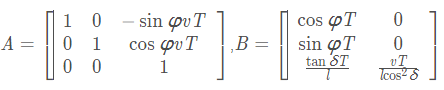


2、线性化



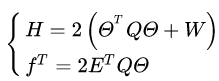
3、离散化





4、目标函数优化求解







用MATLAB函数quadprog可求解二次规划问题，并能生成代码；

MPC模型预测代码如下所示，

function u\_real = fcn(Xout)

Nx=3;

Nu =2;

Tsim =30;

x\_int = [0 0 0.01];

Xout =flipud(Xout);

[Nr,Nc] = size(Xout);

T = 0.05;

L = 2.67;

vd1 = 1;

vd2 = 0;

x\_real=zeros(Nr+1,Nc);

x\_piao=zeros(Nr,Nc);

u\_real=zeros(Nr,2);

u\_piao=zeros(Nr,2);

x\_real(1,:)=x\_int;

x\_piao(1,:)=x\_real(1,:)-Xout(1,:);

X\_PIAO=zeros(Nr,Nx\*Tsim);

XXX=zeros(Nr,Nx\*Tsim);

q=[1 1 0.5];

Q\_cell=repmat(q,1,Tsim);

Q=diag(Q\_cell);

R=0.1\*eye(Nu\*Tsim,Nu\*Tsim);

for i=1:1:Nr

t\_d =Xout(i,3);

a=[1 0 -vd1\*sin(t\_d)\*T;

0 1 vd1\*cos(t\_d)\*T;

0 0 1;];

b=[cos(t\_d)\*T 0;

sin(t\_d)\*T 0;

0 vd1/L\*T;];

A\_cell=zeros(Tsim\*Nx,3); % 60\*3

B\_cell=zeros(Tsim\*Nx,Tsim\*Nu); % 60\*40

for j=1:1:Tsim

A\_cell(j\*Nx-2:j\*Nx,1:3)=a^j;

for k=1:1:Tsim

if k<=j

B\_cell(j\*Nx-2:j\*Nx,k\*Nu-1:k\*Nu)=(a^(j-k))\*b;

end

end

end

A=A\_cell;

B=B\_cell;

H=2\*(B'\*Q\*B+R); % 40\*40

f=2\*B'\*Q\*A\*x\_piao(i,:)'; % 40\*1

A\_cons=[];

b\_cons=[];

options = optimoptions('quadprog','Algorithm','active-set');

xinit = zeros(Tsim\*Nu,1);

X = quadprog(H,f,A\_cons,b\_cons,[],[],[],[],xinit,options);

X\_PIAO(i,:)=(A\*x\_piao(i,:)'+B\*X)'; %

u\_piao(i,1)=X(1,1);

u\_piao(i,2)=X(2,1);

X00=x\_real(i,:);

vd11=vd1+u\_piao(i,1);

vd22=vd2+u\_piao(i,2);

x\_real(i+1,1)=X00(1) + (vd11\*sin(X00(3) + T\*vd22))/vd22 - (vd11\*sin(X00(3)))/vd22;

x\_real(i+1,2)=X00(2) - (vd11\*cos(X00(3) + T\*vd22))/vd22 + (vd11\*cos(X00(3)))/vd22;

x\_real(i+1,3)=X00(3) + T\*vd22;

if(i<Nr)

x\_piao(i+1,:)=x\_real(i+1,:)-Xout(i+1,:);

end

u\_real(i,1)=vd11;

u\_real(i,2)=vd22;

figure(1);

plot(Xout(1:Nr,1),Xout(1:Nr,2),'b','LineWidth',2); %

hold on;

plot(x\_real(i,1),x\_real(i,2),'r\*'); %

hold on;

end

结果

