上海交通大學

课程报告

课程名称：预测控制

题目：无人车轨迹跟踪模型预测控制（A 类）

学院(系)：电子信息与电气工程学院

专　　业：自动化

学生姓名：阮正鑫　　学号:120032910012

2021 年 5 月

一、问题描述

在无人车领域中，如何控制车辆跟随期望的轨迹一直受到学者们的广泛关注。无人驾驶车辆的轨迹跟踪问题是指根据某种控制理论，为系统设计一个控制输入作用，使无人驾驶车辆能够到达并最终以期望的速度跟踪期望轨迹。在惯性坐标系中，车辆必须从一个给定的初始状态出发。这个初始点既可以在期望轨迹上，也可以不在期望轨迹上。在任意时刻k，无人驾驶车辆的轨迹跟踪问题可以如下图所示，其中期望轨迹用一个个离散的轨迹点给出。期望轨迹是指一条几何曲线$f(x_r(t))$，自变量x_r是时间的函数，曲线方程是t的隐函数。给定的速度是指广义的速度变量，也即参考控制输入，包括速度、角速度和前轮偏角等，用u_r表示。在本文中，设定期望轨迹和参考控制输入已经被给出。

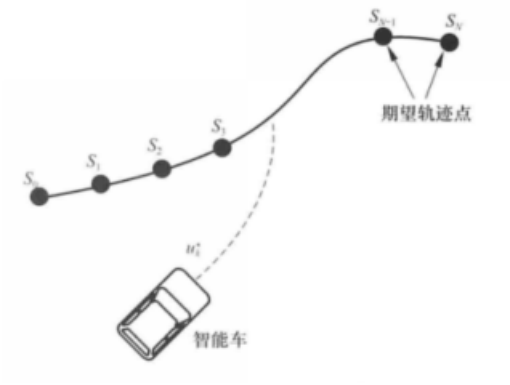

图 1 无人驾驶车辆轨迹跟踪示意图

根据研究内容，对无人驾驶车辆的轨迹跟踪问题做出如下限定：车辆为前轮转向车辆，而且系统能够有效地提供两类信息：

1、可行驶区域的几何描述、路面特征以及路面摩擦系数

2、车辆位置及内部状态，包括纵横向速度、加速度、轮速等参数。

目前已有很多车辆轨迹跟踪控制的方法，其中的经典控制算法以横向的纯跟踪 (pure pursuit) 控制算法与纵向的 PID 控制算法结合为代表。随着状态空间理论和最优控制理论的发展，以模型预测控制(Model Predictive Control, MPC) 算法为代表的最优控制算法逐渐受到人们的青睐，并广泛应用于轨迹规划后的跟踪问题。

模型预测控制最明显的优点即为能够在控制过程中增加多种约束。无人车辆在低速时，车辆平台运动学约束影响较大，而随着速度的增加，动力学特性对运

动规划与控制的影响就越明显，带来多种模式的约束。这正是模型预测控制在无人车辆运动规划方面的应用优势。

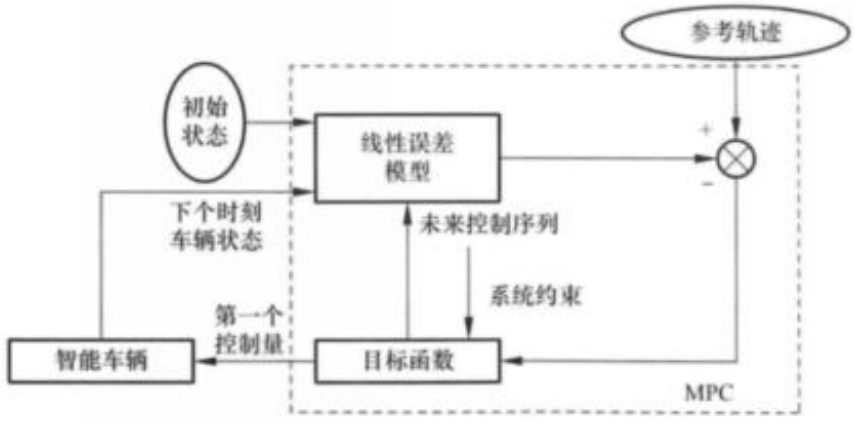

图 2 基于模型预测控制的轨迹跟踪控制器

轨迹跟踪过程中所采用的模型预测控制算法如上图所示。其中虚线框表示的是模型预测控制的主体，主要由线性误差模型、系统约束以及目标函数组成。线性误差方程是轨迹跟踪控制系统的数学描述，也是构建控制算法的基础。系统约束包括车辆执行机构约束、控制量平滑约束以及车辆稳定性约束等。

本文中我们将围绕模型预测控制算法，设计车辆轨迹跟踪算法。我们假设通过前项算法在得到各个自主车辆在规划时域N_p内的二维平面参考运动状态信息$X_r = [x_r, y_r, \theta_r, v_r]^T$和参考控制输入$u_r = [a_r, \delta_r]^T$后，将研究车辆的运动控制方法，以实现安全、快速通过的目标。

在第二章中，我们将会依据车辆运动学模型推导车辆控制的模型预测控制算法，并进行模拟。第三章中，我们将会进一步推导多车场景的模型预测控制算法，实现对场景中所有车辆的控制量进行计算，并实现避免碰撞的功能。第四章中，将会对报告内容做出总结。

二、算法实现

本章中，我们将会按照模型预测控制的算法的结构，从模型-预测-控制这三个方面，来详细阐述我们的算法。

2.1 模型

首先，我们基于小车运动的物理规律建立小车的运动模型。小车可以简化为自行车模型，由后轮提供速度 v，前轮提供偏转角 δ。设小车的横纵坐标为 (x, y)，

与 x 轴夹角为 θ，车辆轴间距为 l，运动学模型如下：

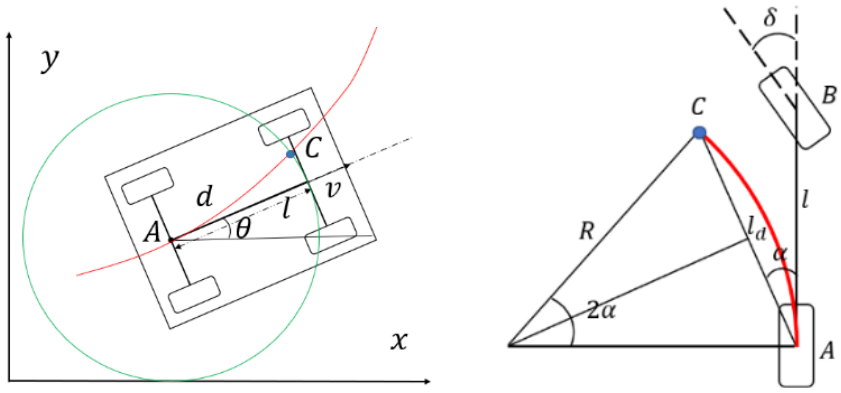

图 3 车辆运动学模型

$$\begin{cases} \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta \\ \dot{\theta} = \frac{v}{l}\tan\delta \end{cases} \tag{1}$$

上式为非线性模型，需要局部线性化。设$\xi = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$，$u = \begin{bmatrix} v \\ \delta \end{bmatrix}$，上式可表示为 $\dot{\xi} = f(\xi, u)$。

在 (ξ_r, u_r) 处进行一阶泰勒展开，可得：

$$\dot{\tilde{\xi}} = f(\xi_r, u_r) + \frac{\partial f}{\partial \xi}(\xi - \xi_r) + \frac{\partial f}{\partial u}(u - u_r) \tag{2}$$

令$\tilde{\xi} = \xi - \xi_r$，$\tilde{u} = u - u_r$，则有

$$\dot{\tilde{\xi}} = \dot{\xi} - \dot{\xi}_r = \frac{\partial f}{\partial \xi}\tilde{\xi} + \frac{\partial f}{\partial u}\tilde{u} = \begin{bmatrix} 0 & 0 & -v\sin\theta \\ 0 & 0 & v\cos\theta \\ 0 & 0 & 0 \end{bmatrix}\tilde{\xi} + \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ \frac{\tan\delta}{l} & \frac{v}{l\cos^2\delta} \end{bmatrix}\tilde{u}$$

$$= A\tilde{\xi} + B\tilde{u} \tag{3}$$

这是线性化的连续模型，需将其转换为离散模型。设时间步长为 T，则：

$$\tilde{\xi}(k+1) = (I + TA)\tilde{\xi}(k) + TB\tilde{u}(k) \tag{4}$$

令$\tilde{A} = (I + TA)$，$\tilde{B} = TB$，则有

$$\tilde{\xi}(k+1) = \tilde{A}\tilde{\xi}(k) + \tilde{B}\tilde{u}(k) \tag{5}$$

2.2 预测

基于以上的离散模型，可以依据接下来一段时间的输入来预测车辆的状态。

$\tilde{\xi}(k+1) = \tilde{A}\tilde{\xi}(k) + \tilde{B}\tilde{u}(k)$

$\tilde{\xi}(k+2) = \tilde{A}\tilde{\xi}(k+1) + \tilde{B}\tilde{u}(k+1) = \tilde{A}^2\tilde{\xi}(k) + \tilde{A}\tilde{B}\tilde{u}(k) + \tilde{B}\tilde{u}(k+1)$

$\tilde{\xi}(k+3)=\tilde{A}\tilde{\xi}(k+2)+\tilde{B}\tilde{u}(k+2)=\tilde{A}^3\tilde{\xi}(k)+\tilde{A}^2\tilde{B}\tilde{u}(k)+\tilde{A}\tilde{B}\tilde{u}(k+1)+\tilde{B}\tilde{u}(k+2)$

.....

$\tilde{\xi}(k+n)=\tilde{A}^n\tilde{\xi}(k)+\sum_{i=1}^{n}\tilde{A}^{i-1}\tilde{B}\,\tilde{u}(k+i-1)$

由 k 时刻状态预测得到的 $k+1$ 到 $k+n$ 时刻状态：

$$\begin{bmatrix}\tilde{\xi}(k+1)\\ \tilde{\xi}(k+2)\\ \cdots\\ \tilde{\xi}(k+n)\end{bmatrix}=\begin{bmatrix}\tilde{A}\\ \tilde{A}^2\\ \cdots\\ \tilde{A}^n\end{bmatrix}\tilde{\xi}(k)+\begin{bmatrix}\tilde{B} & 0 & \cdots & 0\\ \tilde{A}\tilde{B} & \tilde{B} & \cdots & 0\\ \tilde{A}^2\tilde{B} & \tilde{A}\tilde{B} & \cdots & 0\\ \cdots & \cdots & \cdots & \cdots\\ \tilde{A}^{n-1}\tilde{B} & \tilde{A}^{n-2}\tilde{B} & \cdots & \tilde{B}\end{bmatrix}\begin{bmatrix}\tilde{u}(k)\\ \tilde{u}(k+1)\\ \tilde{u}(k+2)\\ \cdots\\ \tilde{u}(k+n-1)\end{bmatrix} \tag{6}$$

将上式简写成如下：

$$\Xi=\Psi\tilde{\xi}(k)+\Theta\tilde{U} \tag{7}$$

在车辆运行的过程中，给定一条期望轨迹，即一系列期望的状态点：

$$\tilde{\xi}_r(k+1),\ldots,\tilde{\xi}_r(k+n)$$

这些期望点由 $\xi_r(k)$ 计算得到，并符合车辆运动模型 $\tilde{\xi}_r(k+j+1)=\tilde{A}\tilde{\xi}_r(k+j)$。

令$\Xi_r=\begin{bmatrix}\tilde{\xi}_r(k+1)\\ \tilde{\xi}_r(k+2)\\ \cdots\\ \tilde{\xi}_r(k+n)\end{bmatrix}$，则有：

$$\Xi_r=\Psi\tilde{\xi}_r(k) \tag{8}$$

设定一个损失函数 $J(k)$

$$J(k)=\sum_{j=1}^{n}\{[\tilde{\xi}(k+j)-\tilde{\xi}_r(k+j)]^T Q[\tilde{\xi}(k+j)-\tilde{\xi}_r(k+j)]+\tilde{u}^T(k+j-1)R\tilde{u}(k+j-1)\}$$

其中前一项代表预测轨迹点与期望轨迹点的差异，后一项代表输入控制量的幅值。该损失函数可转换为如下的矩阵形式：

$$J(k)=(\Xi-\Xi_r)^T Q_e(\Xi-\Xi_r)+\tilde{U}^T R_e\tilde{U} \tag{9}$$

其中 $Q_e=\mathrm{diag}([Q,Q,\ldots,Q])$，$R_e=\mathrm{diag}([R,R,\ldots,R])$（由 n 个 Q 矩阵（或 R 矩阵）构成的分块对角阵）

由 (6)、(7)可得：

$$E = \Psi\tilde{\xi}(k) - \ \Psi\tilde{\xi}_r(k) = \ \Xi - \Theta\tilde{U} - \Xi_r = \Xi - \Xi_r - \Theta\tilde{U}$$

$$\Xi - \Xi_r = E + \Theta\tilde{U} \tag{10}$$

将(9)带入(8)得：

$$J(k) = \left(E + \Theta\tilde{U}\right)^T Q\left(E + \Theta\tilde{U}\right) + \tilde{U}^T R\tilde{U}$$

$$= E^T QE + \left(\Theta\tilde{U}\right)^T Q\left(\Theta\tilde{U}\right) + 2E^T Q\left(\Theta\tilde{U}\right) + \tilde{U}^T R\tilde{U}$$

由于 $E^T QE$ 与 $\tilde{U}$ 无关，因此可以忽略。最终该损失函数可转化为如下的二次规划标准型：

$$minJ(\xi, u(k-1), \tilde{U}) = \tilde{U}^T(\Theta^T Q\Theta + R)\tilde{U} + (2E^T Q\Theta)\tilde{U}$$

$$s.t. \tilde{U}_{\min} \le \tilde{U} \le \tilde{U}_{\max}$$

$$U_{\min} \le U_t + A\tilde{U} \le U_{\min}$$

其中 $U_t = 1_n \otimes u(k-1)$，$A = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & \dots & 1 & 1 \end{bmatrix} \otimes I$

2.3 控制

基于预测部分得到的使损失函数 J 最小的控制序列 $\tilde{U}$，选取该序列中第一个时刻对应的控制量发送给车辆执行，然后用传感器测出车辆运动后更新的 x, y, θ，再送入预测环节中，求出下一时刻的 $\tilde{U}$。如此循环即为模型预测控制的全过程。

2.4 仿真

按照以上的算法流程，我们在 python 环境下，实现了该算法，并且通过仿真验证了算法的性能。

车辆初始位置为 (0，0)，朝向角 $\frac{\pi}{3}$，参考路径为直线：$0 \le x \le 5, y = 2$

Python 环境下的运行结果如上所示。我们可以看出，该算法可以实现对参考路径较好的跟踪。

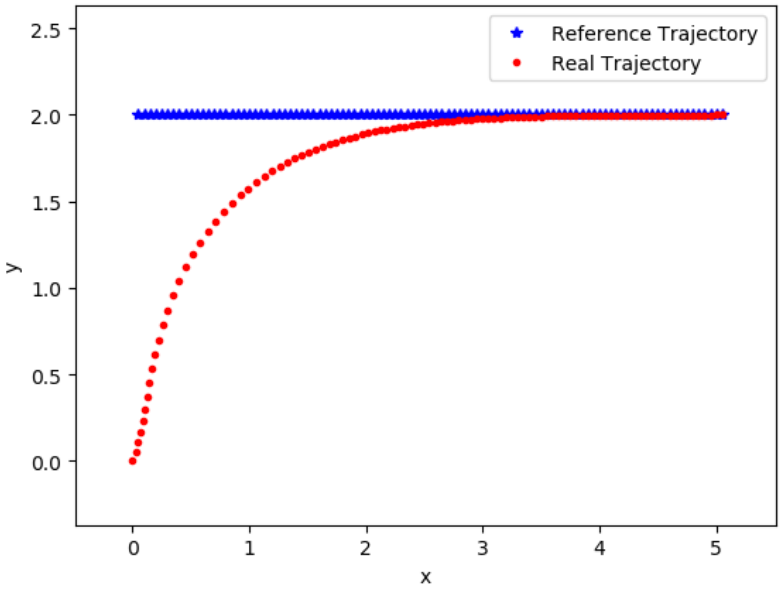

图 4 仿真结果

三、算法改进

在本章中，我们将会在第二章实现的内容的基础上，对算法进行进一步的优化。在多车协同工作的场景下，实现对多车的控制量进行计算。

3.1 设计思路

本章中的多车协同控制算法的设计思路来源于前人对多无人机编队控制的研究中。在文献[1]中，Kentaro Akiyama 等人在多无人机编队控制中使用分布式模型预测控制，使得无人机编队能够克服干扰因素并保持队形沿着期望路径前进。我们的多车协同控制算法即由此演变而来。本节中，我们将会对他们的工作做一个简要的叙述。

首先建立单个无人机的线性化模型，其中 $\hat{d}_y$ 为扰动：

$$x_i[k+1] = A_m x_i[k] + B_m u_i[k] + D_m \hat{d}_y$$

系统中所有无人机都会共享其基于 k 时刻预测的接下来一段时间的状态：

$$h_i[k] = [x_i[k], \hat{x}_i[k+1|k], \hat{x}_i[k+2|k], \dots, \hat{x}_i[k+N|k]]$$

有了其他无人机的预测状态，每一个无人机就可以自己单独做二次规划，计算最优控制输入。对于无人机 i，其损失函数为：

$$J_i = \sum_{k=0}^{N-1} (\eta(x_i[k], h_1, \dots, h_n) + L(x_i[k], u_i[k]))$$

其中第一部分的 η 为描述无人机 i 与其他无人机的相对位置与参考相对位置之差的二次型：

$$\eta(x_i, h_1, \dots, h_n) = \sum_{j=1(j\neq i)}^{n} \left(x_i - h_j - r_{ij}\right)^T Q_a \left(x_i - h_j - r_{ij}\right)$$

第二部分的 L 即为一般的 MPC 中与参考路径的状态差以及输入幅值：

$$L(x_i, u_i) = (x_i - t_i)^T Q (x_i - t_i) + u_i^T R u_i$$

其中 r_{ij} 为无人机 i 与 j 之间期望保持的位置关系，t_i 为无人机 i 的期望路径。

我们参考该文章中引入无人机之间距离来保持队列距离，同样可以引入车车距离来做避碰。

3.2 损失函数优化

为了扩展模型预测控制，实现在多车辆协同运动的情况下的避免碰撞功能，需要对损失函数进行优化。

在单车的模型预测控制中，损失函数为：

$$J(k) = \sum_{t=k}^{k+T} [(X - X_r)^T Q (X - X_r) + \tilde{u}^T R \tilde{u}]$$

在多车的场景下，需要考虑车车之间的距离问题，尽量避免车车间距过小的情况出现。为此，需要增加定义一个用于描述车车距离的损失函数，该损失函数随着两车间距的缩小而增大。

可以设计出两类损失函数，设 X 为当前车辆状态（即位置），X_i 为任一其他车辆的状态。第一类是将两车间距放在分母上，例如：

$$J_d(k) = \frac{1}{(X - X_i)^2 + \varepsilon}$$

$(X - X_i)^2$ 即为两车间距的平方，ε 是一个非常小的正数，仅用来保证分母不为 0。

第一类从逻辑来说比较简单，是在车辆正常行驶的时候尽可能远离其他车辆；但由于其 X 在分母上，将其转换为与输入 $\tilde{u}$ 相关的函数时，得到的结果中还包含了除一次、二次项以外的其他项，在优化计算时无法使用二次规划，需要使用其他的凸优化方法。

第二类是将两车间距放在分子上，但对其取负号，例如：

$$J_d(k) = \begin{cases} 0, & |X - X_i| > d_T \\ \sum_{t=k}^{k+T} [d_T^2 - (X - X_i)^2], & |X - X_i| \le d_T \end{cases}$$

d_T 为距离的阈值。第二类在执行时需要对车辆间距进行判断，当两车间距大于阈值 d_T 时不考虑这部分的损失函数，各走各的；当两车间距小于d_T 时，将 d_T^2 与两车距离平方做差值作为描述车辆间距的损失函数加入到原始的损失函数中。因此，当 $|X - X_i| \le d_T$ 时，损失函数即为：

$$J(k) = \sum_{t=k}^{k+T} [(X - X_r)^T Q (X - X_r) + d_T^2 - (X - X_i)^T P (X - X_i) + \tilde{u}^T R \tilde{u}]$$

其中 P 可定义为$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$，在这里我们只考虑两车之间的位置，而不考虑朝向。

原始的单车损失函数为：

$$J(k) = \sum_{t=k}^{k+T} [(X - X_r)^T Q (X - X_r) + \tilde{u}^T R \tilde{u}]$$

将其写成矩阵形式：

$$J(k) = (\Xi - \Xi_r)^T Q_e (\Xi - \Xi_r) + \tilde{U}^T R_e \tilde{U}$$

其中 $Q_e = \mathrm{diag}([Q, Q, \dots, Q])$，$R_e = \mathrm{diag}([R, R, \dots, R])$（由 n 个 Q 矩阵（或 R 矩阵）构成的分块对角阵），$\Xi - \Xi_r = E + \Theta \tilde{U}$，$E$ 为 n 步内车辆与参考轨迹的预测误差，因此有

$$J(k) = \left(E + \Theta \tilde{U}\right)^T Q_e \left(E + \Theta \tilde{U}\right) + \tilde{U}^T R_e \tilde{U}$$

$$= E^T Q E + \left(\Theta \tilde{U}\right)^T Q_e \left(\Theta \tilde{U}\right) + 2E^T Q_e \left(\Theta \tilde{U}\right) + \tilde{U}^T R_e \tilde{U}$$

$$= \tilde{U}^T (\Theta^T Q_e \Theta + R) \tilde{U} + (2E^T Q_e \Theta) \tilde{U}$$

当 $|X - X_i| \le d_T$ 时，车距部分的损失函数为

$$J_d(k) = \sum_{t=k}^{k+T} [d_T^2 - (X - X_i)^T P (X - X_i)]$$

将车距部分的损失函数写成矩阵形式：

$$J_d(k) = D_T - (\Xi - \Xi_i)^T P_e (\Xi - \Xi_i) = D_T - \left(E_c + \Theta \tilde{U}\right)^T P_e \left(E_c + \Theta \tilde{U}\right)$$

其中 $P_e = \mathrm{diag}([P, P, \dots, P])$，$E_c$ 为当前车辆 X 与其他车辆 X_i 在 n 步内

预测状态之差。对 $J_d(k)$ 展开：

$$J_d(k) = D_T - \left(E_c + \Theta\tilde{U}\right)^T P_e\left(E_c + \Theta\tilde{U}\right)$$
$$= D_T - E_c^T P_e E_c - \tilde{U}^T(\Theta^T P_e\Theta)\tilde{U} - (2E_c^T P_e\Theta)\tilde{U}$$

因为 $D_T - E_c^T P_e E_c$ 与 $\tilde{U}$ 无关，可以忽略，所以最终车距的损失函数为：

$$J_d(k) = -\tilde{U}^T(\Theta^T P_e\Theta)\tilde{U} - (2E_c^T P_e\Theta)\tilde{U}$$

整体的损失函数为：

$$J(k) = \tilde{U}^T(\Theta^T Q_e\Theta + R)\tilde{U} + (2E^T Q_e\Theta)\tilde{U} - \tilde{U}^T(\Theta^T P_e\Theta)\tilde{U} - (2E_c^T P_e\Theta)\tilde{U}$$
$$= \tilde{U}^T(\Theta^T Q_e\Theta - \Theta^T P_e\Theta + R)\tilde{U} + (2E^T Q_e\Theta - 2E_c^T P_e\Theta)\tilde{U}$$

3.3 仿真实验

按照前一节所述，我们将会在仿真环境中对该算法改进的功能进行验证。

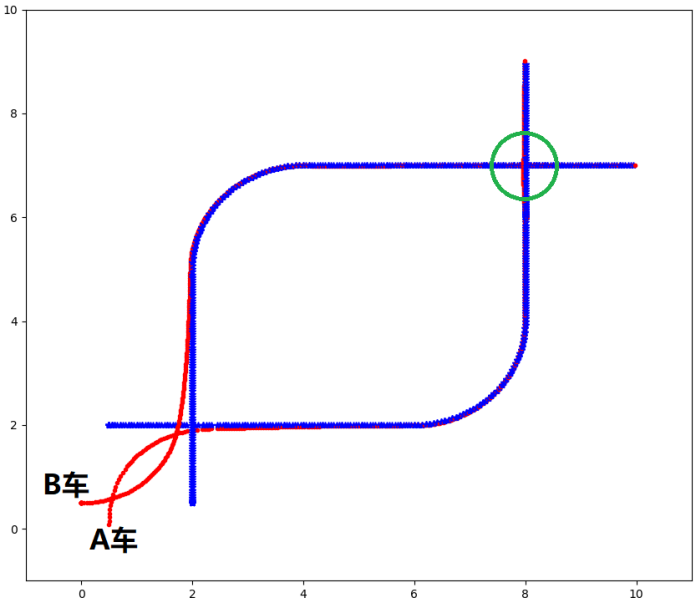

图 5 仿真路径图

车 A 的起点为 $(0.5, 0)$，朝向角 $\frac{\pi}{2}$，跟踪轨迹①(0，2)到(6，2)的直线；②(6，2)到(8，4)的四分之一圆弧；③(8，4)到(8，9)的直线。

车 B 的起点为 $(0, 0.5)$，朝向角 0，跟踪轨迹①(2，0)到(2，5)的直线；②(2，5)到(4，7)的四分之一圆弧；③(4，7)到(10，7)的直线。

两条十字交叉的路径为期望轨迹。若两车各自按基本的车辆模型预测控制运行则必然相撞。若设置两车间距的阈值 $d_T = 1$，让两车按照第二类损失函数运行，可以实现两车的避让功能。但这种方法在避让时似乎更倾向于调整速度而非调整方向，当有相遇的可能性时，一方会停车或倒退以进行避让，而非改变方向

绕行。例如在起点处，两车距离为$\frac{\sqrt{2}}{2}\mathrm{m} < 1\mathrm{m}$，此时执行的结果是 B 车先在原地停车，等 A 车与自己保持足够距离后再出发；在(8，7)附近（圆位置），在接近时 A 停车等待 B 通过再前进。相当于面这段直线的速度被减小了，以让两辆车相遇，A 会等 B 完全通过后再通行，而非在 B 前进的过程中进行绕行。

四、总结

本文基于模型预测控制算法，设计了适用于无人驾驶车辆轨迹跟踪的控制算法。在仿真环境下实现了对车辆的轨迹跟踪控制。并且进一步的拓展了多车场景下的模型预测控制，实现车辆行驶之间避免碰撞的功能，在仿真环境下成功验证了可行性。

五、参考文献

[1] Kentaro Akiyama, et. al., Robust Formation Control Applying Model Predictive Controlto Multi Agent System by Sharing Disturbance Information with UAVs, SICE Annual Conference, 627-632, 2016.

[2] HengYang Wang, et. al., Path Tracking Control for Autonomous Vehicles Based on an improved MPC, IEEE.

[3] S. Chao, S. Yangm and B. Brad, Model Predictive Control for an AUV with Dynamic Path Planning, The Society of Instrument and Control Engineers Annual Conference, SICE, 2015.

[4] N. Thaker, N. George and G. Thomas, A full error dynamics switching modeling and control scheme for an articulated vehicle, International Journal of Control, Automation and Systems,Vol.13, No.5, 1221-1232, 2015.