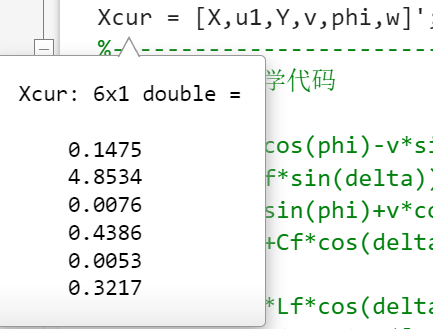
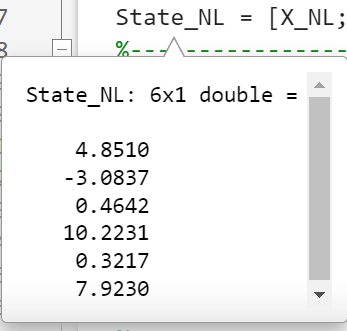
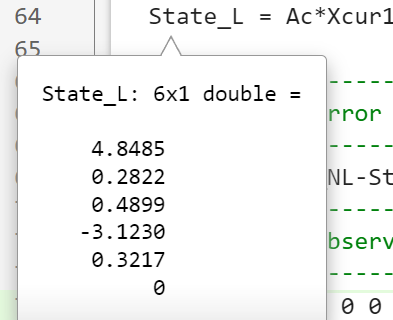
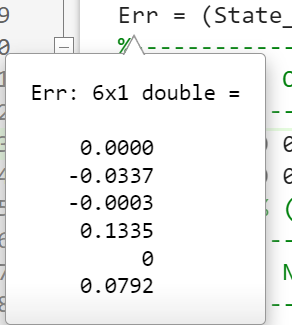
已知Xcur和Xcur1完全一样吧。。不看Xcur1了吧。。。

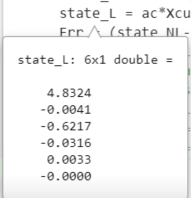
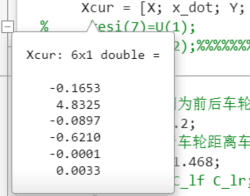


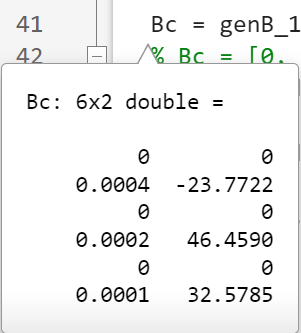
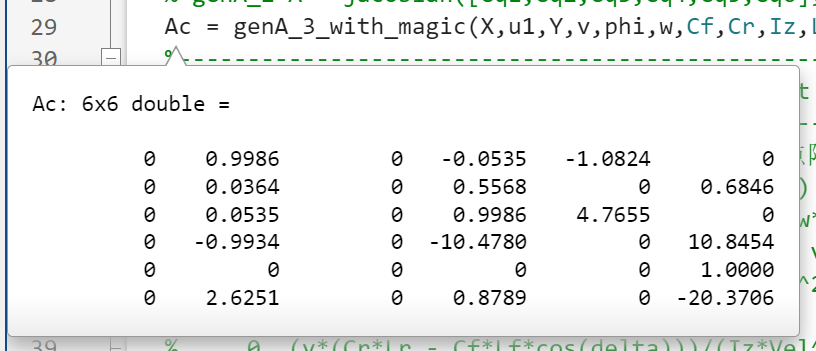
工况：初始5m/s。位置0,0，朝向0.目标[0 10 2 0 0 0]。速度10，Y=2

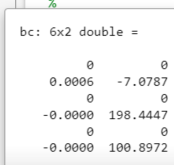
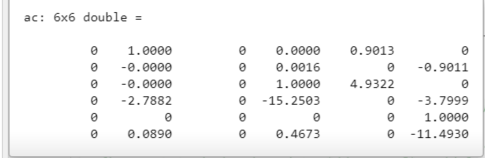
先旧的：

再新的：



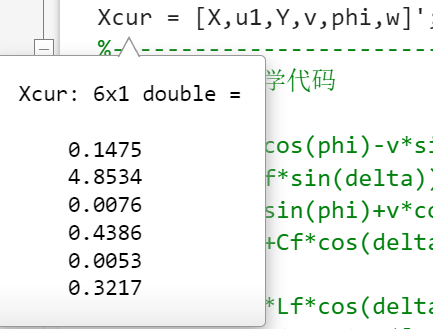
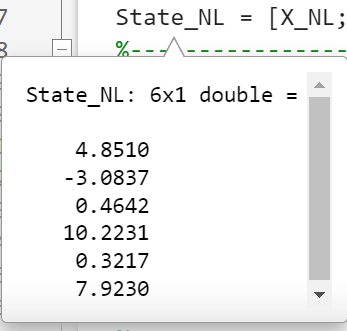
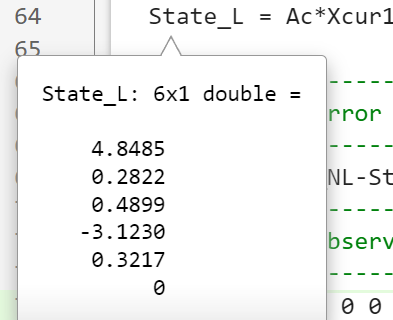
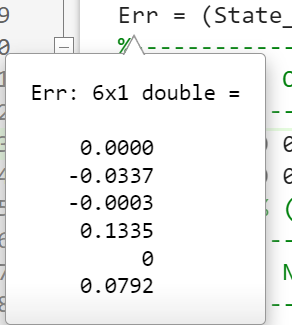




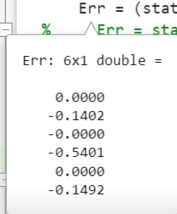
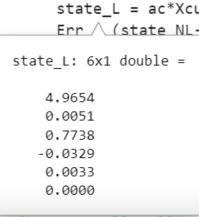
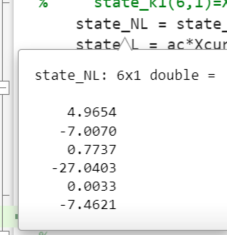
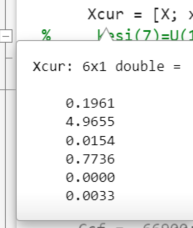
Debug过程中发现问题在于这不是差值的MPC，输出的控制量直接就是绝对值！

改后重新算数：

旧的：

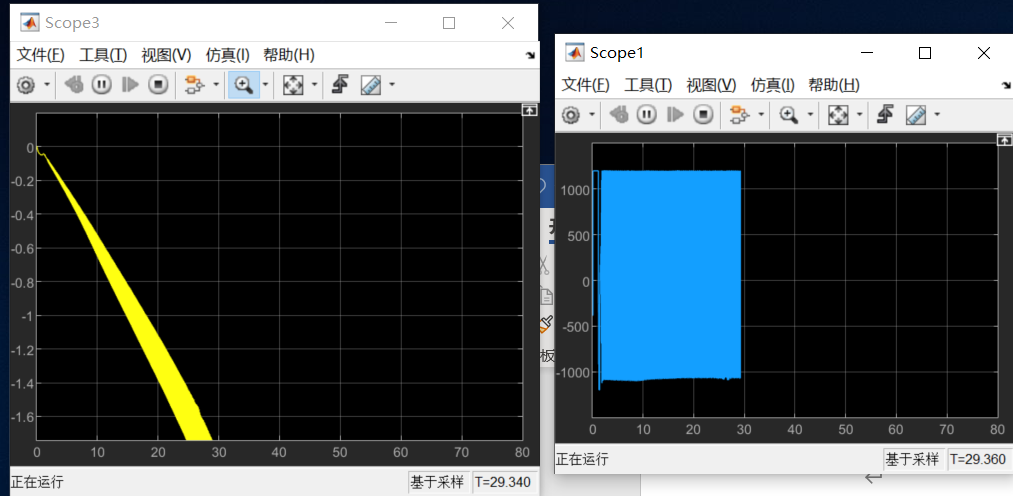
新的：



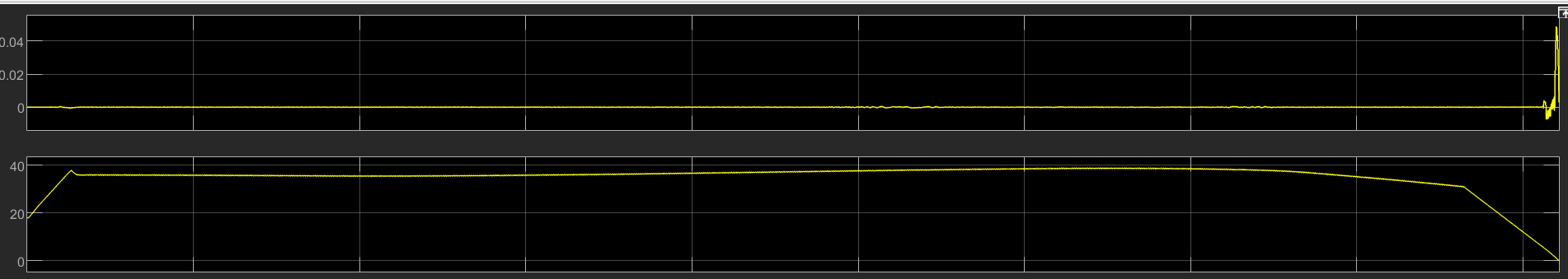
尝试通过调参实现：

Q第一项是对纵向车速的跟踪。Q1（速度误差的权重）大或R1小，加快响应。（R1是驱动力的代价函数，越大则求解器会倾向于输出较小的驱动力）

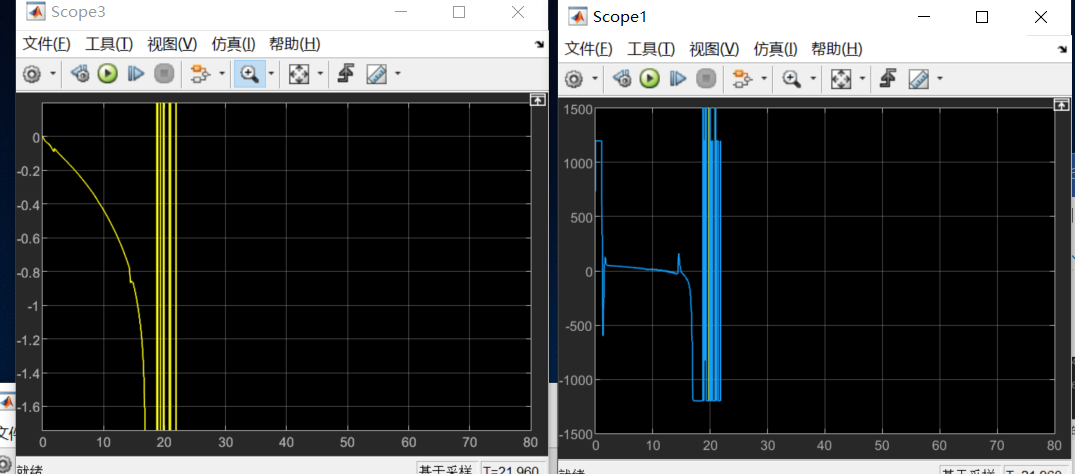
一般先给控制量加一个较小的权重（R）

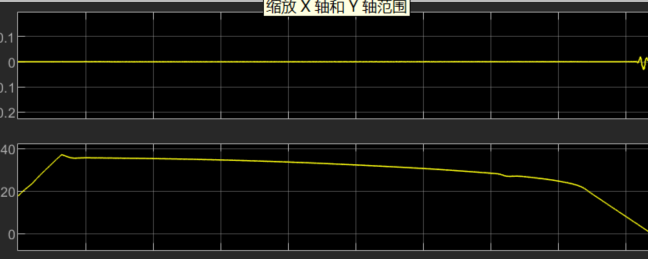
****

q = [1e3 1e5]; Rr = [1e-6 1e6];

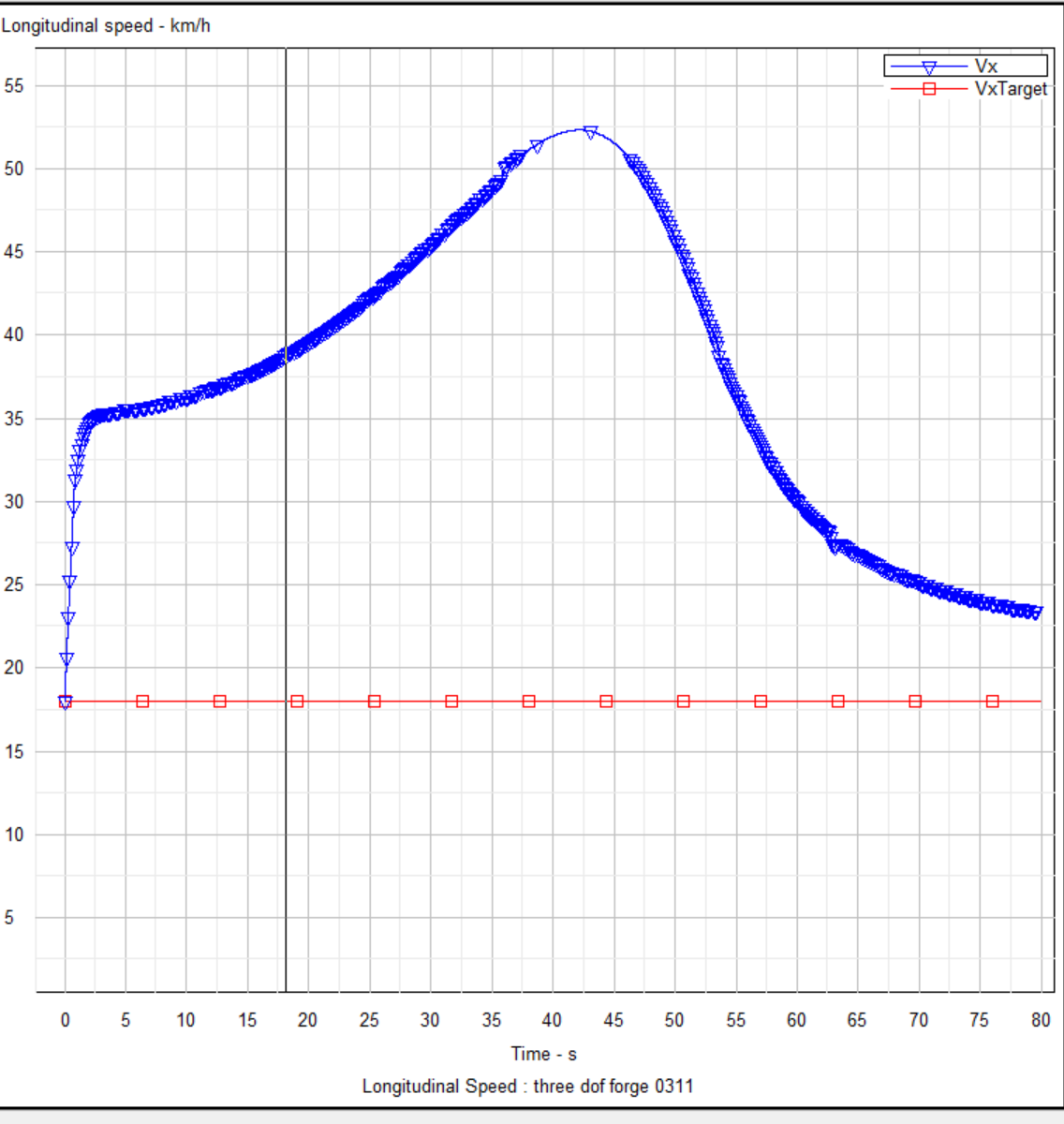


q = [1e2 1e5]; Rr = [1e-6 1e6];





q = [1e1 0]; Rr = [1e-6 1e6];



好

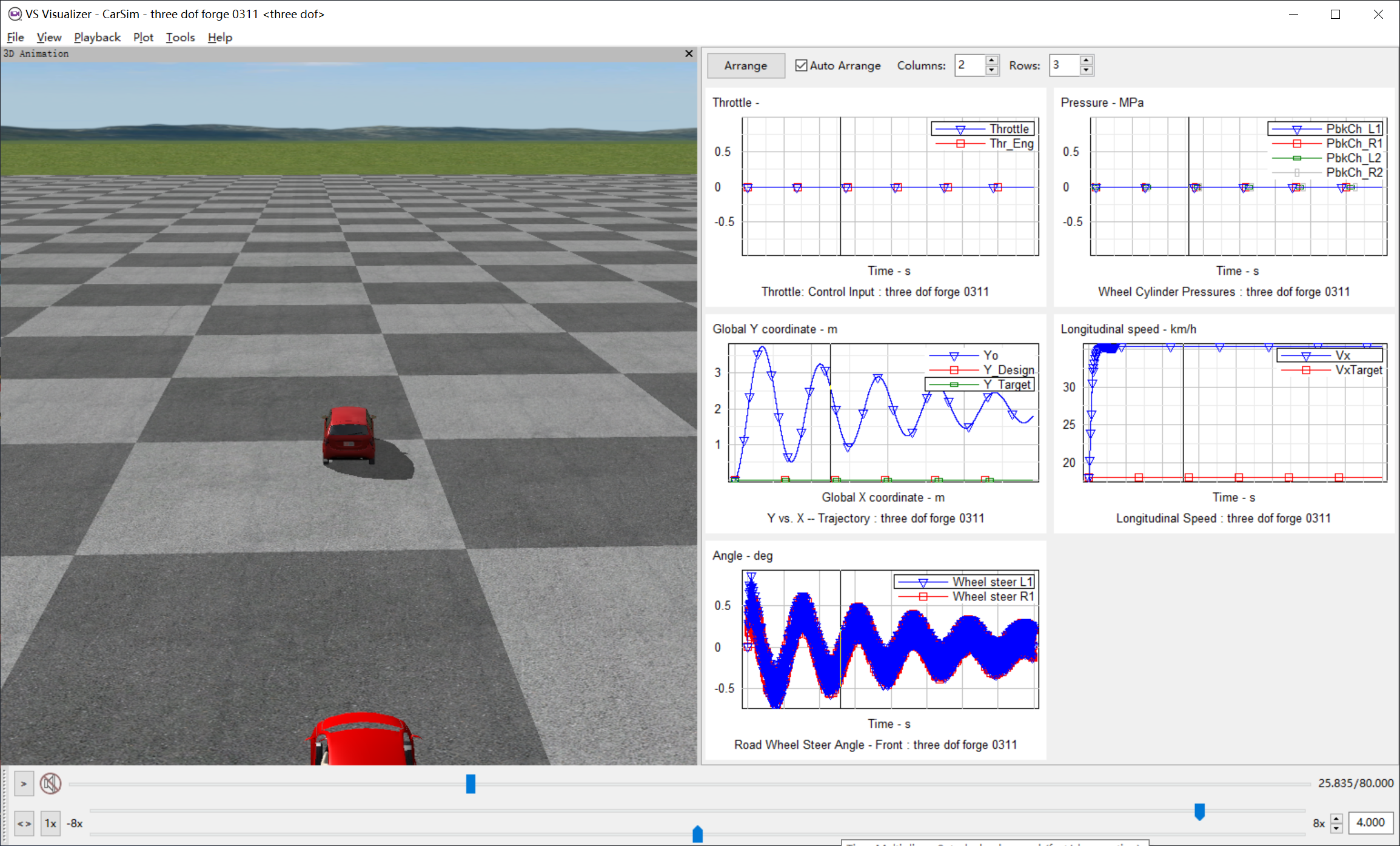
【0312】实现效果：

q = [1e1 1e1]; Rr = [1e-6 1e3];

RefSt = [0 10 2 0 0 0]';

Ccf = -66900;Ccr = -62700;Clf = -66900;Clr = -62700;

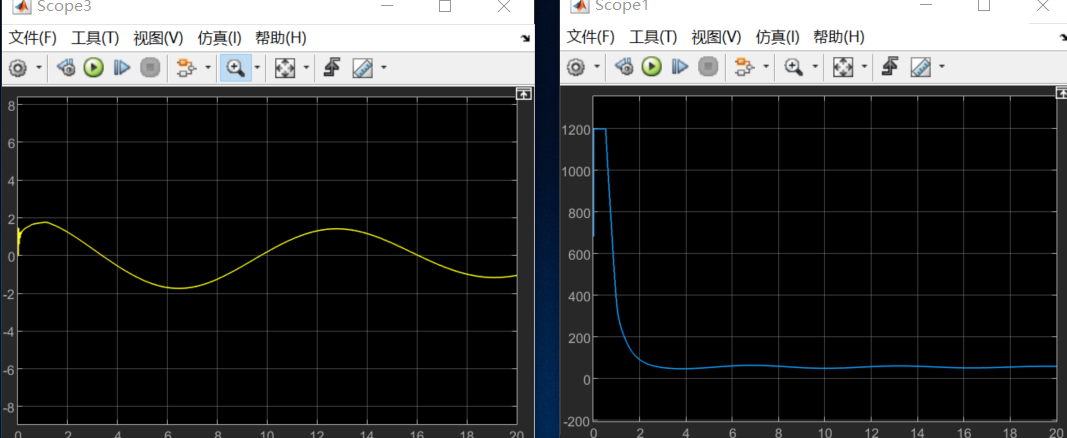
车速跟踪很好，位置跟踪大概对

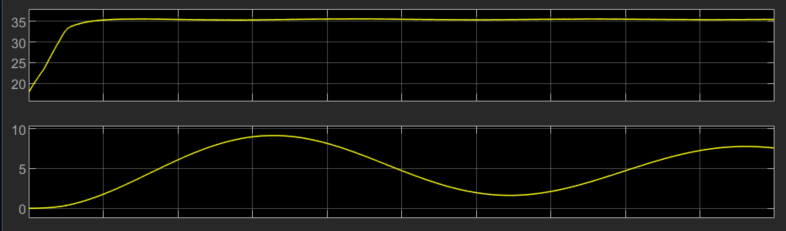


0312\_y\_u\_30km+y2\_1.mat

RefSt = [0 10 5 0 0 0]';

q = [10 10]; Rr = [1e-6 1e3];



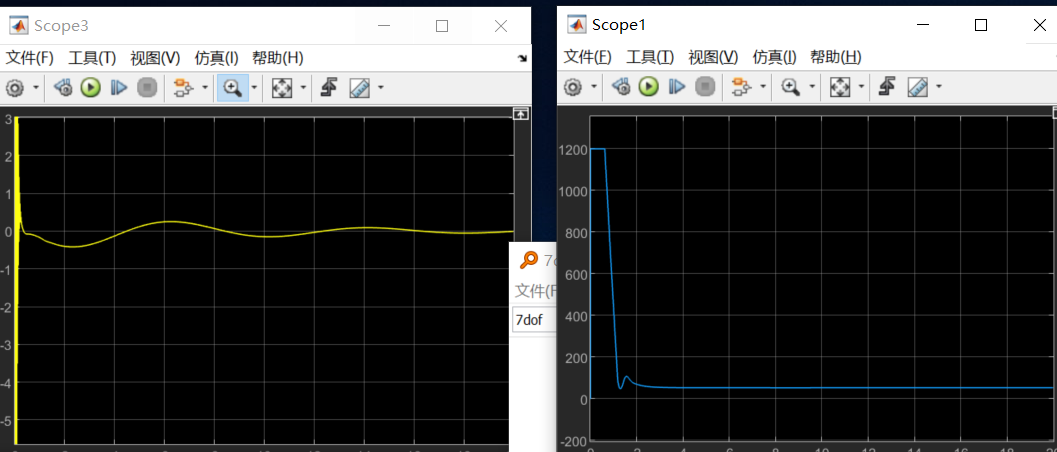


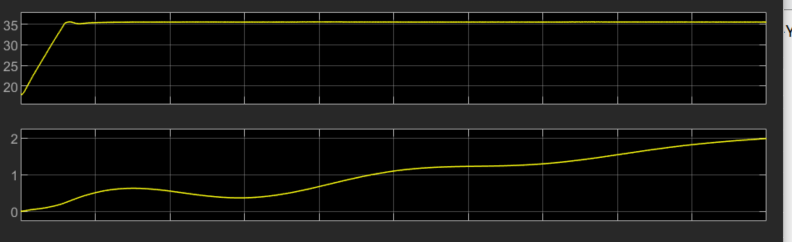
然后又搞了个新的：

RefSt = [0 10 2 0 0 0]';%参考状态 R efSt R efSt = [0 10 0 0 0 0]';

RefSt(2) = 10;

RefSt(3) = 0.1\*t;





0312\_y\_u\_changing\_with\_t\_notgood\_2.mat

（注意这是在Carsim/PiaoYi的包下。文件在D:\Carsim\20200424mpcforselfdrivingvehiclesV4\chap5 Matlab Code\forge0311）

【0312】下午，找不到问题，先这样吧。

下一步思路：

1. 七自由度车辆动力学，或者说能直接跟踪
2. 研究一下障碍物条件下的。或者可能是变化参数的

七自由度：

syms

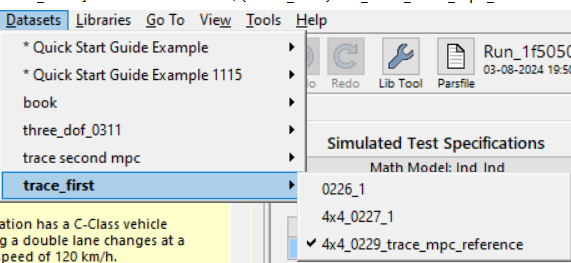
eq;

vars = [dXb; dYb; dYaw; domegaFr; domegaFl; domegaRr; domegaRl; s; n; chi; Xb; Yb; Yaw];

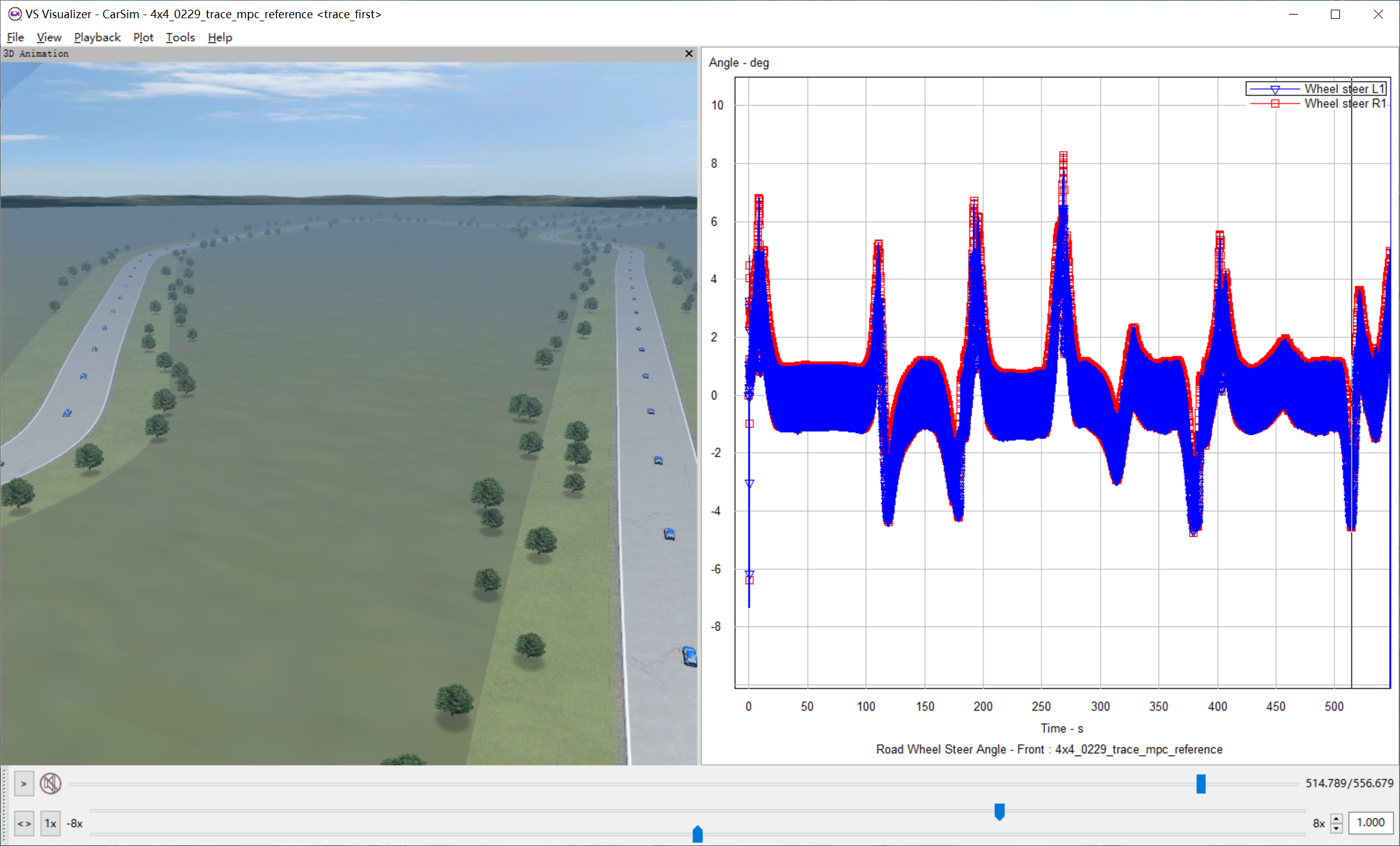
ddXb= Fx/pusr.mt+dYaw.\*dYb;

【0313】下午，先总结一下现在有什么

首先，1MPC路径跟踪。



D:\Carsim\Distributed\_Drive\_Matlab\carsim\trace\_mpc\_0301\_Nurburgring\mpc\_trace\_Nurburgring.mdl



然后。下午研究那个障碍物约束的路径规划

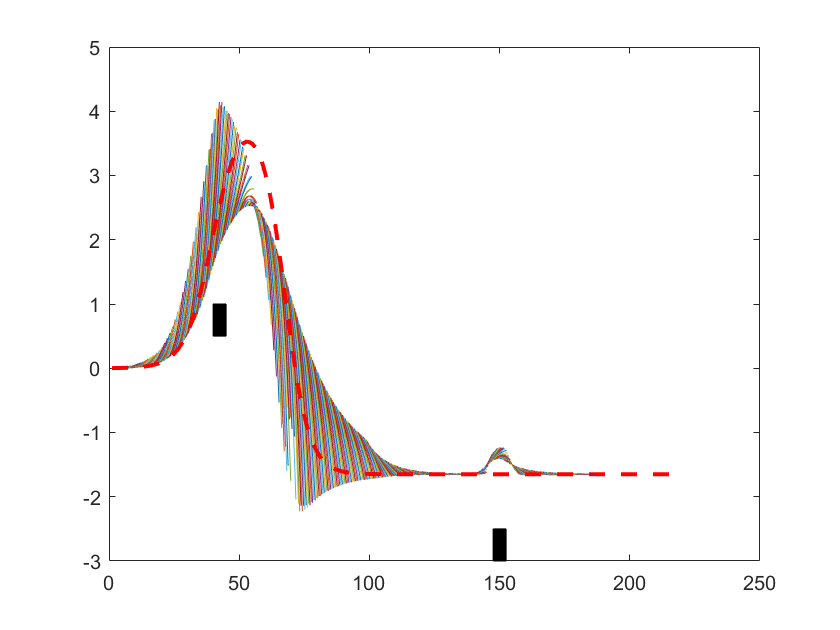
下午没研究明白。。。然后3月13日就过完了，3月14开始研究。

【0314】3.14基本搞明白了障碍物约束的路径规划原理：非线性MPC，优化代价函数为Q矩阵的误差+R矩阵的代价+B矩阵的障碍距离的倒数。

优化对象是ay，也就是横向加速度。具体是ay的两个值，因为控制时域是2。

【0315】（白天不在学校）

考虑像Optimization-Based Hierarchical Motion Planning for Autonomous Racing一样，给一个车速或者时间的倒数作为被优化量。

别人发现的问题：避障代价函数把所有东西都当做障碍物。改进：把障碍物距离改成分段函数。

改为：

从而代价函数：