同济大学

TONGJI UNIVERSITY

毕业设计(论文)

课题名称	无人驾驶闭环控制仿真平台的开发
副标题	
学　院	电子与信息工程学院
专　业	通信工程
学　号	1351659
学生姓名	刘世鹏
指导教师	陈耀　薛亚歌
日　期	2020-06-05

无人驾驶闭环控制仿真平台的开发

装 订 线

摘 要

随着人工智能的发展，无人驾驶汽车因其良好的前景而成为了当前的研究热点，但无人驾驶汽车的昂贵性和实地测试的危险性极大的限制了无人驾驶车辆控制算法的开发和测试，因此，一个可信度高并且易于使用的仿真平台对于控制算法的研发具有重大的意义。本课题基于开源仿真平台 Carla（Car Learning to Act）的虚拟环境开发了专用于车辆控制的控制仿真平台（CSVC），改进了横向距离误差和横摆角误差耦合的滑模控制算法，并尝试使用自适应动态规划改善其跟踪精度，最后对 CSVC 仿真平台做了全面的评估，并在仿真平台中分析验证了改进的控制算法。

CSVC 无人驾驶控制仿真平台在原有 Carla 模块的基础上添加了新的虚拟仿真场景、校准了车辆的动力学模型参数、改进了底盘控制器等；除了 Carla 原有的模块，本文还设计了具有三种曲线拟合算法的轨迹规划模块，使用户能够通过可视化操作生成不同的仿真轨迹；开发了包括仿真动画效果显示和仿真数据可视化工具在内的可视化模块；最后利用机器人操作系统 ROS 将控制模块、轨迹规划模块、可视化模块以及 Carla 模块相连。

针对无人驾驶轨迹跟踪问题，在车辆运动学的基础上设计并改善了无人驾驶车辆的横向控制策略。本文系统地讨论了滑模控制理论及其优势，并在此基础上改进了一种横向距离误差和横摆角误差耦合的滑模控制算法，使其横向距离误差和横摆角误差都能在理论上收敛至零。最后，本文还尝试使用自适应动态规划（Adaptive Dynamic Programming）改进此控制算法，并设计了相应的评价网络结构和控制网络结构。

最后，本文从车辆动力学模型可信度、底盘控制器性能、轨迹规划模块效果三个方面分别验证了仿真平台的可信度和可靠度。校准后的动力学模型十分接近现实车辆，底盘控制器表现良好，能够精确的跟踪速度命令，轨迹规划模块能十分便捷的生成三种不同的轨迹曲线。本文在确保仿真平台的可信度的基础上验证了改进的滑模控制算法相较于 MPC 控制算法的优越性，并证明了使用自适应动态规划学习算法来改进滑模控制算法是可行的。

关键词：无人驾驶汽车，滑模控制，自适应编程控制，动力学模型，ROS

Development of a Closed-loop Simulator for Unmanned Vehicle Control Based on Carla

装 订 线

ABSTRACT

With the rapid development of artificial intelligence, self-driving cars have gradually become a research hotspot. However, the expensiveness of self-driving cars and the insecurity of field testing greatly limit the test of control algorithms. A convenient and reliable simulator is essential for researchers to develop control algorithms. This paper is aimed to develop a Close-loop Simulator for Vehicle Control (CSVC) based on the virtual environment of open-source Simulator Carla. A sliding mode control algorithm coupling lateral error and orientation error is then developed and an adaptive dynamic programming method is used to improve the tracking accuracy. Finally, the CSVC simulation platform is evaluated comprehensively, and the proposed control algorithm is verified in the CSVC simulator.

We added a new virtual simulation scene on the basis of the Carla module, calibrated the dynamic model of the vehicle, and improved a new low-level control algorithm. Excepted the CARLA module, a trajectory plan module with three kinds of interpolation curve fitting method is designed, which enables users to simulate different trajectories by visual operation. The visualization module including simulation animation and simulation data is developed. The Robot Operating System, ROS is used to connect the control module, plan module, visualization module and Carla module.

For trajectory tracking problem, the control module was designed and implemented on the basis of kinematics of self-driving cars. This paper systematically discussed the sliding mode control theory and its advantages, and proposed a new sliding mode control algorithm coupling lateral error and orientation error, which promises the convergence of both lateral error and orientation error to zero. This paper also attempts to use Adaptive Dynamic Programing(ADP) to improve the accuracy of control algorithm, and designed the corresponding critic network structure and action network structure.

Finally, this paper evaluated and verified the credibility of the simulation platform from three aspects, vehicle dynamic model, low-level controller, and trajectory planning module. The vehicle dynamics model is very reliable after the calibration and the performance of low-level controller is good and it can accurately track the desired speed command. The trajectory planning module can be very convenient to for users to generate three different trajectory curves. The proposed sliding mode control algorithm is verified to be better when compared with MPC control algorithm and adaptive dynamic programming learning algorithm is proved to be effective to improve performance of the control algorithm.

Key words： self-driving cars, sliding mode control, adaptive dynamic programming, dynamic model, ROS

目 录

装

订

线

1 引言

1.1 研究背景及意义

自 1885 年德国工程师卡尔本茨研制出第一辆汽车以来，汽车已经成为人们出行的必备工具。它在给人类带来极大的出行便利的同时，也制造了数以万计的交通事故以及日益严重的交通拥堵。随着计算机，电子信息行业的高速发展，无人驾驶（Autonomous Driving）这一概念应运而生。无人驾驶是指车辆通过自身携带的传感器感知环境信息，对信息进行进一步的处理后能做出自主决策，然后控制车辆按照规划的轨迹和决策命令完成驾驶任务。

根据国际自动机工程师学会（SAE）提出的标准可将无人驾驶划分为六个级别（L0-L5）[1]：

（1）L0 级人工驾驶：由人类驾驶者驾驶汽车

（2）L1 级辅助无人驾驶：有限场景下的横向或纵向控制的其中一项由车辆自动完成

（3）L2 级部分无人驾驶：有限场景下的横向或纵向控制的多项由车辆自动完成

（4）L3 级条件无人驾驶：有限场景下的绝大部分驾驶操作由车辆完成，需人类监督

（5）L4 级高度无人驾驶：有限场景下的所有驾驶操作由车辆完成，不需人类监督

（6）L5 级完全无人驾驶：不限场景下的所有驾驶操作由车辆完成，不需人类监督

无人驾驶的研究发展，将极大的改变未来人们的交通出行，不仅能减轻驾驶员的负担，而且更加的智能，舒适，安全。信息技术的不断发展更是会将车辆，人，交通设施都连接起来，达到更高效，更安全的交通调度效果，从而改善交通拥堵的现状，减少事故的发生率。因此无人驾驶的研究具有重大意义[2]。

实现完全的无人驾驶是人类由来许久的梦想，许多公司和机构已经在无人驾驶领域取得了巨大的进展，自 2004 美国国防高级研究计划局（DARPA）举办挑战赛[3]以来，一些著名大学已经开始研究无人驾驶，如斯坦福大学的 Stanley[4]，卡耐基梅隆的 BOSS 智能车都在挑战赛取得了很好的成绩。在 2006 年欧洲陆地机器人实验赛（European Land-Robot Trial, ELROB）上，德国智能车“途锐”取得了冠军。中国在无人驾驶领域起步较晚，在 2011 年，国防科技大学和一汽集团研发的红旗 HQ3 实现从长沙到武汉全程约 286km 的无人驾驶。放眼世界，无论是谷歌公司的无人驾驶车 Waymo，还是国内百度无人驾驶汽车都宣称实现了 L4 级别的无人驾驶，但是可靠性和道德伦理仍然需要长时间的验证和改善，L4 级别的无人驾驶尚未完全成熟[5]。

无人车造价十分昂贵，而且测试时容易发生事故，造成设备损毁，这使得无人驾驶的新型控制算法在测试和验证时遇到了极大的困难。一个新开发的算法直接在实车上进行测量是不现实的，仿真平台则较好的解决了这个问题。它具有良好的便捷性、安全性、经济性，而且便于重复实验和收集相关数据。搭建一个性能良好的仿真平台，为极限工况下无人驾驶控制算法的有效性和安全性验证以及车辆运动性能分析提供了代价低，性能优越的方法。因此，无人驾驶控制仿真平台的开发和优化非常具有实际意义。

1.2 现有无人驾驶仿真平台

现有的国内外无人驾驶仿真平台大致可分为三类：基于汽车动力学的仿真软件、基于机器人

的仿真软件、基于游戏引擎的软件。

1.2.1 基于汽车动力学的仿真软件

基于汽车动力学的仿真软件的优势在于对物理和汽车动力学的仿真非常的精准，但传感器和渲染效果较差。如 Mechanical Simulation Corporation 推出的汽车仿真软件 Carsim、专门仿真卡车等大型车辆的 Trucksim、MSC Software 推出的多体动力学仿真工具 ADAMS 等[6]。

Carsim 和 Trucksim 主要从整车角度进行仿真，其本质上通过提供的模型库、参数库来对车辆进行仿真，其自身已经自带了相当数量的模型，用户免去了繁杂建模和调参数的过程，只要调整车辆的特性参数即可进行仿真。这样的模式不仅提升了仿真速度，而且其精度也能满足一般的需求。Carsim 也能将仿真结果生成动画，但是其动画的渲染效果较差，场景不真实，而且它作为一款商业软件而非开源的仿真平台，这大大降低了它的使用范围。虽然用户也可以通过 Simulink 和 Carsim 联合仿真来测试控制算法的性能，但是却依旧不方便。

ADAMS 的主要功能是多体运动学建模和仿真，该仿真软件可以对汽车精确到各个组件的动力学仿真，但用户需要做的工作量十分巨大，而且大大的降低了仿真的速度，因此比较适合单独分析汽车某一组件的动力学如，汽车的悬架等。无人驾驶控制算法的验证并不需要如此精密的动力学仿真软件。

（a）Carsim　　（b）ADAMS

图 1.1 基于汽车运动学的仿真软件

1.2.2 基于机器人的仿真软件

基于机器人的仿真软件主要是针对各种机器人的专业软件，如 Gazebo，Webots 等。Gazebo 是 ROS 机器人系列中经常用到的 3D 机器人仿真软件，使用 ODE 物理引擎，能够在复杂的室内和室外环境中准确有效的模拟机器人，也支持目前流行的开源机器人框架 ROS，但其界面并不友好，操作起来比较复杂，建立渲染效果比较好的车辆模型的难度较高，而且缺乏一些无人驾驶相关的专业场景的支持[7]。这使得在 Gazebo 上开发一整套用于无人驾驶控制算法验证的场景的成本非常昂贵，并且达不到最佳效果，此外，Gazebo 对 windows 系统的兼容程度也不高。而 Webots 是一款老牌的商业软件，尽管最近已经开源，且功能上相比 Gazebo 更加强大，但使用范围相对

来说还是有所欠缺，且缺乏技术支持。总而言之，选择机器人的仿真软件进行车辆控制仿真平台的搭建需要极为繁琐的工作，且效果不如无人驾驶仿真平台。

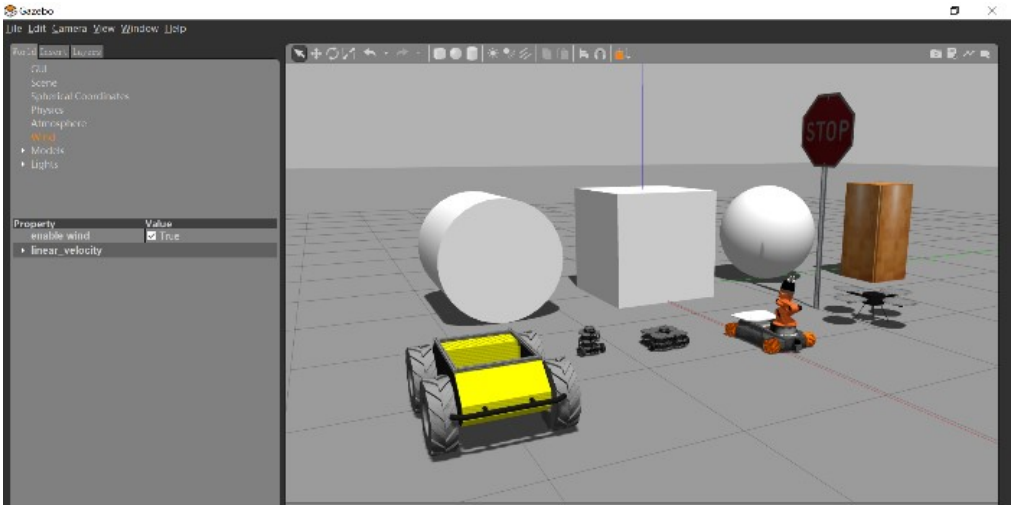
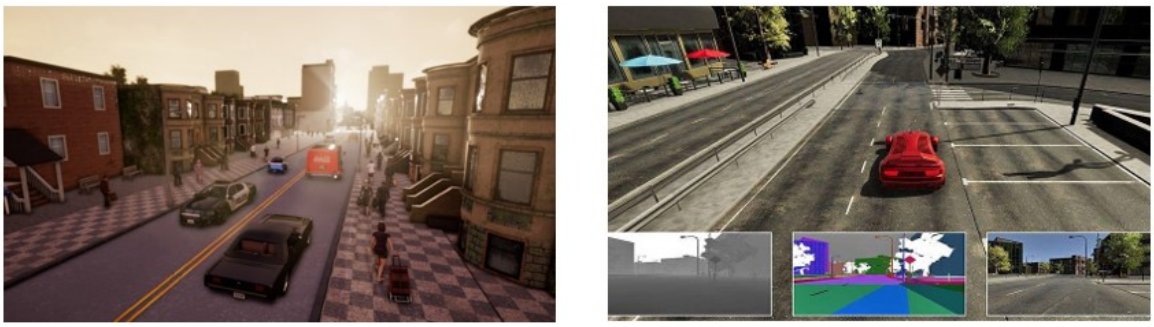


图 1.2 Gazebo

1.2.3 基于游戏引擎的仿真软件

基于游戏引擎的仿真软件具有几近真实的仿真视景，可以模拟比较复杂的仿真。微软开源的 Airsim 和英特尔开发的 Carla 都是基于游戏引擎开发的，该类仿真软件包含了多车道、行人、障碍物、环岛等复杂的交通环境，可以实现从视觉，规划，控制等多层次的仿真，端到端的学习，并且可以仿真车辆与真实世界的交互，如交通规则的仿真等等。因此，选择此类仿真平台方便用户在更复杂，更贴近真实城市交通的路况下测试控制算法。

Airsim 相较于 Carla 能提供更为细腻的环境，但它的地图并不对外开放，用户只能通过 API 调用并控制车辆，因此没法做进一步的开发和可重复性的测试。Carla 提供了多个场景的地图，给出了一系列的 Python 接口和 Python 实例，可以对自动控制算法进行不同场景、不同天气、不同环境的测试，并给出测试性能评估报告，但 Python API 的方式较为复杂，缺乏图形化的操作，使用起来不方便，且无法针对多种轨迹规划的曲线进行自定义测试，无法充分验证控制器的性能。



（a） Carla （b） Airsim

图 1.3 基于游戏引擎的仿真软件

综上所述，现有的无人驾驶仿真平台功能虽然已经十分强大，但是在某些方面仍然需要得到改进。随着无人驾驶研究热潮的兴起，其对仿真平台的要求也必然随之增加。

1.3 无人驾驶横向路径控制方法概述

车辆的运动控制可分为车辆的纵向控制和横向控制。车辆的控制系统的主要作用是根据车辆的运动目标和动力学约束，对车辆的加速，制动，转向等机构进行控制使其能以期望状态行驶，并保证运动过程中的安全性。纵向控制主要通过对刹车和油门的控制来跟踪期望的速度，横向控制通过对方向盘打角进行控制从而实现对期望车道线的准确跟踪。横向控制只包含位置信息，与时间无关，它的主要目的是消除期望位姿和车辆实际位姿的偏差，保证车辆在期望路径下行驶，其基本思路是通过计算车辆的实际位姿和期望位姿的差别，计算对应该误差下合理的方向盘转角，以此来控制小车不断缩小误差，并沿期望路径行驶。因此，车辆横向控制包括横向误差模型，车辆运动学模型以及控制器本身。

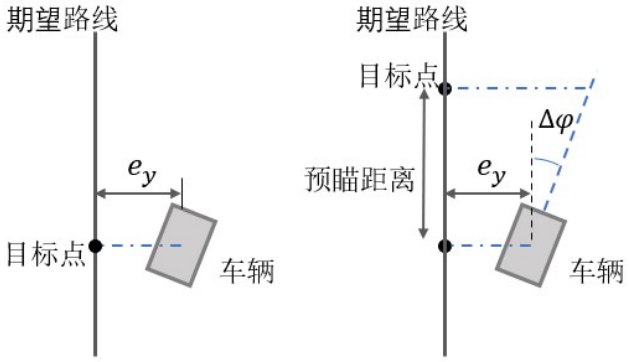


图 1.4 车辆横向误差选择模型

如图 1.4 所示，横向误差模型可以分为无预瞄误差模型和预瞄误差模型。无预瞄误差模型主要是取当前时刻的位姿误差进行反馈，这种模式通常选取车辆质心位置到期望路径的最短距离，以及车辆横摆角与最短距离处的路径横摆角的误差作为横向误差。文献[8][9][10]均采用这种无预瞄误差模型，其中文献[8] 的控制方法曾被应用于斯坦福大学 2005 年 DARPA 的参赛车辆中，并取得良好成绩。但是，由于该控制方法反馈的信息较少，往往适用于速度较低，道路曲率较小的情况下。无预瞄误差模型所选择的参考点可以认为是车辆质心距离期望轨迹的最近点，因为车辆在控制过程中存在滞后，使用该种控制方式容易造成控制器的不稳定。而预瞄的方式能有效缓解该现象，如同人类驾驶员模型也会以前方道路的情况为参考路径一样，其中比较经典的预瞄算法之一是纯追踪算法（Pure Pursuit）。该算法基于车辆运动学模型，在车辆速度恒定时，车辆的轨迹可近似为圆弧，该圆弧的半径与转角成正比，因此在确定车辆当前位置和预瞄点位置后，即可以根据圆弧半径计算控制量[11][12]。

目前，无人驾驶汽车横向运动控制的方法主要有非线性理论控制、基于模型的控制等。其中非线性理论控制包括鲁棒控制，预测控制，滑模控制和自适应控制。滑模控制作为一种先进的非线性控制方法，具有不受外界的干扰，鲁棒性强等优点。文献[13]基于滑模控制的方法提出一种轨迹跟踪控制律，具有较强的鲁棒性，但忽略了控制过程中的抖振，控制效果不理想。也有文献

针对减弱滑模控制引起的抖振展开研究，文献[14]提出了一种抗饱和自适应滑模控制方法来实现机器人轨迹跟踪，然而其控制精度依赖于观测器观测增益的选择。文献[15]研究滑模控制策略控制机器人跟踪轨迹时的转弯角速度，在滑模面附近引入边界层，对滑模控制器采用脉冲函数代替符号函数，但其误差只收敛到零附近一个很小邻域，在一定程度抑制了抖振，但牺牲了控制精度。

1.4 论文主要工作

在文献回顾的基础上，本文搭建了一款新的专用于无人驾驶控制仿真的平台（CSVC），CSVC仿真平台在原有 Carla 模块的基础上添加了新的停车场虚拟仿真场景、校准了车辆的动力学模型参数、改进了底盘控制器等；除了 Carla 原有的模块，本文还设计了具有三种曲线拟合算法的轨迹规划模块，使用户能够通过可视化操作生成不同的仿真轨迹；开发了包括仿真动画效果显示和仿真数据可视化工具在内的可视化模块；最后利用机器人操作系统 ROS 将控制模块、轨迹规划模块、可视化模块以及 Carla 模块相连。

（1）轨迹规划模块可操作性强，支持 2D 和 3D 的视角操作

（3）与 ROS 兼容，在仿真平台中开发和验证的控制算法能一键部署到实车

（3）具有可信度高的车辆动力学模型

（4）支持多场景重复仿真

（5）可视化效果逼真，数据显示便捷易用

改进了基于汽车运动学模型的滑模控制算法，通过设计新的滑模面将横摆角误差和距离误差耦合起来，从而控制车辆方向盘打角。该方法从理论上保证了滑模面的收敛，而不用依靠经验调参的方法保证角度环的收敛速度大于距离环，并使用优化过的切换函数来保证滑模面两侧控制量连续，在一定程度上避免了抖振现象。

引入自适应动态规划 ADP（Adaptive Dynamic Programming）的学习方法，通过学习系统当前状态学习输出一个补偿控制量来帮助调节滑模控制器以改善其性能，最终证明了该方法用于改进控制器的可行性。

本文从车辆动力学模型、底盘控制器性能、轨迹规划模块效果三个方面分别分析了仿真平台的可信度和可靠度。其动力学模型效果十分接近真实的车辆，底盘控制器表现良好，能够精确的跟踪速度命令，所设计的轨迹规划模块能生成三种不同的轨迹曲线来进行仿真。最后，验证了改进的滑模控制算法相较于 MPC 控制算法的优越性，并证明了使用自适应动态规划学习算法来改进滑模控制算法是可行的。

1.5 论文主要结构

本文第一章首先对本课题研究背景及意义给予详细论述，阐述了无人驾驶的概念及发展趋势，其次还介绍了无人驾驶仿真平台的国内外现状以及发展趋势，以及无人驾驶横向控制算法的国内外研究现状。

第二章主要展示了本文开发的控制仿真平台的架构，包括 Carla 仿真平台修改过的模块、轨迹规划模块、可视化模块、以及控制器模块的组成和实现。

第三章详细论述了本文的车辆运动学模型和预瞄误差模型，改进了一种新型的横向距离误差

和横摆角误差耦合的滑模控制算法，并使用自适应动态规划学习算法改进该滑模控制器。

第四章是仿真平台评估以及控制算法的分析，主要是对 CSVC 仿真平台性能的分析以及对比本文改进的控制算法和 MPC 算法的控制效果。

第五章是结论与展望，结论是对于本课题进行的小结，展望部分是对本文工作不足之处的说明以及控制方法的改进。

2 仿真平台框架设计

2.1 仿真平台架构

本文仿真平台的设计架构图如图 2.1 所示，它集成了两个模块，包括现成的模块（来自 Carla 仿真平台）和添加的 CSVC 新模块。Carla 中的模块功能主要是提供地图和车辆的虚拟模型以及车辆在地图环境中的状态信息，如位姿信息、速度及加速度信息等，并将无人驾驶车辆上层的控制命令通过物理模型转换为动画效果显示。CSVC 的新模块用于满足仿真控制算法的需求和提高仿真效率，它包括轨迹规划模块、控制模块以及可视化模块，如图 2.1 中紫色虚线框中所示。模块之间的连接和通信基于开源机器人框架 ROS[16][17]实现，如图 2.2 所示。

具体来说，轨迹规划模块包括一个 3D 场景设计器和 2D 轨迹规划器，通过场景设计器的图形界面，用户可以自定义车辆运行的地图，地图中障碍物的位置，车辆轨迹经过的路径点。轨迹规划器的目的是通过用户设定的场景，使用三次样条曲线、Catmul-Rom 曲线以及 B 样条曲线三种曲线生成车辆的目标轨迹。控制模块分为横向控制器和纵向控制器，分别采用了不同的控制器进行控制，得益于机器人系统 ROS 强大的分布式节点设计以及多种编程语言的支持能力，用户能十分便捷的开发并验证自己的控制器，并一键部署到真实的车辆上。可视化模块包含数据可视化和场景可视化两大模块，数据可视化在 ROS 的 RQT_PLOT 软件的基础上实现了包括控制命令、车辆状态信息以及环境信息在内的多种数据的实时显示及画图，用户也能通过发布 ROS 消息十分方便的可视化调试需要的信息。场景可视化模块利用 RVIZ 软件开发了多角度实时显示仿真效果的 ROS 节点，分别采集了车辆驾驶的摄像头视角，车辆后方摄像头视角，车辆上方的俯视图视角以及整个仿真路径下的全局视角，让用户能够直观的感受控制器的效果。注意，在图 2.1 中，带有橙色标志的模块是本文在 CARLA 现有的模块基础上进行二次开发得到。

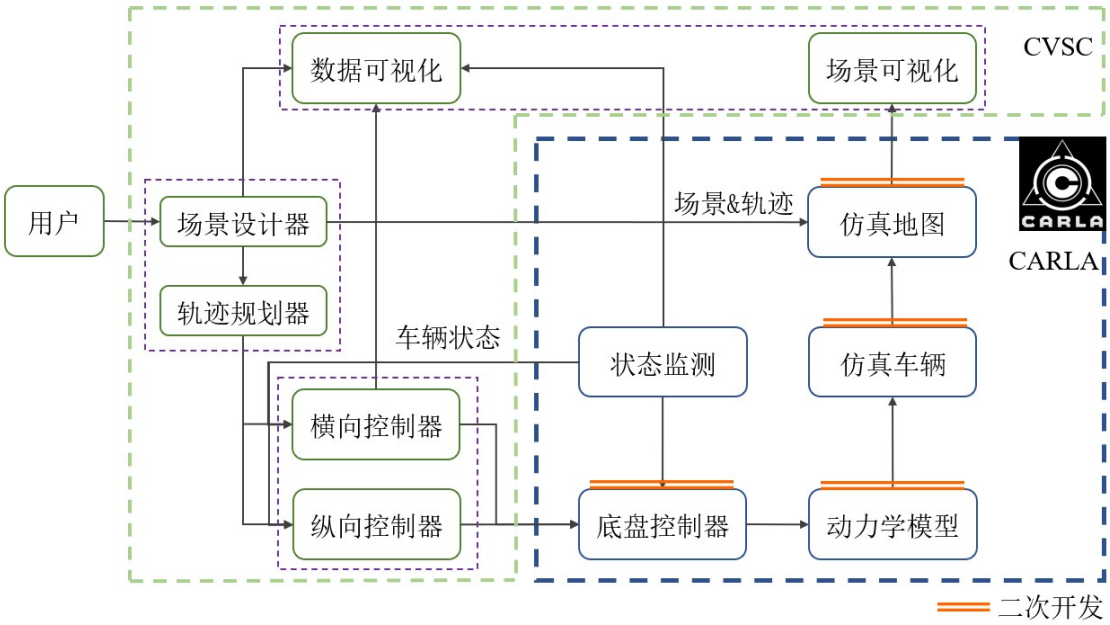


图 2.1 CSVC 仿真平台架构

2.2 Carla 二次开发

Carla 是一款开源的无人驾驶仿真平台，它可以模拟真实的交通环境，行人行为，汽车传感器信号等。该模拟器由数个地图构成，用户可以选择一个地图运行模拟器，地图中有道路，建筑，街景，交通灯等，使用者可以在模拟器外的 Python API 对模拟器的环境进行操作和控制，比如获得车载摄像头的图像，发布车辆的目标轨迹，添加车辆，控制车辆的物理属性，以及通过油门，刹车，方向盘等控制量对车辆进行控制等。本文通过对 Carla 源码编译，添加了一张用于控制仿真的停车场地图以及车辆“夸父”的 3D 模型，并对车辆物理学模型进行了参数校准，改善了底盘的控制器，使其跟踪效果更贴近实际。本文利用开源机器人框架 ROS 的节点管理将每一个模块连接，节点之间的消息封装成 ROS Message 进行管理，本文中仿真平台的 ROS 节点和 ROS Message 如图 2.2 所示：

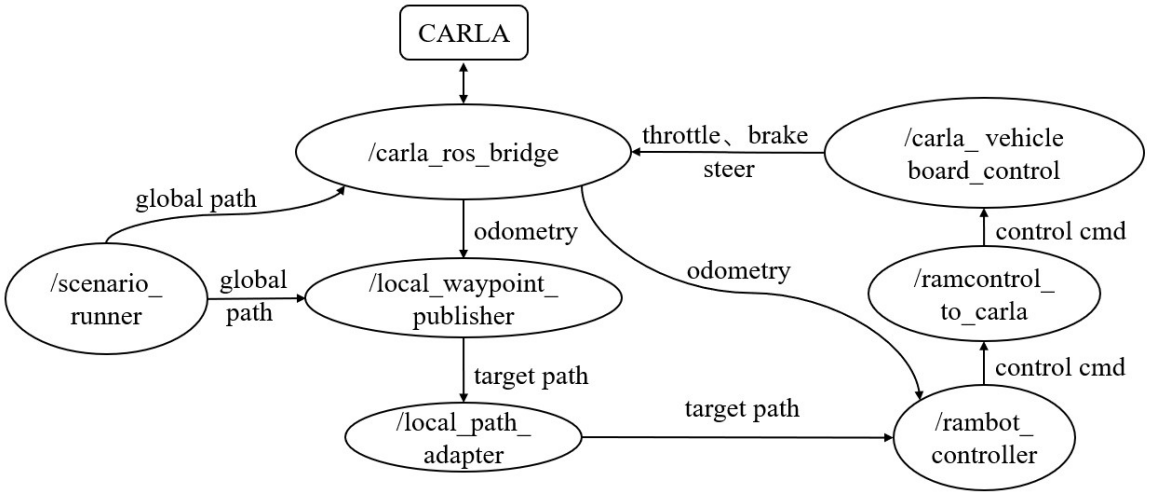


图 2.2 CSVC 仿真平台 ROS 节点图

在图 2.2 所示的节点图中，/scenario_runner 和/local_waypoint_publisher 节点对应于图 2.1 中的轨迹规划模块，/rambot_controller 节点对应控制模块，/carla_vehicle_board_control 对应底盘的刹车油门控制器，可视化的模块未在 ROS 节点图中显示，因为每一个模块均可传递消息到可视化模块。下面将分别讨论二次开发的 Carla 模块。

2.2.1 仿真地图

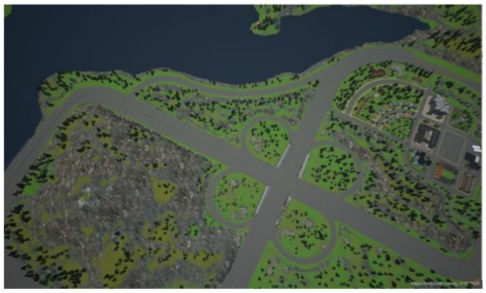
一个良好的仿真平台需要具有良好的沉浸性和交互性，因此地图的建立应该尽量模拟真实的情况。CARLA 仿真器提供了七张非常精致的地图，包括了城市地图和郊区地图，如图 2.3（a）（b）所示。但控制器的测试要求满足任意曲线的行驶，因此 Carla 仿真器中的虚拟场景不满足测试的需求。本文对仿真器中的测试场景重新进行了设计，在保证足够的物理环境真实度的情况下，满足了控制器仿真的要求，重新设计的场景应满足如下需求：

(1) 车辆在地图中可以进行任意轨迹测试

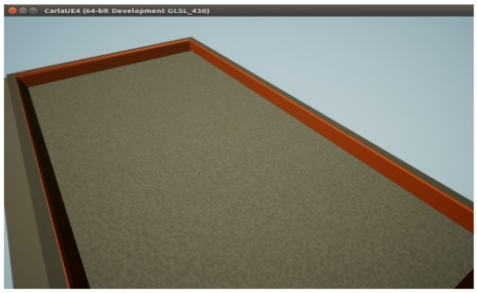
(2) 地面具有碰撞检测属性

虚幻引擎是全球最放、最先进的实时 3D 虚拟仿真平台，具有照片级逼真的渲染功能、动态物理效果，逼真的动画，健壮的数据接口等。它是一个开放而且可以自由扩展的平台，具有庞大的社区资源，能够快速的构建真实和实时交互的三维环境。针对地面建模，虚幻引擎编辑器中有专门用于地形建模的资产，由于本文开发的是控制器验证的仿真平台，要求路面平整，采用常规的材料建立地面，如图 2.3（c）（d）所示。

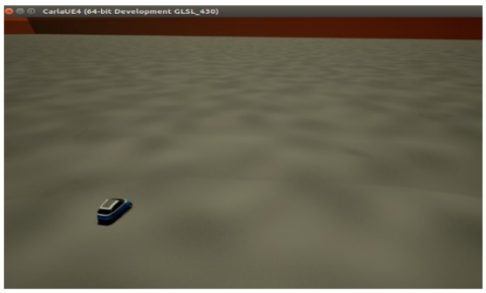
(a)CARLA地图Town3



(b) CARLA地图Town4



(c)CSVC地图



(d)CSVC地图局部放大图

图 2.3 仿真平台地图

2.2.2 仿真车辆

仿真平台要求在准确真实的车辆物理模型基础上，虚拟车辆与实际车辆拥有相似或一致的外观，以达到更逼真的动画效果。本文将 C4D 软件中已经建立好的车辆模型保存为 fbx 文件并导入 Unreal Engine 中，图 2.4（b）为最终效果图，图 2.4（a）为“夸父”无人车平台的真实照片。

（a）“夸父”无人车

（b）CSVC 仿真车辆效果图

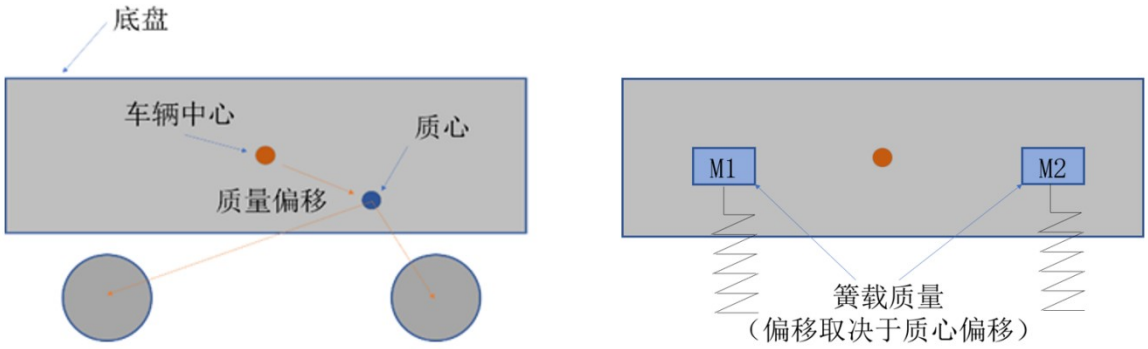
图 2.4 仿真车辆效果

车辆在虚拟场景中进行仿真时，碰撞检测是必不可少的功能。在无人驾驶仿真中，碰撞检测分为与地面的碰撞、与障碍物的碰撞、与路边线的碰撞等。因为本文建立的是空白地图，所以其没有车道线，车辆不需要检测与路边线的碰撞。与地面的碰撞检测是为了防止车辆钻地而入，与障碍物的碰撞在本文的仿真中表现为与四周围墙的碰撞。碰撞检测的实现方法为：对车辆设置碰撞检测长方体、对地面和四周的围墙采取顶点连线的方式检测碰撞。

2.2.3 车辆力学模型

Carla 所使用的 PhysX Vehicle 力学模型是全面的车辆模型，被广泛用于车辆以及无人驾驶的仿真之中，它支持发动机，离合器，齿轮，自动变速箱，差速器，车轮，轮胎，悬架和底盘的模拟。

PhysX Vehicle 将车辆建模为弹簧承载质点的集合，其中每个弹簧承载质点代表一条悬挂线，包含相关的车轮和轮胎数据。这些簧载质量点的集合可以作为一个刚体作用体的表示，该刚体作用体的质量、质心和惯性矩与使用的簧载质量点及其坐标完全匹配，如图 2.5 所示。



（a）刚体车辆模型　　（b）使用簧载质量集合表示车辆

图 2.5 车辆模型表示

通过如下的刚体质心方程可以从数学上表示簧载质量点表示的原理：

$$\begin{cases} M = M1 + M2 \\ X = \dfrac{M1 \times X1 + M2 \times X2}{M1 + M2} \end{cases} \tag{2.1}$$

其中 $M1$ 和 $M2$ 为簧载质量点; $X1$、$X2$ 为簧载质量点坐标; M 是刚体质量; X 是刚体质心的偏移量。

PhysX Vehicle 使用上述簧载质量点模型计算悬架力和轮胎力，然后将其应用到汽车所代表的刚体上，从而修正它的速度以及角速度。其中弹簧的压缩和伸长可以分别对车辆施加作用。每一个弹簧被压缩或拉长所产生的悬架力都被应用到代表车身的刚体上，此外，它还被用来计算轮胎所受到的载荷，然后这个载荷用来确定接触面上产生的力，而后这个力被施加到车身刚体上。轮胎力的计算实际上取决于许多因素，包括转向角、外倾角、摩擦力、车轮转速和刚体动量，最后，PhysX Vehicle 将所有轮胎和悬挂力的合力作用于与车辆刚体。

在簧载质量点模型的基础上，PhysX Vehicle 还集成了许多经典驱动模型。如下拉杆模型[18]，默认的轮胎模型是从 CarsimEd 中导出的 Pacejka 轮胎模型[19]，驱动模型的中心是一个扭转离合器模型[20]，它通过离合器[21]两边的转速差异产生的力将车轮和发动机连接在一起。离合器的一侧是发动机[22]，发动机被建模为一个刚体，它的运动是纯转动的，直接由油门踏板提供动力，并且被限制在一个转动自由度上。离合器的另一侧是齿轮传动系统、差速器模型[23]和车轮。这种模式允许扭矩在发动机和车轮直接来回传播，就像一个标准的汽车[24]。经过对夸父无人车测试，在 Physx Vehicle 车辆模型的基础上建立“夸父”无人车的动力学模型，下表 2.1 为修改的部分车辆参数列表，参数含义请参考[24]，其余的参数均为模型默认或建议参数：

表 2.1 车辆动力学参数

车辆参数	值
轮胎摩擦系数	3.5
轮胎阻尼率	0.25
车辆总质量（kg）	600
质心偏移（cm）	（x，y，z）=（21，0，-40）
发动机最大转速（r/m）	1250
发动机转动惯量	1
车长（cm）	187

2.3 场景&轨迹规划器

本文中轨迹规划模块基于 xml 配置文件[25]建立，通过 scenario_runner 自定义一个或多个场景，场景的要素包括：地图；车辆；车辆起始位置；车辆目标轨迹；障碍物的位置。本文开发了一个 3D 场景设计器以及 2D 轨迹规划器，使用户可以通过手动操作代替 xml 配置文件的编写，从而极大的方便了用户场景的定义，场景设计器效果如图 2.6 所示，其能在设定障碍物以及数个需要经过的点后，通过三种不同轨迹曲线拟合出车辆的目标轨迹。如果使用空白地图，则三种不同轨迹曲线的拟合以及车头初始位置和初始横摆角均可以在 2D 轨迹规划器中实现，如图 2.7 所示。完成场景和轨迹生成后，导出的 xml 样例配置文件如下所示，可以复制 scenarios 标签一次性完成多次重复且包括不同地图的无人驾驶车辆仿真。

a） 主视图　　（b） 俯视图

图 2.6 3D 场景设计器

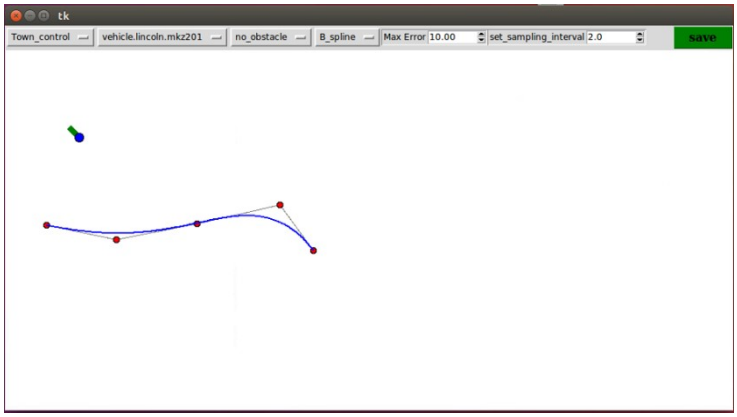


图 2.7 2D 轨迹规划器

部分场景 xml 配置文件如下：

```
<?xml version="1.0"?>
<scenarios>
    <scenario name="ControlAssessment" type="ControlAssessment" town="Town_control">
        <ego_vehicle x="22.8110" y="2.84962" z="0.0" yaw="0" model="vehicle.kuafu"/>
        <obstacle>
            <position x="28.8450" y="5.4284" z="0.0" yaw="0.0" name="constructioncone"/>
        <obstacle>
        <route>
            <waypoint x="22.8110084534" y="2.84962892532" z="0.0" yaw="-83.7545"/>
            <waypoint x="23.7731933594" y="3.40834927559" z="0.0" yaw="-82.6658"/>
            ….
        </route>
    </scenario>
</scenarios>
```

其中 scenario 标签表示一个场景，在 scenarios 标签下可拥有多个 scenario 标签，从而实现多个场景的仿真，town 属性为仿真使用的地图，ego_vehicle 标签代表仿真使用的车辆，obstacle 标签下含有多个不同的障碍物的位置信息，route 标签则含有车辆目标轨迹的一系列路径点信息。在进行仿真时，local_waypoint_publisher 会读取该文件，并结合车辆的状态计算当前目标轨迹。

一个性能良好的控制器需要经过不同路径和不同场景下的验证，比如直线和弯道跟随能力，起始点偏离控制，低速和高速下的轨迹跟踪能力等等。控制器的性能优良与否，与轨迹规划有着密切相关的联系，轨迹规划必须在满足车辆的动力学约束情况下，最小化特定的变量，比如最小化总时间，或是总加速度的大小。本文基于三次样条规划，Catmul-Rom 算法，以及 B 样条规划三种不同的轨迹生成算法，在给定路径点的情况下，生成车辆的目标轨迹。

2.3.1 三次样条规划

轨迹规划需要在给定起始点和终点，以及中间期望经过的路径点的情况下，获得最小化输入变化率的最优路径曲线[26]。根据变分学原理：

$$x^*(t)=\arg\min_{x(t)}\int_0^T \mathcal{L}(\dot{x},x,t)dt \tag{2.2}$$

该函数能最小化代价函数$\mathcal{L}$对t的积分，车辆的动力学系统是2阶系统，则需要最小化第2阶变量即加速度的输入，即：

$$x^*(t)=\arg\min_{x(t)}\int_0^T (\ddot{x})^2 dt \tag{2.3}$$

据欧拉-拉格朗日方程可以得到：

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot{x}}\right)-\frac{\partial\mathcal{L}}{\partial x}-\frac{d^2}{dt^2}\left(\frac{\partial\mathcal{L}}{\partial\ddot{x}}\right)=0 \tag{2.4}$$

其中代价函数$\mathcal{L}$等于为：

$$\mathcal{L}=(\ddot{x})^2 \tag{2.5}$$

则欧拉-拉格朗日方程可以化简为：

$$x^{(4)}=0 \tag{2.6}$$

因此得到的轨迹应该为如下形式：

$$x(t)=c_4+c_3t+c_2t^2+c_1t^3 \tag{2.7}$$

事实上，这就是三次样条规划的本质[27]，如果该物体的动力学系统是四阶，则需要使用四次样条拟合算法来进行轨迹规划。三次样条规划可以最小化输入加速度的大小。对无人驾驶车辆来说，不考虑z方向上的位移和速度，如图2.8（a）所示，假设中间要求经过三个路径点，每段的时间经过归一化为1。x方向和y方向各需要求解16个系数，因此一共由四个方程以及16个边界条件，分别是：

四条多项式曲线的方程可以表示为：

$$\begin{cases} x_1(t)=c_{14}+c_{13}t+c_{12}t^2+c_{11}t^3 \\ x_2(t)=c_{24}+c_{23}t+c_{22}t^2+c_{21}t^3 \\ x_3(t)=c_{34}+c_{33}t+c_{32}t^2+c_{31}t^3 \\ x_4(t)=c_{44}+c_{43}t+c_{42}t^2+c_{41}t^3 \end{cases} \tag{2.8}$$

相应的边界条件为：

$$\begin{cases} x_1(0)=x_0; x_2(0)=x_1; x_3(0)=x_2; x_4(0)=x_3; \\ x_1(1)=x_1; x_2(1)=x_2; x_3(1)=x_3; x_4(0)=x_4; \\ \dot{x}_1(0)=0; \dot{x}_4(0)=0; \ddot{x}_4(0)=0; \\ \dot{x}_1(1)=\dot{x}_2(0); \dot{x}_2(1)=\dot{x}_3(0); \dot{x}_3(1)=\dot{x}_4(0); \\ \ddot{x}_1(1)=\ddot{x}_2(0); \ddot{x}_2(1)=\ddot{x}_3(0); \ddot{x}_3(1)=\ddot{x}_4(0); \end{cases} \tag{2.9}$$

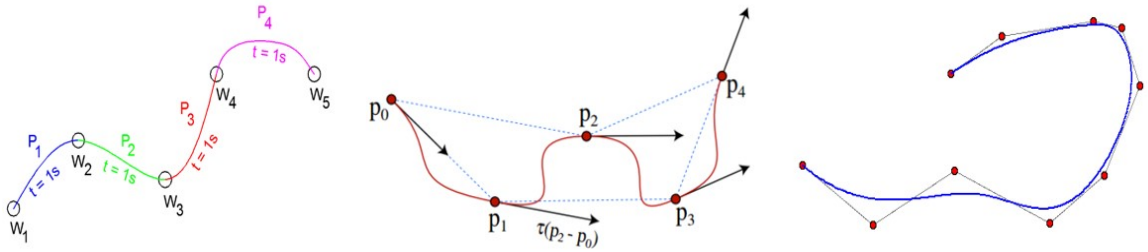
可使用矩阵的运算求出各个多项式的系数，从而求得每一个点的位置，横摆，速度以及加速度。

2.3.2 Catmul-Rom 规划

Catmul-Rom 算法[28]用于四个点及以上的点之间的轨迹规划，它保证一下两个条件：

（1）曲线通过每一个点。

（2）某一点的切线平行于左右相邻两点的连线。



（a） 三次样条轨迹　　（b） Catmul-Rom 轨迹　　（c） B 样条轨迹

图 2.8 轨迹对比示意图

作为一个立方插值函数[29]，可以假设每一段曲线（以 P_1-P_2 为例）可以表示为：

$$P(x)=c_4+c_3x+c_2x^2+c_1x^3 \tag{2.10}$$

根据 Catmul-Rom 算法原理，可得到其边界条件为：

$$P(0)=P_1;P(1)=P_2;\dot{P}(0)=\tau(P_2-P_0);\dot{P}(1)=\tau(P_3-P_1); \tag{2.11}$$

同理根据矩阵运算，即可十分方便的求得曲线函数。Catmul-Rom 算法对于第一个点和最后一点存在无法求解的问题，因此，通常采用在最前方和最后方虚构一个点的方法来求解出一条曲线，并去掉两端多余的曲线。

2.3.3 B 样条规划

B 样条曲线[30]拟合是 Bezier 曲线的改进，k 次 B 样条曲线的总方程可以描述为：

$$P(u)=\sum_{i=0}^{n}P_iF_{i,k}(u) \tag{2.12}$$

其中P_i是控制曲线的特征点，$F_{i,k}(t)$ 是 k 阶 B 样条基函数，其数学表达式可以表述为：

$$\begin{cases} F_{i,0}(u)=\begin{cases}1, & u_i<u<u_i\\ 0, & else\end{cases}\\ F_{i,k}(u)=\dfrac{u-u_i}{u_{i+k}-u_i}F_{i,k-1}(u)+\dfrac{u_{i+k+1}-u}{u_{i+k+1}-u_{i+1}}F_{i+1,k-1}(u)\\ \dfrac{0}{0}=0 \end{cases} \tag{2.13}$$

假设 B 样条曲线上的$n-k$个分段连接点对应于控制多边形上除两端各$(k+1)/2$个顶点外其余的$n-k$个控制顶点，根据这些连接点，可以 B 样条曲线的节点向量为：

$$U=[0,0,0,0,\frac{l_1}{L},\frac{l_1+l_2}{L},\frac{l_1+l_2+l_3}{L},\frac{l_1+l_2+...+l_{n-2}}{L},1,1,1,1] \tag{2.14}$$

其中$l_1...l_n$表示连接点之间的距离，L表示距离之和。通过求解如下的矩阵[31]方程，便可以求得控制点的坐标。

$$\begin{pmatrix} b_1 & c_1 & ... & ... & ... & a_1 \\ a_2 & b_2 & c_2 & ... & ... & ... \\ ... & a_3 & b_3 & c_3 & ... & ... \\ ... & ... & ... & ... & ... & ... \\ ... & ... & ... & a_{n-3} & b_{n-3} & c_{n-3} \\ c_{n-2} & ... & ... & ... & a_{n-2} & b_{n-2} \end{pmatrix} \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{n-3} \\ d_{n-2} \end{pmatrix} = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_{n-3} \\ e_{n-2} \end{pmatrix} \tag{2.15}$$

其中：

$$\begin{cases} a_i = \dfrac{(l_{i+2})^2}{l_i+l_{i+1}+l_{i+2}} \\ b_i = \dfrac{l_{i+2}(l_i+l_{i+1})}{l_i+l_{i+1}+l_{i+2}} + \dfrac{l_{i+1}(l_{i+2}+l_{i+3})}{l_{i+1}+l_{i+2}+l_{i+3}} \\ c_i = \dfrac{(l_{i+1})^2}{l_{i+1}+l_{i+2}+l_{i+3}} \\ e_i = (l_{i+1}+l_{i+2})q_{i-1} \qquad (i=1,2,...,n-2) \end{cases} \tag{2.16}$$

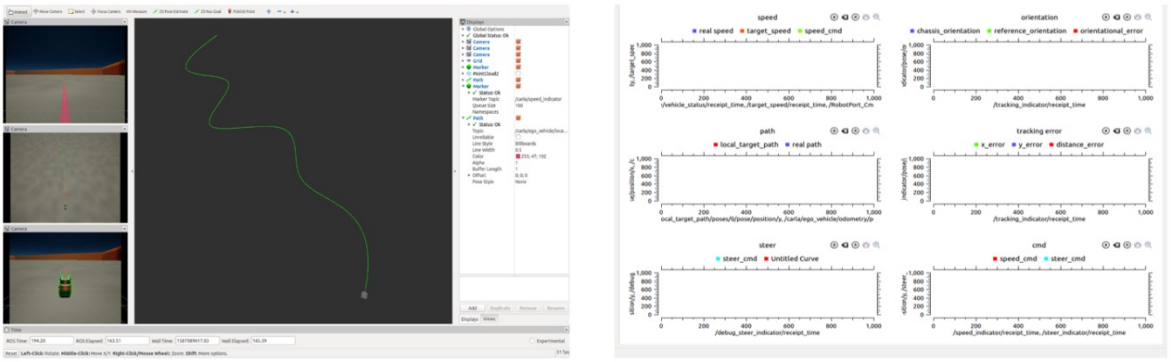
注意平面上的轨迹点具有 x 坐标和 y 坐标，因此需要单独求解两次。得到控制点之后，即可通过迭代求出 B 样条曲线[32]。其轨迹效果如图 2.8（c）所示：

2.4 可视化模块

2.4.1 场景可视化

为了更直观的体现控制器的效果，本文利用 RVIZ 软件[33]开发了多角度实时显示仿真效果的 ROS 节点，分别采集了车辆驾驶的摄像头视角，车辆后方摄像头视角，车辆上方的俯视图视角以及整个仿真路径下的全局视角，让用户能够直观的感受控制器的效果，其效果如图 2.9（a）所示，绿色曲线为全局轨迹，红色轨迹为当前发布的轨迹。

RVIZ 是集成在 ROS 里面的软件，通过编写程序可以订阅 ROS 节点中的消息，然后在可视化界面中显示出来，它具有特定的消息类型，如：Camera， Video， Marker， Path 等消息类型，能满足机器人和无人驾驶开发的需要，因此本文通过订阅/carla/ego_vehicle/local_target_path 话题得到路径信息，并在 RVIZ 中显示，通过订阅/carla/ego_vehicle/camera/rgb/top/image_color 获得俯视图图像，同理获得其它方向的摄像头图像。 编写 carla/ego_vehicle/config/sensors.json 文件，可以编辑 Carla 中返回的传感器数据，如下所示，是一个摄像头的配置代码，其余的摄像头同理。



（a） 场景显示效果　　（b） 数据可视化效果

图 2.9 可视化模块效果

sensor.json 配置文件如下：

```
1.    "sensors": {
2.          "type": "sensor.camera.rgb",
3.          "id": "front",
4.          "x": 2.0, "y": 0.0, "z": 2.0, "roll": 0.0, "pitch": 0.0, "yaw": 0.0,
5.          "width": 800,
6.          "height": 600,
7.          "fov": 100 }
```

同时需要在 RVIZ 软件编写摄像头读取程序，其中一个 Camera 的部分配置程序如下所示（其他程序详情请见附件），若需要继续添加传感器，只需要在 Displays 面板中，增加 Camera 类。

RVIZ 配置文件如下：

```
1.  Displays:
2.    - Class: rviz/Camera
3.      Enabled: true
4.      Image Rendering: background and overlay
5.      Image Topic: /carla/ego_vehicle/camera/rgb/front/image_color
6.      Name: Camera
7.      Visibility:
8.        Camera: true
9.        Grid: true
10.       Marker: true
11.       Path: true
```

2.4.2 数据可视化

控制器的仿真不仅需要直观的动画模拟效果，更需要实时的控制数据和状态信息并对其进行可视化分析。单独在某一个节点中的 Python 或者 C++文件中实现实时画图较为麻烦，且效果不好。因此，本文在 ROS 系统 RQT_Multiplot 插件的基础上，编写如下所示的配置文件，即可达到如图 2.9（b）的效果。

二维数据可视化配置文件如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<rqt_multiplot>
 <table>
   <background_color>#ffffff</background_color>
   <foreground_color>#000000</foreground_color>
   <plots>
     <row_0> <column_0> ..... </column_0>  .....  </row_0>
     <row_1> <column_1> ..... </column_1>  .....  </row_1>
   </plots>
 </table>
</rqt_multiplot>
```

以上为配置程序的框架，在<column>标签中编写每一个图的配置程序，通过如下两行程序修改该图显示的数据变量（为 ROS 节点的其中一个话题）：

```
<topic>/carla/ego_vehicle/vehicle_status</topic>
<type>carla_msgs/CarlaEgoVehicleStatus</type>
```

2.5 控制器

CSVC 仿真平台的控制器部分如图 2.10 所示，它由横向控制器、纵向控制器、以及 Carla 模块中的底盘控制器组成。其中横向控制器包括滑模控制器以及自适应动态规划滑模控制器，横向控制器获得由横向误差模型计算出的横向距离误差和横摆角误差，并根据误差计算车辆的方向盘转角命令。纵向控制器为 PID 控制器，根据位置误差以及规划速度计算出速度命令并发送至底盘控制器。底盘控制器分为刹车油门控制器和转角控制器，其中刹车油门控制器通过输入的速度命令和当前速度得到速度偏差，并利用 PID 控制器求取加速度命令，然后根据加速度命令，采用模糊规则控制的方法来控制油门和刹车，即通过专家制定先验知识的办法来控制车辆的刹车和油门。转角控制器对转角增量进行一个限制，让其更加符合实际情况。注意横向误差和纵向位置误差都是通过当前的车辆状态信息和目标车辆状态信息得到。其中目标车辆状态信息由规划器根据当前车辆状态信息计算得到，而当前的车辆状态信息是调用 Carla 中的 Python API 返回得到。

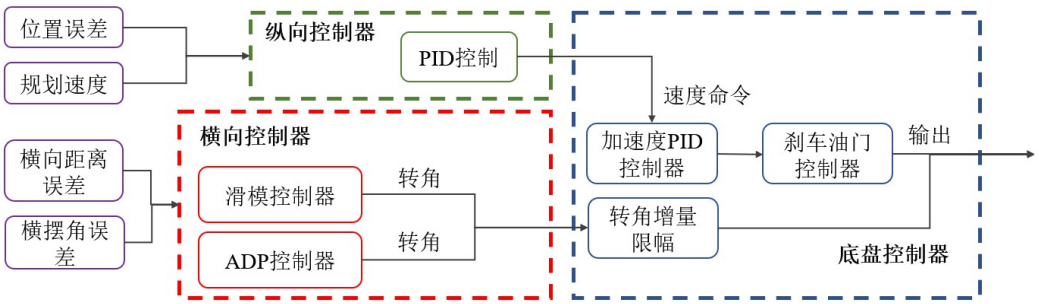


图 2.10 控制器模块架构

在 CSVC 仿真平台下的 ego_vehicle.py 文件中，程序通过 CARLA 的 API 获得车辆的状态信息，并通过 ROS 将之发布，其获取的车辆状态信息如表 2.

表 2.2 车辆信息

变量	值
速度/（$m\cdot s^{-1}$，$m\cdot s^{-1}$，$m\cdot s^{-1}$）	/
加速度/（$m\cdot s^{-2}$，$m\cdot s^{-2}$，$m\cdot s^{-2}$）	/
姿态/（rad，rad，rad）	/
油门	（-1，1）
刹车	（0，1）
转角/rad	（-0.6109，0.6109）

2.5.1 横纵向控制器

横纵向控制器能使车辆在不同的车速、载荷、路况以及风阻条件下自动跟踪行车路线，并保持一定的舒适性和平稳性的要求。本文中横向控制器的具体结构如图 2.11 所示，滑模控制器或 ADP 滑模控制器通过预瞄误差模型得出的横向距离误差和横摆角误差来控制车辆方向盘转角。而纵向控制主要表现为对速度的控制，其工作原理如图 2.12 所示，由 PID 位置控制器[34]根据纵向位移偏差输出一个补偿速度，并结合规划速度输出一个目标速度到底盘控制器（如图 2.13 所示）。底盘控制器再根据车辆当前速度进行加速度的控制。由于本文主要研究的是横向控制算法，滑模控制以及 ADP 滑模控制在第四章中有更详细的叙述，有关 PID 的车辆纵向控制请参考文献[34]。

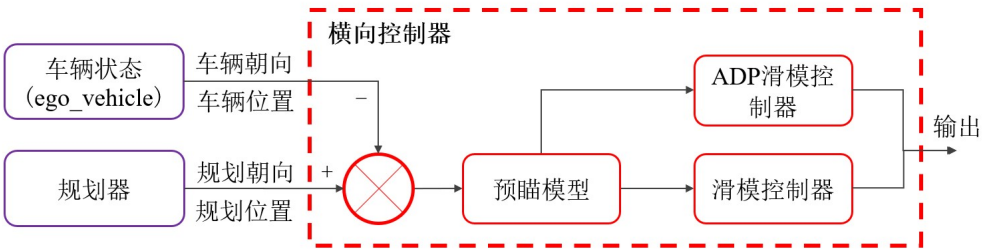


图 2.11 横向控制器

装 订 线

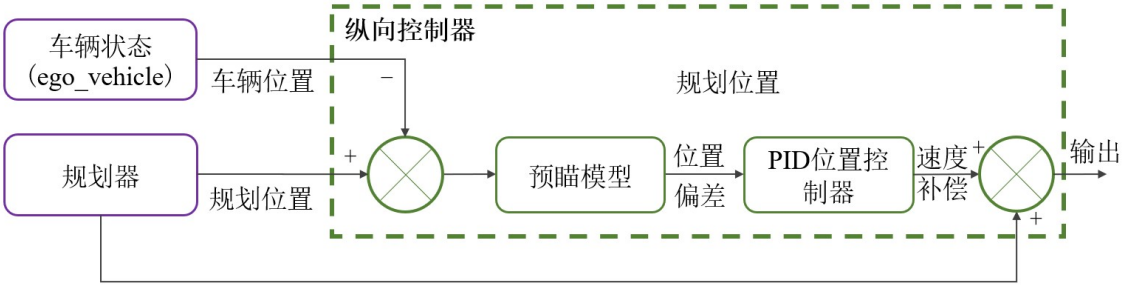


图 2.12 纵向控制器

2.5.2 底盘控制器

为了将上层控制器中的速度命令映射为油门以及刹车命令并根据上层的方向盘转角命令控制方向盘，底盘控制通过 PID 控制器输出目标加速度控制油门和刹车，并通过模糊选择规则选择油门控制还是刹车控制， 定义速度偏差 $e(k)$ 为目标速度减去实际速度 $e(k)=v_{desired}-v_{real}$，以及阈值 $e_{switch}=10km/h$，其选择规则如图 2.14 所示。当速度误差 $e(k)$ 小于零，即实际速度大于目标速度，且速度误差的绝对值大于阈值 e_{switch} 时，启用刹车控制，并将油门置零，而当速度误差的绝对值减小到 e_{switch} 以内，且此时目标速度不为零时，将刹车置零，并启用油门控制，使车辆的速度误差稳定在零值。

底盘控制器还对上层控制器的转角命令做了限幅的控制，根据实际情况，方向盘转动的速度一般在 $0.5075\,rad/s$， 通过对当前方向盘转角以及转角命令的差值做一个限幅，使方向盘的控制更加贴近真实情况。

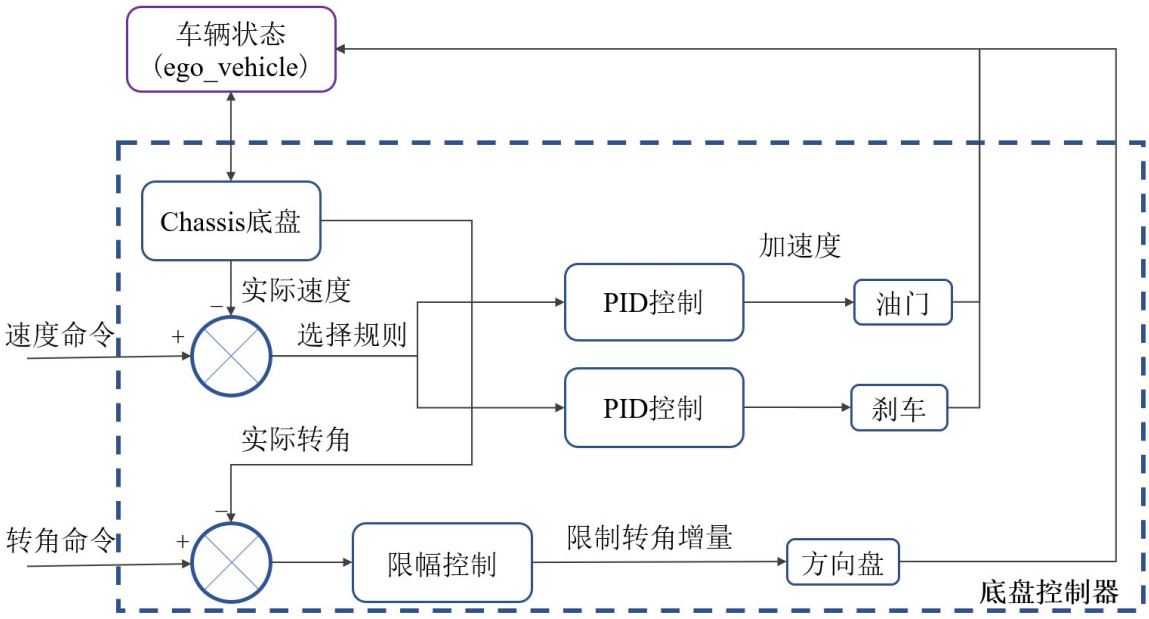


图 2.13 底盘控制器

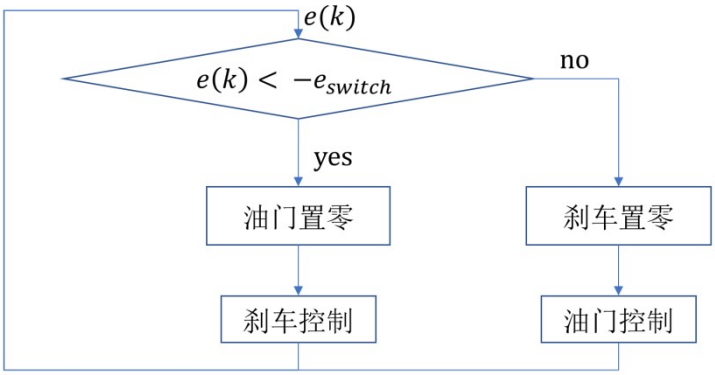


图 2.14 刹车-油门控制选择规则

3 横向控制器设计与改进

研究车辆的路径跟踪是研究车辆如何在已有动力学约束的情况下更好的控制自身的姿态和位置，这也是实现任意环境无人驾驶的基础和关键技术。由于车辆在实际运动的过程中有着极度复杂的动力学表现，影响其动力学模型的因素非常多，且该模型呈高度非线性。本课题基于滑模控制策略，改进了一种新型的横向距离误差和横摆角误差耦合的滑模模型，并利用自适应动态规划输出一个补偿控制量，有效的改善了滑模控制器的性能，并在 CSVC 仿真平台上对不同的目标运动轨迹进行了仿真分析，比较了有无自适应动态补偿控制的差别。

3.1 横向控制器架构

本文无人驾驶车辆横向跟踪控制方案如图 3.1 所示。在本设计中，滑模控制器部分采用了耦合的滑模面设计原理，将横向距离误差和横摆角误差耦合到一个滑模面 S，然后让该滑模面的倒数等于所设计的控制律，以求出车辆的目标转角，从而确保横向距离误差和横摆角误差在该控制律的作用下收敛至零，最后该转角控制量会传入到 CSVC 仿真平台中的车辆动力学系统，以获得下一时刻车辆的状态更新。ADP 自适应动态规划补偿控制部分选取横向误差和横摆角误差作为输入，通过反向传播算法训练网络的权重，对滑模控制器做一个补偿控制，从而缩小和减轻滑模控制器的追踪误差和抖振现象。

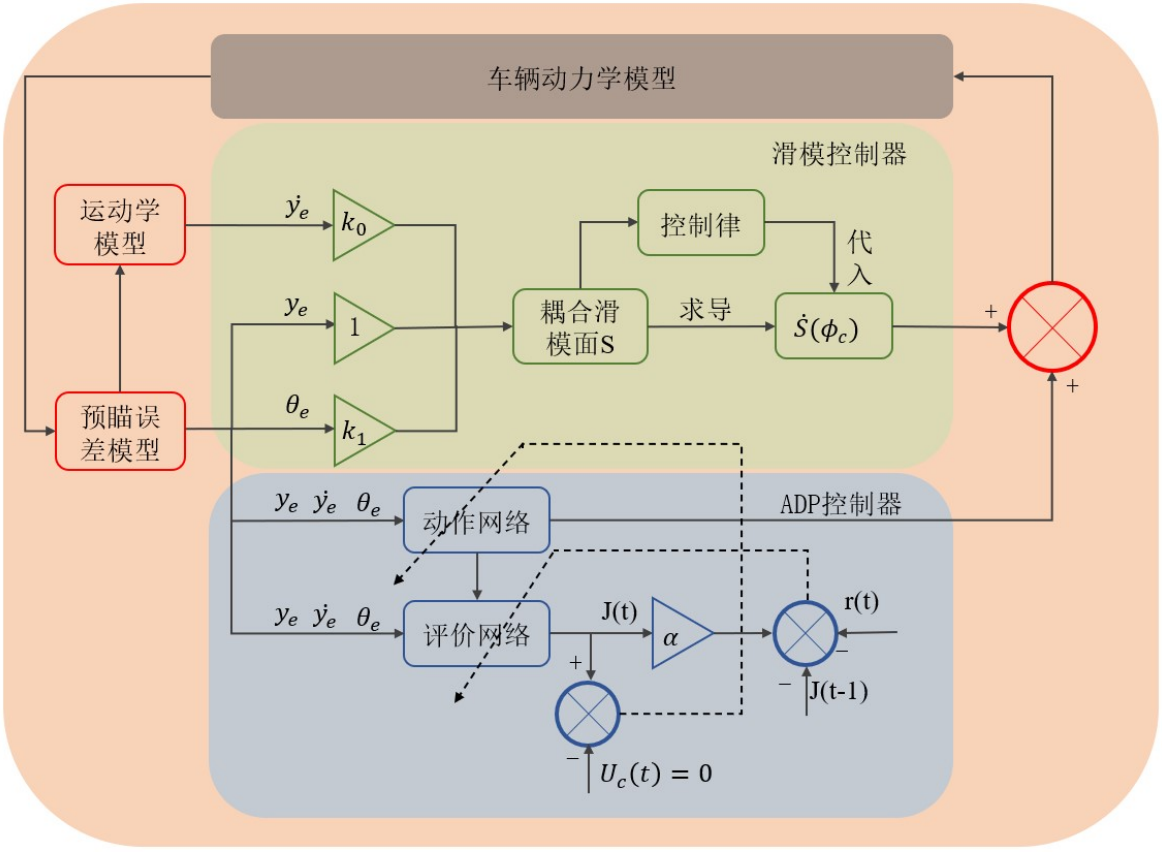


图 3.1 横向控制器架构

3.2 车辆模型

3.2.1 运动学模型

要实现对车辆的控制，需要研究车辆驾驶时的相互作用，即研究行驶时汽车受到的约束，其中又包括运动学模型和动力学模型。运动学模型是指不考虑力的作用，从几何学的角度研究物体的运动规律，控制器能够随意的控制速度，即不考虑加速度以及更深层的约束。动力学模型则通过对轮胎和路面之间的复杂相互作用来描述车辆的运动，因为车辆高速运动会影响轮胎与地面之间的受力情况，从而影响控制效果，这时需要引入动力学模型。

自行车模型（Bicycle Model）是最常见的车辆模型，它假设车辆左右对称，即左右的两个轮胎拥有一样的角度和转速，在自行车模型的基础上搭建的运动学模型具有较真实的车辆特性并且十分简单，建立运动学自行车模型需要做如下假设：

（1）忽略车辆的 z 轴方向的移动，即竖直方向上的移动，车辆只在水平面内移动。

（2）车辆行驶速度较慢，可忽略前后轴载荷的转移

在上述情况下，汽车可以等效成一辆自行车，如图 3.2 所示，

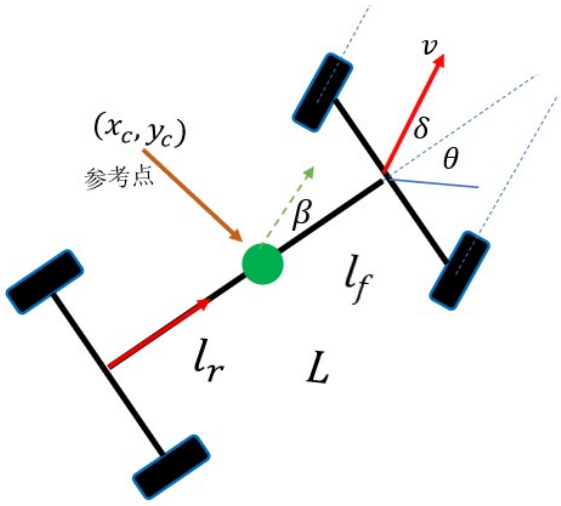


图 3.2 运动学自行车模型

当车辆为前轮驱动时，其后轮的偏角可以近似等于零。假设车辆是纯刚体，可以用质心代表其运动，则车辆质心（图中绿色点）的速度（绿色虚线）为 v，角度为 β。 由正弦法则可得：

$$\begin{cases} \dfrac{\sin(\delta-\beta)}{l_f} = \dfrac{\sin(\frac{\pi}{2}-\delta)}{R} \\ \sin(\beta) = \dfrac{l_r}{R} \end{cases} \tag{3.1}$$

联立可得：

$$\tan(\delta)\cos(\beta) = \frac{L}{R} \tag{3.2}$$

Theta 为航向角，因为车辆后轮的偏角近似等于零，航向角的变化率等于车辆前轮对后轮的角速度：

$$\begin{cases} \dot{x} = v\cos(\theta+\beta) \\ \dot{y} = v\sin(\theta+\beta) \\ \dot{\theta} = \dfrac{v}{R} = \dfrac{v\tan\delta\cos\beta}{L} \\ \beta = \tan^{-1}(\dfrac{l_r\tan\delta}{L}) \end{cases} \tag{3.3}$$

为了简化计算和建模，本文以后轴中心为参考点，以单车模型为基础建立车辆的动力学模型，状态量为$[x,y,\theta]$ ，被控量为v ，此时滑移角β极小，可以假设为 0，车辆的动力学模型[35]可以简化为：

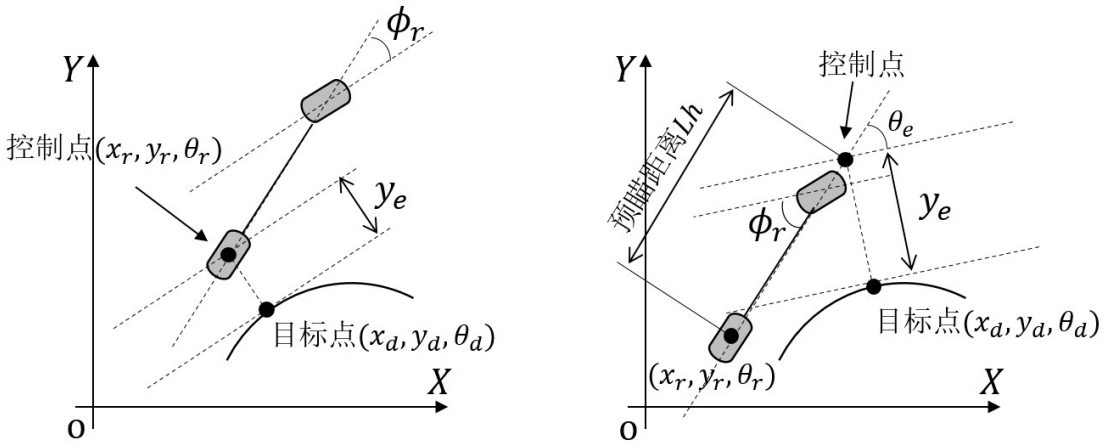
$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \tan\delta/L \end{bmatrix} [v] \tag{3.4}$$

在无人车的运动学模型计算中，一般采取速度v和角速度w作为输入，因此修改运动学模型为（其中$w = v\tan\delta / L$）：

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} [v] + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} [w] \tag{3.5}$$

3.2.2 预瞄误差模型

在本文的控制器设计过程中，采用预瞄模型来确定当前状态下的横纵向误差，从图 3.3 可以很容易获得车辆的横向误差向量，如式 3.6 所示：



（a）无预瞄模型　　（b）有预瞄模型

图 3.3 横向距离误差和横摆角误差模型

在路径跟踪中，横向控制的目标是确保车辆正确地沿着参考路径行驶。为了这个目的，需要最小化横向误差 y_e 和横摆角误差 θ_e。车辆的可行期望路径由仿真器中的轨迹规划器给出，图 3.3（a）所示无预瞄的情况下，其误差向量为：

$$\begin{bmatrix} y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} -\sin\theta_d & \cos\theta_d & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x_d \\ y_r - y_d \\ \theta_r - \theta_d \end{bmatrix} \tag{3.6}$$

其中 (x_r, y_r, θ_r) 指的虚拟的理想车的位置，横向误差 y_e 定义为车辆控制点与期望轨迹上最近点之间的距离。相应的误差变化率为：

$$\begin{cases} \dot{y}_e = v_r \cdot \sin\theta_e \\ \dot{\theta}_e = \dfrac{v_r}{l} \cdot \tan\phi_r \end{cases} \tag{3.7}$$

在车辆有预瞄距离的情况下，式 3.6 和 3.7 可以化为：

$$\begin{bmatrix} y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} -\sin\theta_d & \cos\theta_d & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x_d + Lh\cos\theta_r \\ y_r - y_d + Lh\sin\theta_r \\ \theta_r - \theta_d \end{bmatrix} \tag{3.8}$$

$$\begin{cases} \dot{y}_e = v_r \cdot \sin\theta_e + Lh\dfrac{v_r}{l}\tan\phi_r \cos\theta_e \\ \dot{\theta}_e = \dfrac{v_r}{l} \cdot \tan\phi_r \end{cases} \tag{3.9}$$

其中 v_d 和 ϕ_d 分别是对应的目标速度和目标前轮转角，在本文中假定转角误差的绝对值 $|\theta_e| < \pi/2$，这就意味着车辆的朝向与车辆轨迹的方向同向。

3.3 耦合滑模控制器设计

3.3.1 滑模控制概述

滑动模态控制（Sliding Mode Control， SMC）的主要特点表现在系统模型并不固定，可以在系统的动态变化过程中，根据系统当前的状态，有选择地不断变化，从而让系统沿预定的状态平面，也就是滑模面运动。由于滑模控制采用的控制方式是改变模型状态，它具有快速响应，强大地鲁棒性，实现简单等优点，但其最大地缺点就是抖振现象，即到达滑模面时难以收敛，从而在滑模面做来回穿越运动，即抖振运动。

自 1977 年，V.I.Utkin[36] 发表的一篇有关变结构的综述论文提出了滑模有关的控制方法以来，各国学者对变结构控制的研究兴趣急剧上升，在 K. D. Young 等的论文中对滑模控制以及其抖振现象进行了精确的评估和分析，并提出了其中可以有效抑制抖振现象的办法。中国学者高为炳院士等首次提出了趋近率的概念，聚焦于进入滑模面的运动，通过控制进入滑模面的运动，能有效的改善滑模运动的抖振现象。

滑模变结构控制策略因其控制输入的不连续性区别于常规的控制，即设计一个滑模面，通过

系统的开关和控制使得系统在滑模面附近做小幅高频的来回穿梭运动。因为该滑模面具有的可自行设计性，抗干扰性，所以处于滑模状态的系统具有良好的鲁棒性。

滑模控制的方法在解决非线性控制的综合问题时显得非常有效，但其控制量高频的抖振对其在实际的应用造成了极大的阻碍，因为物理的执行机构难以实现如此高频的抖动。因此如何改善滑模控制的抖振现象成为研究的重点和聚焦的地方。

3.3.2 滑模控制数学解析

在一阶系统中，状态方程可以表述为：

$$\dot{x} = f(x) \tag{3.10}$$

在该状态空间中自定义一个超平面 $S(x) = S(x_1, x_2, ..., x_n) = 0$， 如图 3.4 所示，在该平面附近点的运动状态有三种：穿越点、起始点、终止点，分别对应图中的 A，B，C 三点。点 A 穿过滑模平面继续向前，点 B 在滑模面附近受到排斥，从滑模面的两边远离滑模平面，点 C 在滑模面 S 的附近受到牵引，从滑模面的两边收敛到滑模平面。在滑模控制中，类似于点 C 这样的情况是有意义的。在设计滑模面的时候，其中一个条件就是保证滑模面周围的点都是终止点，因为在这样的条件下，趋近于该滑模面的运动点都将会被吸引在滑模面上运动，此时的状态就叫做滑动模态。

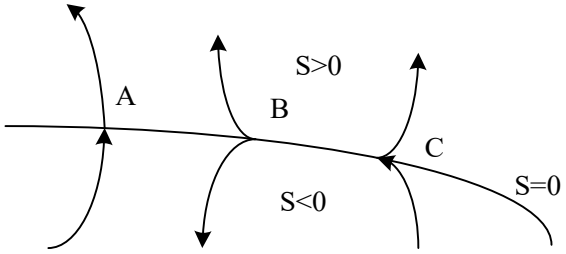


图 3.4 滑模面的原理

满足运动点是终止点的条件为，当运动点在切换面 $S(x) = 0$ 附近时，必有：

$$\lim_{s \to 0^+} \le 0 \&\& \lim_{s \to 0^-} \ge 0 \tag{3.11}$$

式 3.11 也可写成：

$$\lim_{s \to 0^+} S \cdot \dot{S} \le 0 \tag{3.12}$$

该不等式满足李雅普诺夫定理，该滑模面上的点是渐进稳定的[37][38]，即式 3.12 得到 S 的平方倒数是负半定的，系统自身稳定与条件 S=0。

3.3.3 耦合滑模横向控制器

根据车辆状态变化率的等式 3.9，经典的滑模控制面[39]可以定为：

$$S = \dot{y}_e + k y_e \le 0 \tag{3.13}$$

其中

$$y_e = y - y_{desired} \tag{3.14}$$

理想的滑模控制面应满足如下条件：

$$\begin{cases} S = \dot{y}_e + k y_e = 0 \\ \dot{S} = \ddot{y}_e + k \dot{y}_e = 0 \end{cases} \tag{3.15}$$

这也意味着所控制的车辆的状态轨迹应满足方程 $S = 0$，该条件也可用如下的数学表达式来表达：

$$S\dot{S} \le 0 \tag{3.16}$$

其物理意义可表述为，滑动模态存在于 $S = 0$ 这一平面上，其车辆的状态误差的导数指向该 S 平面，并使其收敛于 S 平面，从而状态误差收敛于零。

在车辆的横向控制和横向误差选择中，距离误差 y_e 和横摆角误差 θ_e 都代表了横向误差。本文改进了一个新的滑模面，将距离误差和横摆角误差耦合在一个滑模面中，因此只需要一个滑模面即可完成控制。而经典的滑模控制需要构造两个滑模面，并且需要保证横摆角误差的收敛速度大于距离的收敛速度，而这种经验性的构造并没有严格的数学保证。改进的耦合滑模面形式如下：

$$S = \dot{y}_e + k_0 y_e + k_1 \operatorname{sgn}(y_e)\theta_e \tag{3.17}$$

在车辆的横向控制中，只有一个横向输入变量车辆前轮转角，因此从理论上来说只能构建一个滑模面进行车辆的横向控制。在经典的滑模控制中，一般都采用 x_e 和 y_e 构建滑模面，通过这两个滑模面，求得一个车辆的速度方向，因此可得一个理想的车辆横摆角，再通过控制方法使得横摆角的收敛速度大于横向距离误差和纵向误差的收敛。而在本文改进的滑模面中，车辆横摆角和横向距离误差被耦合到一个滑模面中，因此直接采用该滑模面和纵向控制器即可完成车辆的控制。

车辆行进过程中，其状态变量会以特定速率抵达该滑模面，该特定速率又叫做趋近律或者控制律，即 $\dot{S}$。经典的控制律[39]如下形式所示：

$$\dot{S} = -Qg(S) - P\operatorname{sgn}(S) \tag{3.18}$$

其中 Q 和 P 是常参数，在趋近律中，Q 和 P 的不同选择指定了不同的到达速率并产生了不同的控制律结构。下面给出了式 3.16 的三个实例：

1. 恒定速率控制律

$$\dot{S} = -P\operatorname{sgn}(S) \tag{3.19}$$

这一定律迫使开关变量 S 以恒定速率 $|S_i| = -P_i, i = 1,2...$ 到达滑模面。

2. 比例速率控制律

$$\dot{S} = -QS - P\operatorname{sgn}(S) \tag{3.20}$$

通过加入比例速率项 $Q \cdot S$，当 S 较大时，状态量被迫更快地接近滑动模态面。

3. 指数控制律

$$\dot{S} = -Q|S|^{\alpha} \operatorname{sgn}(S) \tag{3.21}$$

该趋近律在状态远离滑模面时提高了到达速度，但在状态接近滑模面时降低了到达速度。本文选择了比例速率控制律，从式 3.17 和式 3.20 可以得到下述式 3.22：

$$\dot{S}=\ddot{y}_e+k_0\dot{y}_e+k_1\,\mathrm{sgn}(y_e)\theta_e=-QS-P\,\mathrm{sgn}(S) \tag{3.22}$$

结合前面的式 3.9，3.22，将式 3.22 左边化简，可得到车辆转角 ϕ_c 的控制方程：

$$\phi_c=\arctan\left(\frac{l}{v_r}\cdot\frac{-QS-P\,\mathrm{sgn}(S)+k_0v_r\sin\theta_e}{v_r\cos\theta_e+k_1\,\mathrm{sgn}(y_e)}\right) \tag{3.23}$$

而在车辆有预瞄的情况下，车辆转角 ϕ_c 的控制方程为：

$$\begin{aligned}\phi_c=&\frac{l\cos^2(\phi_r)}{v_rLh\cos\theta_e}\cdot\\&\left(-QS-P\,\mathrm{sgn}(S)-k_0\dot{y}_e-v_r\dot{\theta}_e\cos\theta_e+Lh\dot{\theta}_e^{\,2}\sin\theta_e-k_1\,\mathrm{sgn}(y_e)\dot{\theta}_e\right)\end{aligned} \tag{3.24}$$

因此将 $V=1/2\cdot S\cdot S$ 定义为李雅普诺夫函数，其导数为：

$$\begin{aligned}\dot{V}&=S\dot{S}\\&=S(-QS-P\,\mathrm{sgn}(S))\\&=-QS^2-P\frac{S^2}{|S|}\end{aligned} \tag{3.25}$$

因为 $\dot{V}$ 要求为负半定的函数，所以得到 $Q,P\geq 0$。

3.4 ADP 补偿控制器设计

3.4.1 动态规划原理

1957 年 Bellman 提出了一种求解最优控制问题的有效工具，动态规划（Dynamic programing，DP）方法[40]。动态规划的核心是贝尔曼最优性原理[41][42][43][44]，是求解最优化和最优控制问题的一个非常有用的工具。最优性原则表示为： 最优策略的属性是，无论初始状态和初始决策是什么，其余的决策必须构成与第一个决策产生的状态相关的最优策略。最优性原理可以应用于离散时间系统或连续时间系统、线性系统或非线性系统、时不变系统或时变系统、确定性系统或随机系统等。首先讨论一下非线性离散时间系统。假设有一个离散时间非线性系统，其系统动态方程为：

$$x(k+1)=F[x(k),u(k),k],k=0,1... \tag{3.26}$$

其中，$x\in R^n$ 为系统的状态向量，$u\in R^m$ 为控制向量，F 为系统函数，与该系统相关的代价函数为：

$$J[x(i),i]=\sum_{k=i}^{\infty}\gamma^{k-i}U[x(k),u(k),k] \tag{3.27}$$

其中初始状态 $x(k)=x_k$ 给定，U 是效用函数，γ 是在[0，1]之间的折扣因子。代价函数 J，也叫做 cost-to-go 函数，与初始时间 i 以及初始状态 $x(i)$ 有关。动态规划的目的是求解最优控制序列 $u(k),k=i,i+1,....$ 是代价函数 J 最小。根据贝尔曼最优性原理，第 k 时刻任意状态的最

小代价包括两部分，其中一部分是第 k 时刻内所需最小代价，另一部分是从第 k+1 时刻开始到无穷的最小代价累加和，即：

$$J[x^*(k)] = \min\{U[x(k),u(k)] + \gamma J[x^*(k+1)]\} \tag{3.28}$$

相应的第 k 时刻任意状态下最优控制策略可以表示为：

$$u^*[x(k)] = \arg\min_{u(k)}\{U[x(k),u(k)] + \gamma J[x^*(k+1)]\} \tag{3.29}$$

式 3.28 式离散时间系统的优化准则，它允许通过反向传播算法对变量进行优化。而非线性连续系统的系统状态可以表示为：

$$\dot{x}(t) = F[x(t),u(t),t], t \geq t_0 \tag{3.30}$$

其代价函数定义为：

$$J[x(t),t] = \int_t^{\infty} U[x(\tau),u(\tau)]d\tau \tag{3.31}$$

对于连续时间系统，也可以应用 Bellman 的最优性原理来求解。最优代价 $J^*[x_0] = \min\{J[x_0,u(t)]\}$ 满足哈密顿-雅可比-贝尔曼方程[45]：

$$-\frac{\partial J^*[x(t)]}{\partial t} = \min_{u\in U}\{U[x(t),u(t),t] + \left(\frac{\partial J^*[x(t)]}{\partial x(t)}\right)^T \times F[x(t),u(t),t]\} \tag{3.32}$$

即：

$$-\frac{\partial J^*[x(t)]}{\partial t} = U[x(t),u^*(t),t] + \left(\frac{\partial J^*[x(t)]}{\partial x(t)}\right)^T \times F[x(t),u(t),t]\} \tag{3.33}$$

式 3.28 和 3.33 称为动态规划的最优性方程，是实现动态规划的基础。在上面的例子中，如果已知式 3.26 或 3.30 中的函数 F 和式 3.27 或 3.31 中的代价函数 J，那么求解 $u(k)$ 就是一个简单的最优化问题。如果系统是线性的，且代价函数为状态量和控制量的二次函数，则最优控制策略为状态的线性反馈，可通过求解标准的黎卡提方程[46]得到。而如果系统是非线性系统或代价函数为非二次型函数，则最优状态反馈控制需要取求解哈密顿-雅可比-贝尔曼（HJB）方程的解，该方程通常为非线性偏微分方程或差分方程，其求解是十分困难的。此外，随着状态变量和输入变量维数的增加，DP 方法对计算量和存储量有十分高的要求，这就是众所周知的维度灾难问题的结果[47][48]。多年来，人们已经取得了一些进展，通过构建一个称为批评家的系统来规避维数的诅咒，以接近动态规划中的成本函数。其思想是通过使用函数近似结构（如神经网络、多项式等）近似代价函数来近似动态规划解，这就是所谓的自适应动态规划方法（ADP）。

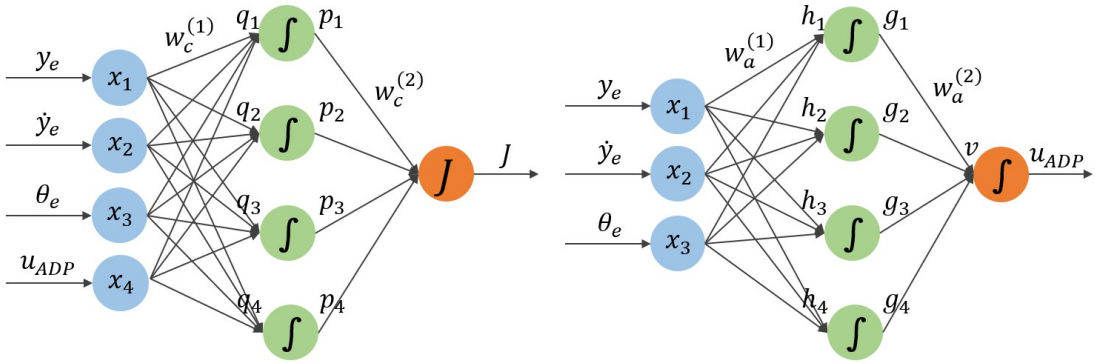
3.4.2 ADP 补偿控制器

自适应动态规划（ADP）算法一般由两个主要部分组成： 控制网络（action network）和评价网络（critic network）。控制网络根据映射网络所代表的学习策略生成控制信号μ，而评价网络则用来近似表示动态规划中 Bellman 方程的代价函数 J，图 3.1 蓝色部分给出了本文 ADP 算法[49][50][51]的基本原理图。当前跟踪误差 $\Delta\Omega = (y_e, \dot{y}_e, \theta_e)$ 为控制网络的输入，该网络的输出为自适

应的补偿控制信号车辆转角。强化信号 $r(t)$ 来源于被控制的车辆，如式 3.56 所示。

在该在线学习补偿控制器中，控制器刚开始的时候，动作网络和批评网络的权值/参数都是随机初始化的。一旦其观察到系统的状态，随后它将根据网络中参数进行操作。其参数在不断操作的过程中会被自定义的强化信号调整从而调整控制值，从而满足最优性原理方程。

下面将详细讨论控制网络和评价网络的组成，以及 ADP 具体是如何学习的。在本文的设计中，控制网络和评价网络都是非线性多层前馈网络。



（a） 评价网络　　（b） 控制网络

图 3.5 ADP 网络结构

A. 评价网络

图 3.5（a）给出了本文设计的评论家网络。它是一个含有 4 个隐藏神经元的三层前馈神经网络，它的输入向量为系统跟踪误差以及控制网络的输出 u_{ADP} ，输出向量是一个近似的代价函数 $J(t)$，它的输入形式如式 3.34，3.35，3.36 所示：

$$J(t)=\sum_{i=1}^{N_h} w_{c_i}{}^2(t)\,p_i(t) \tag{3.34}$$

$$p_i(t)=\frac{1-\exp^{-q_i(t)}}{1+\exp^{-q_i(t)}},\quad i=1,\ldots N_h \tag{3.35}$$

$$q_i(t)=\sum_{j=1}^{n+1} w_{c_{ij}}{}^{(1)}(t)x_j(t),\quad i=1,\ldots N_h \tag{3.36}$$

其中 q_i，p_i 为第 i 个隐藏神经元的输入值和输出值，N_h 为神经元的数量，在本文中为 4，n+1 为评价网络的输入变量的数目（包括 u_{ADP} ）。

代价函数 $J(t)$ 需要近似从时间 t 开始的总回报，具体地说， $R(t)$ 可以表示为：

$$R(t)=\sum_{k=1}^{\infty}\alpha^{k-1}r(t+k) \tag{3.37}$$

式中 $R(t)$ 是从时间 t 开始的未来累计的代价函数，α 是时间折扣常数（0<α<1），在本文设计的控制器中，时间折扣常数 α 被选为 0.95。$r(t+k)$ 表示在 $t+k$ 时刻的外部强化信号。

为了近似代价函数$J(t)$，评价网络的预测误差被定义为：

$$e_c(t)=\alpha J(t)-[J(t-1)-r(t)] \tag{3.38}$$

在训练过程需要最小化的目标函数$E_c(t)$为：

$$E_c(t)=\frac{1}{2}e_c{}^2(t) \tag{3.39}$$

通过梯度下降算法，得到评判网络的权值更新规则为：

$$w_c(t+1)=w_c(t)+\Delta w_c(t) \tag{3.40}$$

$$\Delta w_c(t)=l_c(t)\left[-\frac{\partial E_c(t)}{\partial w_c(t)}\right] \tag{3.41}$$

$$\frac{\partial E_c(t)}{\partial w_c(t)}=\left[-\frac{\partial E_c(t)}{\partial J(t)}\cdot\frac{\partial J(t)}{\partial w_c(t)}\right] \tag{3.42}$$

其中$l_c(t)>0$和$w_c(t)$分别为批评家网络在 t 时刻的学习速率和权重向量，根据链式法则，权重的更新如式 3.43，3.44 所示：

$$\Delta w_{c_i}{}^2=l_c(t)\cdot\left[-\frac{\partial E_c(t)}{\partial J(t)}\cdot\frac{\partial J(t)}{\partial w_{c_i}{}^{(2)}(t)}\right]=\alpha e_c(t)p_i(t) \tag{3.43}$$

$$\begin{aligned}\Delta w_{c_{ij}}{}^{(1)}(t)&=l_c(t)\cdot\left[-\frac{\partial E_c(t)}{\partial J(t)}\cdot\frac{\partial J(t)}{\partial p_i(t)}\cdot\frac{\partial p_i(t)}{\partial q_i(t)}\cdot\frac{\partial q_i(t)}{\partial w_{c_{ij}}{}^{(1)}(t)}\right]\\&=\alpha e_c(t)p_i(t)w_{c_i}{}^{(2)}(t)\left[\frac{1}{2}(1-p_i{}^2(t))\right]x_j(t)\end{aligned} \tag{3.44}$$

B. **控制网络**

图 3.5（b）为控制网络，它同样是含有 4 个隐藏神经元的三层前馈神经网络，但其输出需要再经过一次非线性变换，控制网络的输入跟踪误差向量$\Delta\Omega=(y_e,\dot{y}_e,\theta_e)$，其输出是对车辆的补偿控制信号$u_{ADP}$，它的相关方程为：

$$u_{ADP}(t)=\frac{1-\exp^{-v(t)}}{1-\exp^{-v(t)}} \tag{3.45}$$

$$v(t)=\sum_{i=1}^{N_h}w_{a_i}{}^{(2)}(t)g_i(t) \tag{3.46}$$

$$g_i(t)=\frac{1-\exp^{-h_i(t)}}{1+\exp^{-h_i(t)}},\qquad i=1,....,N_h \tag{3.47}$$

$$h_i(t)=\sum_{j=1}^{n}w_{a_{ij}}{}^{(1)}(t)x_j(t),\qquad i=1,....N_h \tag{3.48}$$

其中v为执行节点的输入值，g_i，h_i为第 i 个隐藏神经元的输入值和输出值，N_h神经元的数

量，在本文中为 4，n 为控制网络的输入变量的数目（不包括 u_{ADP} ）。动作网络被训练来间接反向传播期望的最终目标 U_c 和来自评价网络的近似成本函数 J 之间的误差。动作网络的训练通过最小化函数 $E_a(t)$ 来更新权值：

$$e_a(t) = J(t) - U_c(t) \tag{3.49}$$

$$E_a(t) = \frac{1}{2} e_a{}^2(t) \tag{3.50}$$

动作网络的权值更新规则如下：

$$w_a(t+1) = w_a(t) + \Delta w_a(t) \tag{3.51}$$

$$\Delta w_a(t) = l_a(t)\left[-\frac{\partial E_a(t)}{\partial w_a(t)}\right] \tag{3.52}$$

其中 $l_a(t) > 0$ 和 $w_a(t)$ 分别为动作网络在 t 时刻的学习速率和权向量，$U_c(t)$ 为最终的目标，在本文中被设为零。根据链式法则，权重的更新如式 3.53，3.54 所示：

$$\begin{aligned}\Delta w_{a_i}^{(2)}(t) &= l_a(t)\cdot\left[-\frac{\partial E_a(t)}{\partial J(t)}\frac{\partial J(t)}{\partial u(t)}\frac{\partial u(t)}{\partial v(t)}\frac{\partial v(t)}{\partial J(t)}\right] \\ &= e_a(t)\left[\frac{1}{2}(1-u^2(t))\right] g_i(t)\cdot\sum_{i=1}^{N_h}\{w_{c_i}^{(2)}(t)\left[\frac{1}{2}(1-p_i^2(t))\right] w_{c_{i,n+1}}^{(1)}(t)\}\end{aligned} \tag{3.53}$$

$$\begin{aligned}\Delta w_{a_{ij}}^{(1)}(t) &= l_a(t)\cdot\left[-\frac{\partial E_a(t)}{\partial J(t)}\frac{\partial J(t)}{\partial u(t)}\frac{\partial u(t)}{\partial v(t)}\frac{\partial v(t)}{\partial g_i(t)}\frac{\partial g_i(t)}{\partial h_i(t)}\frac{\partial h_i(t)}{\partial w_{a_{ij}}^{(2)}(t)}\right] \\ &= e_a(t)\left[\frac{1}{2}(1-u^2(t))\right] w_{a_i}^{(2)}(t)\left[\frac{1}{2}(1-p_i^2(t))\right]\cdot x_j(t) \\ &\quad\cdot\sum_{i=1}^{N_h}\{w_{c_i}^{(2)}(t)\left[\frac{1}{2}(1-p_i^2(t))\right] w_{c_{i,n+1}}^{(1)}(t)\}\end{aligned} \tag{3.54}$$

如上所述，ADP 控制器的输入设计如下：

$$\Delta\Omega = (y_e, \dot{y}_e, \theta_e) \tag{3.55}$$

ADP 的主要强化信号控制器设计如下[50][51]：

$$\begin{cases} r(t) = -\Delta\Omega(t) * Q * \Delta\Omega(t)^T \\ Q = diag\left(\left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]\right) \end{cases} \tag{3.56}$$

4 仿真平台评估及控制算法分析

4.1 仿真平台评估

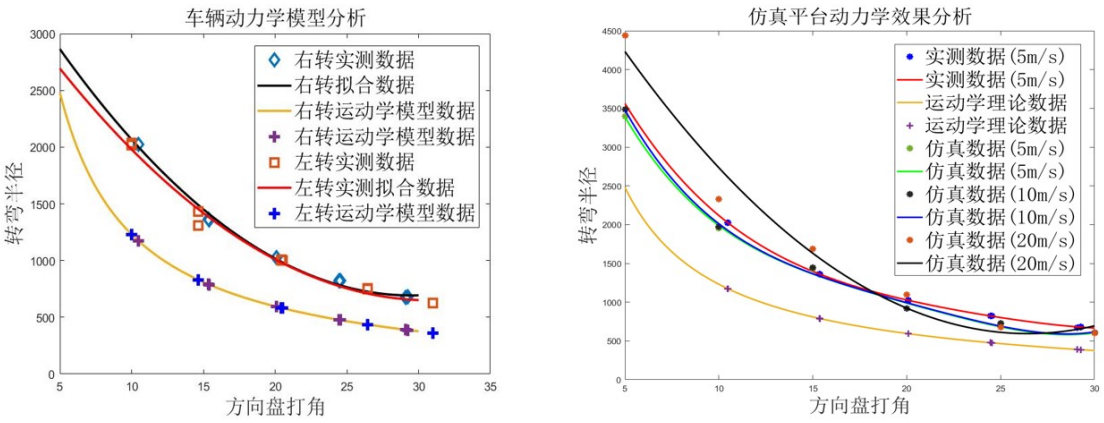
仿真平台的评估应该包括仿真的可信度和可靠性。本章从车辆的动力学模型、底盘控制器性能、以及轨迹规划器的影响三个部分来对该仿真平台进行评估，并设计了与之相对应的实验。每一个场景都设计了合适的场景，包括不同的控制器、轨迹类型、以及车辆的初始位置等，如表 4.1 所示。每一个场景实验都会在下列的章节里展开详述。

表 4.1 场景实验配置

实验目标	测量数据	控制器	轨迹
动力学模型分析	转弯半径	底盘控制器	圆弧
底盘控制器分析	速度	底盘控制器	直线
轨迹算法分析	车辆轨迹形状	横纵向控制器	三次样条轨迹 Catmul-Rom 轨迹 B 样条轨迹

4.1.1 动力学模型分析

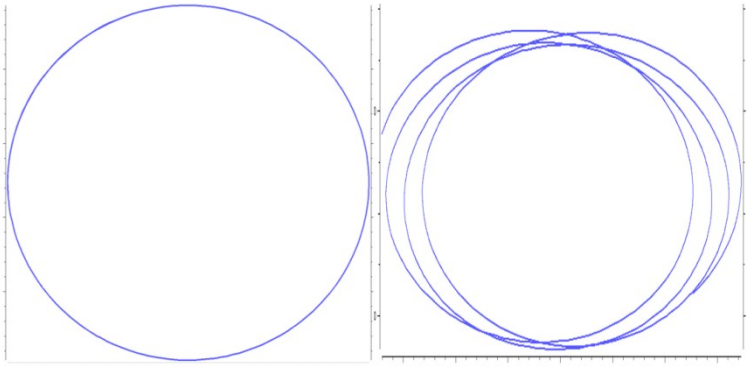
本节中分析了车辆动力学模型参数矫正的效果，验证了仿真平台的车辆物理反应的真实性。测量车辆的动力学反应是一件极为复杂且繁琐的工作，为了在获得有效数据的基础上简化测量过程，本文在车辆恒定打角的情况下，测量了车辆的转弯半径，并以此比较车辆的横向动力学反应。此外，本文还测量了不同速度下仿真平台中车辆的转弯半径，并进行了分析对比。如图 4.1（a）所示，测量的数据为实际车辆的转弯半径和车辆运动学理论转弯半径。当方向盘打角较小的时候，实车测量的数据与运动学理论数据较为接近，但整体而言，车辆运动学模型与实际车辆的物理反应，还存在不小的误差，因此在仿真平台中，无法使用运动学模型来仿真车辆的反应。



（a） 实际车辆与运动学理论模型对比　　（b）仿真车辆与实际车辆对比

图 4.1 动力学模型分析

如图 4.1（b）所示，本文对仿真平台环境下，车辆速度分别为 5m/s， 10m/s， 20m/s 的情况下，车辆在不同方向盘打角下的转弯半径进行了测量，实际车辆由于考虑到安全原因，则只对 5m/s 的情况进行了测量。从图中可以看出，车辆在速度为 5m/s，10m/s 的情况下，其车辆的转弯半径极为接近实际所测得的数据，从而证明了该仿真平台车辆动力学的真实性和可信度。但是，车辆在 20m/s 的速度情况下，与所得实际情况有较大的偏差，这是因为在高速的情况下，车辆的物理模型为高度非线性化，其动力学曲线已经发生了改变，如图 4.2（a）（b）所示，车辆高速下的转弯轨迹有明显的失真。这也证明该仿真平台对高速车辆下的仿真具有一定的真实性。

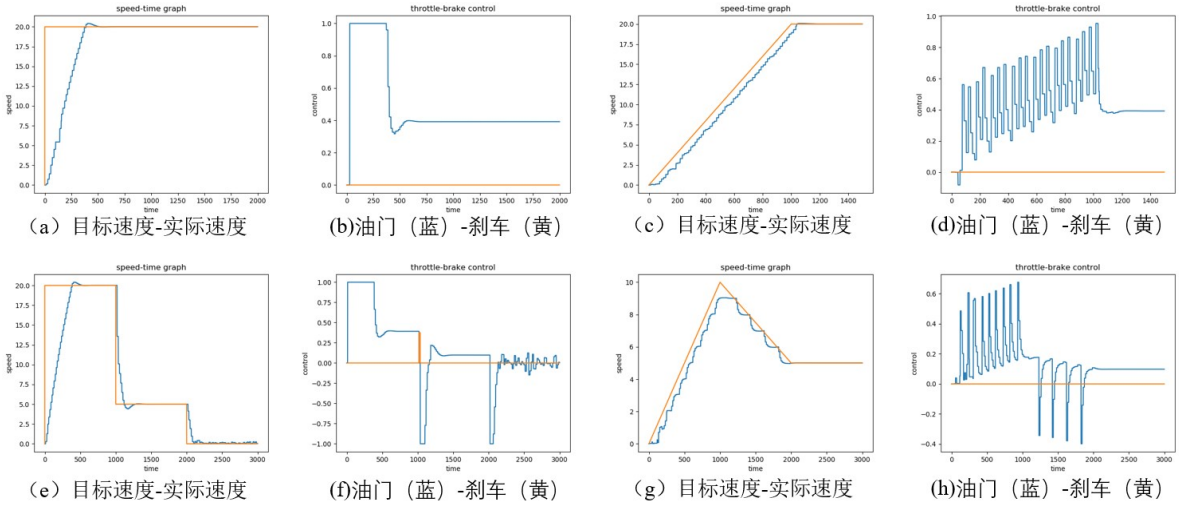


（a） 车速 5m/s （b）车速 20m/s

图 4.2 恒定打角下仿真车辆轨迹

4.1.2 底盘控制器性能分析

为了更加真实的模拟驾驶车辆的控制情况，本文根据模糊规则设计了油门刹车模拟控制器，本节通过仿真验证了四种不同的速度误差条件下刹车油门的协调控制规则，并分析了在不同控制规则下底盘控制器的控制效果，四种不同的误差条件如表 4.2 所示，相对应的控制器效果如图 4.3 所示：



（a）目标速度-实际速度 (b)油门（蓝）-刹车（黄） （c）目标速度-实际速度 (d)油门（蓝）-刹车（黄）

（e）目标速度-实际速度 (f)油门（蓝）-刹车（黄） （g）目标速度-实际速度 (h)油门（蓝）-刹车（黄）

图 4.3 控制器分析效果图

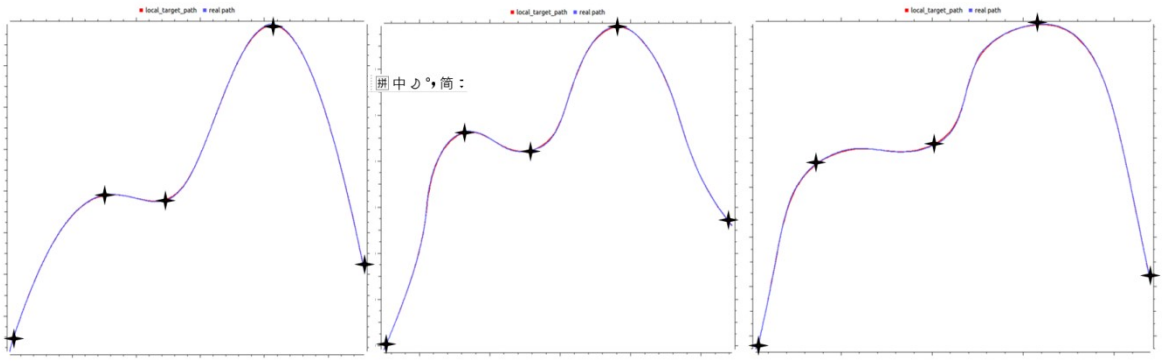
表 4.2 刹车-油门验证场景参数

场景	速度误差值 m/s	是否连续	效果图	PID 参数
1	20	否	（a）（b）	（5,0.1,0.1）
2	20	是	（c）（d）	（5,0.1,0.1）
3	-15/5	否	（e）（f）	（5,0.1,0.1）
4	10/-5	是	（g）（h）	（5,0.1,0.1）

在场景一中，给定一个确定的目标速度，底盘较准确的跟踪到了给定的目标速度，且其油门的变化足够平缓。在场景二中，目标速度从零缓慢上升至目标速度，在此场景中，车辆依然能够较稳定跟踪到目标速度，但其油门值突变较大，容易引起乘客不适，也容易造成车辆损坏。造成这种情况的原因为底盘控制器无法预知目标速度的变化趋势，即无法预瞄，因此可采用滤波的方法解决这一问题。在场景三中，目标速度从 20m/s 迅速减为 5m/s， 在此种情况下，采用先刹车后油门控制的方法，当启用刹车使速度误差减小到小于临界值时，停止刹车，启用油门控制。当目标速度从 5m/s 减为 0m/s 时，由于速度误差绝对值较小，直接采用油门控制。在场景四中，目标速度由 10m/s 缓慢减至 5m/s， 但由于无法得知目标速度的变化，其油门变化频率太快，该控制器性能不理想。

4.1.3 轨迹规划器

一个良好的轨迹规划器是控制器中必不可少的一环，因此研究控制器在不同的轨迹下的跟踪效果很有意义。本文在搭建的滑模控制器的基础上分别研究了三种轨迹的跟踪效果，证明了该仿真平台此项功能的必要性。



（a） B 样条曲线 （b）三次样条曲线 （c） Catmul_Rom 曲线

图 4.4 车辆轨迹图

如图 4.4 所示，同样的控制在不同的轨迹曲线上起到的跟踪效果略有不同，红色为目标轨迹，蓝色为实际轨迹。这三种曲线的效果虽然相差无几，但是却是因为此三种曲线都是比较符合汽车动力学特性的曲线。如果换一个没有规划过的简单的线段相连接，则车辆控制器将会难以跟踪。最终，我们选择了 Catmul_Rom 曲线作为控制器的后续测试曲线。

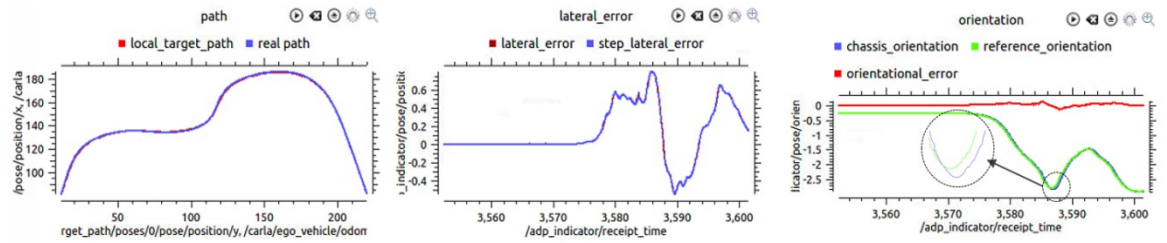
4.2 控制算法分析

4.2.1 耦合滑模控制

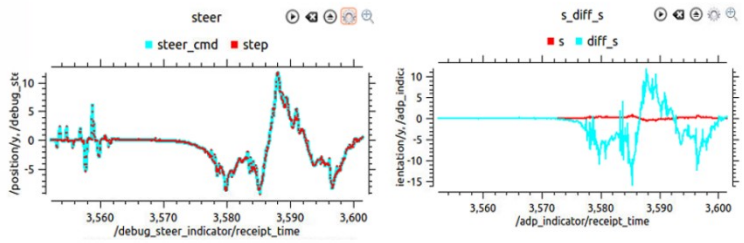
在本节中，对第三节改进的耦合滑模控制方法在第二节所搭建的仿真平台进行了验证和分析。轨迹由第二节中所开发的轨迹规划器通过 Catmul_Rom 曲线生成，如图 4.4（a）所示。为了验证高速大曲率下控制器的性能，该轨迹的目标速度被设为 20m/s， 最小曲率半径被设为 25m。式 3.17，3.18 中的参数被选取 $k_0=0.5, k_1=0.05, Q=P=10$。式 3.17，3.20 中的符号函数 sgn 被换成了阈值为 0.5 的饱和函数，从而达到减少抖振的效果。预瞄距离 Lh 被设置为 0.8m。

A. 情况一：无初始误差

图 4.5-4.6 展示了本文改进的耦合滑模面控制器和 MPC 控制器[52]在无初始位置误差的轨迹跟踪控制结果。在初始位置无误差的情况下，所设计的滑模控制器在跟踪效果上要优于 MPC 控制器，其基本能跟踪到目标曲线，其距离误差和横摆角误差也分别控制在 $\pm 0.6m$ 和 $\pm 0.2rad$ 以内。而线性 MPC 控制器在此种情况下展现出了较大的跟踪误差，距离误差最大处达 $-6m$。但该滑模控制器在已经加入平滑滤波，限制转角急速变化的情况下，仍展现出了较大的抖振现象。



（a）目标轨迹（红），实际轨迹（蓝） （b）横向距离误差 （c）朝向误差



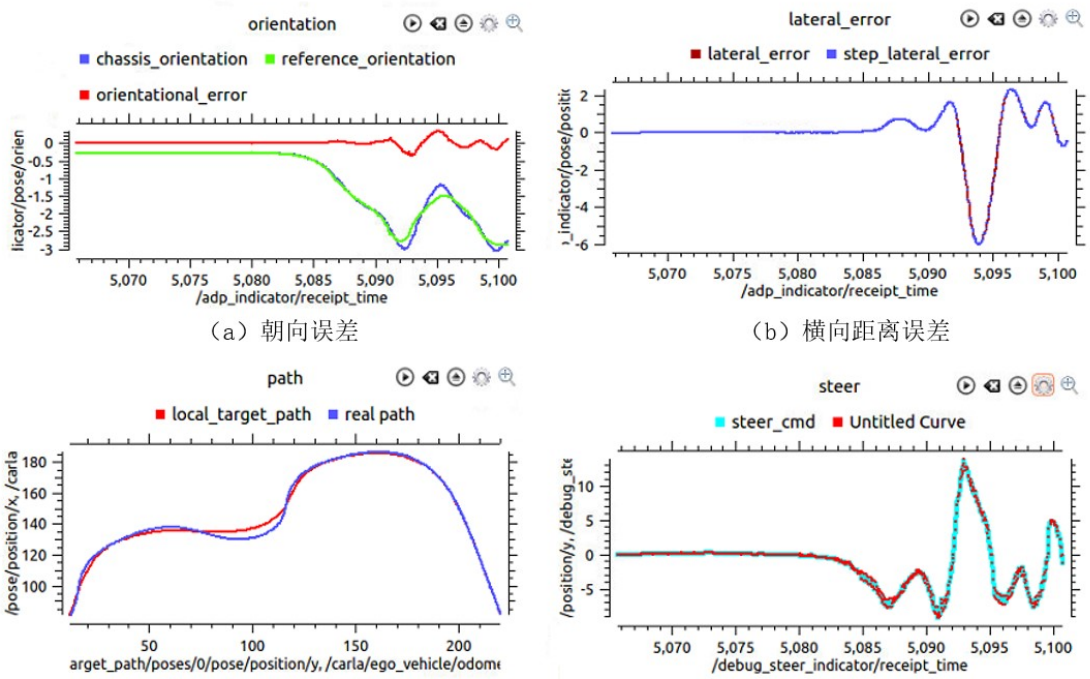
（d）方向盘打角 （d）滑模面（红），滑模面导数（绿）

图 4.5 滑模控制器

B. 情况二：初始横摆角误差

图 4.7 展示了在初始误差仅有横摆角误差的情况下，所设计滑模控制器以及 MPC[52]控制器的跟踪效果、横向误差以及横摆角误差图。为了更加方便的比较，图 4.7 均是选取了该段轨迹运行的一部分放大图。

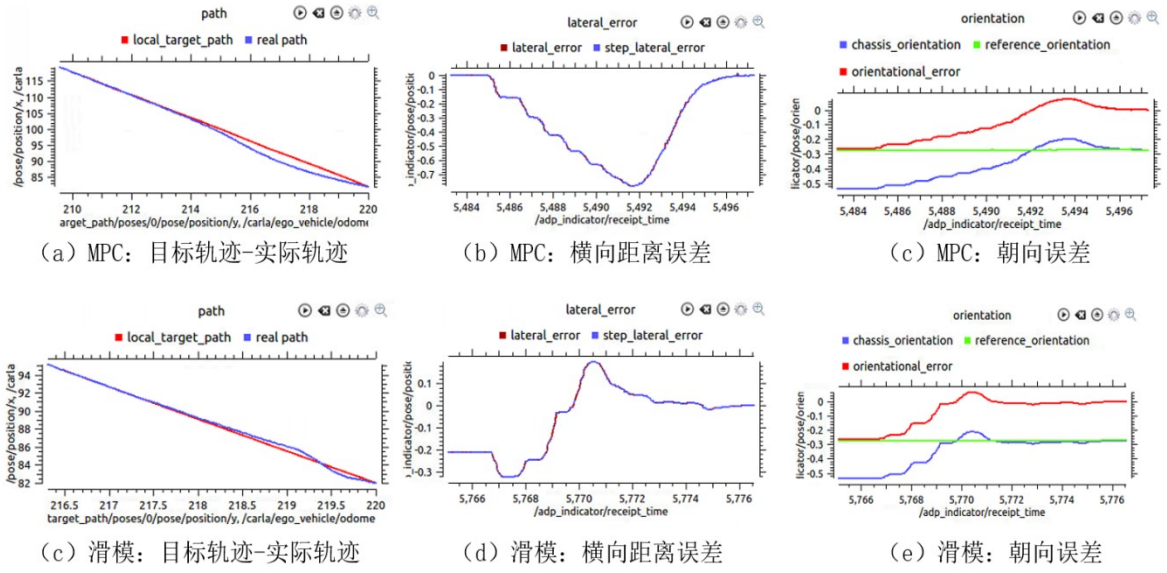
在有初始横摆角误差的情况下，MPC 控制器能够在第二次到达目标轨迹即将横摆角误差收敛至零，而滑模控制器则需要来回两次才能收敛至目标轨。而 MPC 控制器的横向误差总平均值却大于滑模控制器，但这种收敛方式无疑更符合人类的舒适感需求。



（a）朝向误差　（b）横向距离误差

（c）目标轨迹（红），实际轨迹（蓝）　（d）方向盘打角

图 4.6 MPC 控制器



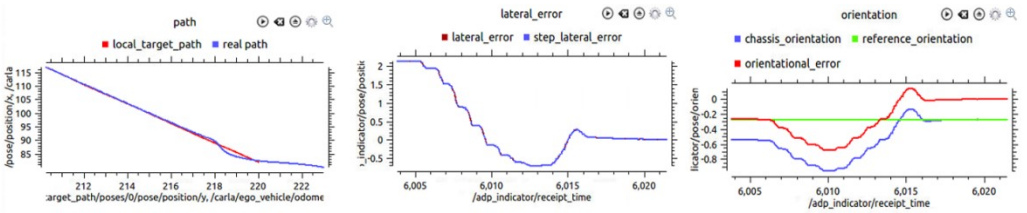
（a）MPC：目标轨迹-实际轨迹　（b）MPC：横向距离误差　（c）MPC：朝向误差

（c）滑模：目标轨迹-实际轨迹　（d）滑模：横向距离误差　（e）滑模：朝向误差

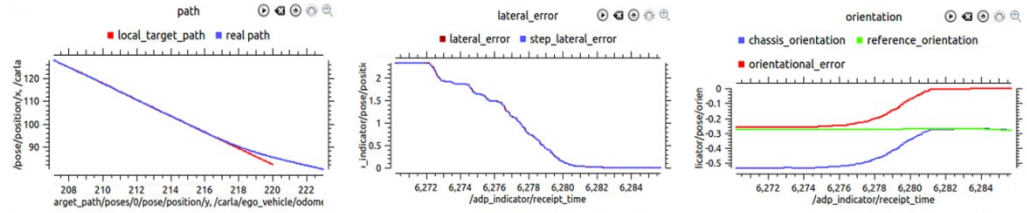
图 4.7 滑模 vs MPC 控制器（有初始横摆角误差）

C. 情况三：初始横摆角误差+距离误差

图 4.8 展示了在初始误差存在横摆角误差和距离误差的情况下滑模控制器的跟踪效果，并与 MPC 控制器进行了对比。滑模控制器以更大的总横摆角误差和横向距离误差的两次摆动为代价换的比之 MPC 更快的收敛速度和更小的总横向距离误差。这种特性也限制了滑模控制器在更多场景的应用。



（a）MPC：目标轨迹-实际轨迹　（b）MPC：横向距离误差　（c）MPC：朝向误差



（c）滑模：目标轨迹-实际轨迹　（d）滑模：横向距离误差　（e）滑模：朝向误差

图 4.8 滑模 vs MPC 控制器（有初始横摆角误差和距离误差）

4.2.2 ADP 补偿控制

为了解决滑模控制器控制精度以及控制输入抖振现象严重的问题，本文尝试引入了自适应动态规划这一在线学习算法，它通过过去输入的表现评估，来实现对某一个输入所有未来的奖励值的预测。自适应补偿控制分为两部分，第一部分为代价函数的估计，第二部分为根据代价函数的变化调整控制网络控制补偿输入。其代价函数的估计效果如图 4.9 所示：

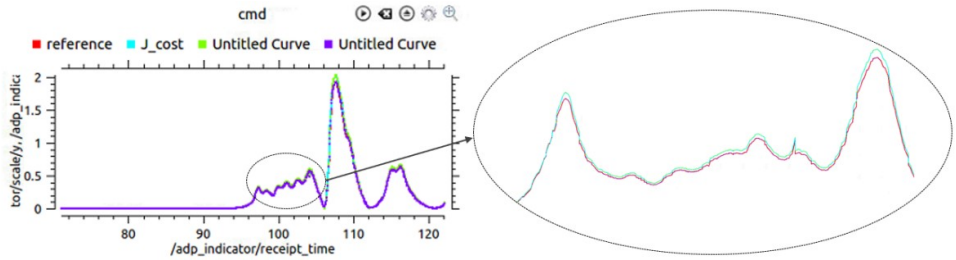
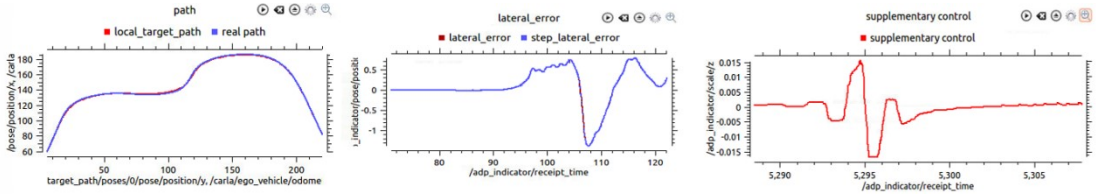


图 4.9 代价函数的估计效果

图 4.9 展现出本文中的评价网络良好的性能。图 4.10 为加入补偿控制后，滑模控制器的改善效果。从图中可以看出，其横向误差相较于纯粹的滑模控制有所改善，但效果不是很明显。猜测其原因可能是无法像 MPC 控制器能够制定优化目标函数，但是这证明了自适应动态规划这一自主学习的方法可以被用来辅助控制。



（a）目标轨迹-实际轨迹　（b）横向距离误差　（c）补偿控制量

图 4.10 ADP-滑模控制器效果

5 结论和展望

5.1 结论

无人驾驶控制算法的研究因其危险性和不确定性必须要求先在仿真平台中模拟数次。本文针对无人驾驶控制仿真平台搭建、横向控制算法的设计等方面做了深入研究，并取得了一下研究成果：

（1）搭建了无人驾驶控制仿真平台，在 CARLA 模块的基础上搭建了新的虚拟仿真场景、校准了车辆的动力学模型、设计新的底盘控制等；设计了具有三种插值曲线拟合算法的轨迹规划器，使用户能够通过可视化操作不同的仿真轨迹；开发了包括仿真动画显示效果和仿真数据可视化工具在内的可视化模块；利用 ROS 将控制器、规划器、显示模块、CARLA 模块相连接，使控制器能一键部署至实车上。

（2）设计了无人驾驶横纵向控制策略模块化系统，并在车辆运动学模型的基础上，设计了基于耦合滑模面的横向控制方法，并在搭建的仿真平台与 MPC 算法进行了对比，验证了该算法的有效性和优越性，其满足无人驾驶运动控制的要求。

（3）在耦合滑模横向控制器的上，改进以自适应动态规划的方法来尝试改进该滑模控制器的精度和优化某些变量。最后通过实验，验证了该自主学习方法的可行性和潜力，证明了该方法能够让车辆具备自主学习的能力。

5.2 展望

本文对无人驾驶仿真平台开发以及车辆横向控制算法进行了研究，在此过程中，需要进一步展开的研究包括：

（1）搭建的仿真平台虽然操作简单，动画逼真，动力学模型较为精准，具备仿真多种路径的能力，但其图形化界面对显卡的要求比较高，并且不支持大规模多车协同仿真，后期可加入多车控制仿真的功能，以及构建更丰富的虚拟仿真地图，比如高速公路等，以满足不同用户的需求。

（2）设计的底盘控制器虽然具有较好的跟踪性能，但控制量的变化不够平滑，需要做进一步的优化，使其更加平滑。

（3）设计的横向耦合滑模控制算法虽然能使两个变量均收敛置零，但不同情况下可考虑不同的耦合方式，以获得最好的控制效果。滑模控制算法的鲁棒性十分好，收敛速度十分快，但却是在牺牲了控制量的平滑性的基础上达到的。在进一步的研究中，可以考虑和 MPC 控制算法相结合，将未来的时刻也加入考虑范围，从而达到更好的控制效果。

（4）改进的自适应动态规划方法，虽然证明其可行性，但是从目前来看，效果并不明显。在未来的工作中，应考虑重新定义代价函数，代价函数的定义应该和滑模控制器的特性相结合，这样才能更加有效的改善滑模控制器。

参考文献

[1] 翁莎. MATLAB：构建无处不在的自主技术[J]. 汽车与配件, 2017(20):54-54.

[2] 许占奎. 无人驾驶汽车的发展现状及方向[J]. 科技展望, 2015(32):231-232.

[3] 刘小辉. DARPA 开展“机器人挑战赛”[J]. 国外坦克(4):46-48.

[4] 靳欣宇，张军，刘元盛等. 基于 Stanley 算法的自适应最优预瞄模型研究[J]. 计算机工程, 044(7):42-46.

[5] 杨帆. 无人驾驶汽车的发展现状和展望[J]. 上海汽车, 2014(3):35-40.

[6] Huamei Cui, Quan Chen, Xing Qi,等. Electric vehicle differential system based on co-simulation of Carsim/Simulink[C]. 2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA): IEEE, 2016.

[7] David J. Roberts, Robin Wolff, Oliver Otto et al. Constructing a Gazebo: Supporting Team Work in a Tightly Coupled, Distributed Task in Virtual Reality[J]. Presence, 2003, 12(6):644-657.

[8] Hoffmann G M, Tomlin C J, Montemerlo M, et al. Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing[C]. American Control Conference. IEEE, 2007.

[9] Kranz T, Hahn S, Zindler K. Nonlinear lateral vehicle control in combined emergency steering and braking maneuvers[C]. 2016 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2016.

[10] Park H, Gerdes J C. Optimal tire force allocation for trajectory tracking with an over-actuated vehicle[C]. Intelligent Vehicles Symposium. IEEE, 2015.

[11] Park J B, Bae S H, Koo B S, et al. When path tracking using look-ahead distance about the lateral error method and the velocity change method tracking comparison[C]. 2014 14th International Conference on Control, Automation and Systems (ICCAS). IEEE, 2014.

[12] Park M W , Lee S W , Han W Y . Development of lateral control system for autonomous vehicle based on adaptive pure pursuit algorithm[C]. 2014 14th International Conference on Control, Automation and Systems (ICCAS). IEEE, 2014.

[13] 廖煜雷，庄佳园，李　晔，等. 欠驱动无人艇轨迹跟踪的滑模控制方法[J]. 应用科学学报，2011， 29(4):1179-1193.

[14] 郭一军，俞　立，徐建明. 基于扩张状态观测器的轮式移动机器人抗饱和自适应滑模轨迹跟踪控制[J]. 系统科学与数学，2017，37(5):1179-1193.

[15] 杨兴明，李文静，朱建. 基于 RBF 神经网络的机器人的路径跟踪控制[J]. 合肥工业大学学报: 自然科学版， 2015，38(11):1477-1483.

[16] Behzadan V , Minton J , Munir A . TrolleyMod v1.0: An Open-Source Simulation and Data-Collection Platform for Ethical Decision Making in Autonomous Vehicles[J]. 2018.

[17] Kei, OKADA. Recent Status and Future Prospects of ROS (Robot Operating System)[J]. Journal of the Society of Instrument & Control Engineers, 2018.

[18] Wikipedia.Anti-rollbar[EB/OL].(2012-12-09)[2014-12-1].https://en.wikipedia.org/wiki/Antiroll_

bar.

[19] Locost Intro. Tire Modeling[EB/OL]. (2008-3-23) [2009-5-14]. https://loco st7.info/.

[20] Fuso. Engine Torque Curves[EB/OL]. (2005-8-15) [2008-3-13]. https://www.mitsubishi-fus.

[21] Karim Nice, Charles W.Bryant. How Clutchs Work[EB/OL]. (2003-9-15)[2004-3-21]. https://auto.howstuffworks.com/clutch.htm.

[22] WikiHow. How to Drive an Abbot SPG (Tank) [EB/OL]. (20012-5-13)[2014-12-1]. https://www.wikihow.c om/Drive-an- Abbot-SPG-%28Tank%29.

[23] Karim Nice. How Differentials Work[EB/OL]. (2009-6-30)[2012-3-16]. https://auto.howstuffworks.com/differential.htm.

[24] PhysX Vehicle. NVIDIA PhysX SDK 3.4.0 Documentation[EB/OL]. (2001-5-21)[2003-2-08]. https://auto.works.com/differential.htm.

[25] Unreal Engine. Vehicle Art Setup[EB/OL]. (2010-4-23)[2011-4-23]. https://docs.unrealengine.com/enUS/Engine/Physics/Vehicles/VehicleContentCreation/index.html.

[26] 吴宪祥, 郭宝龙, 王娟. 基于粒子群三次样条优化的移动机器人路径规划算法[J]. 机器人，2009， 31(6):556-560.

[27] Zhang S , Deng W , Zhao Q , et al. Dynamic Trajectory Planning for Vehicle Autonomous Driving[C]. Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics. IEEE, 2013.

[28] 何鹏, 高峰, 魏厚敏. 基于 Catmull-Rom 样条曲线的弯曲车道线检测研究[J]. 汽车工程学报，2015(04):46-51.

[29] Pacheco R R , Hounsell M D S , Junior R S U R , et al. Smooth Trajectory Tracking Interpolation on a Robot Simulator[C]. IEEE Latin American Robotics Symposium & Intelligent Robotics Meeting. IEEE, 2010.

[30] 任重, 杨灿军, 陈鹰. 轨迹规划中的 B 样条插值算法%B-spline interpolation algorithm in motion planning[J]. 机电工程， 2001， 018(005):38-39.

[31] Berglund T , Brodnik A , Jonsson H , et al. Planning Smooth and Obstacle-Avoiding B-Spline Paths for Autonomous Mining Vehicles[J]. IEEE Transactions on Automation Science & Engineering, 2010, 7(1):167-172.

[32] Elbanhawi M , Simic M , Jazar R N . Continuous Path Smoothing for Car-Like Robots Using B-Spline Curves[J]. Journal of Intelligent & Robotic Systems, 2015, 80(1 Supplement):23-56.

[33] 佚名. RViz: a toolkit for real domain data visualization[J]. Telecommunication Systems, 2015, 60(2):337-345.

[34] 郝宇赟. 自主汽车的智能纵向控制与实现研究[D]. 天津：天津大学, 2006.

[35] Frédéric Holzmann. Vehicle dynamics model[M]. Adaptive Cooperation between Driver and Assistant System. 2008.

[36] 郭应时, 蒋拯民, 白艳等. 无人驾驶汽车路径跟踪控制方法拟人程度研究[J]. 中国公路学报, 2018(8):189-196.

[37] 刘伟，肖旭辉，魏敬东. 无人驾驶汽车横向滑模控制仿真研究[J]. 北京汽车, 2017, 000 (004):31-34.

[38] Healey A J , Lienard D . Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles[J]. IEEE Journal of Oceanic Engineering, 1993, 18(3):P.327-339.

[39] 韩玉敏,李晗宇,潘可耕. 基于 RBF 神经网络的汽车稳定性滑模控制[J]. 黑龙江工程学院学报 (4):11-14.

[40] Von Raumer T , Dion J M . Applied nonlinear control of an induction motor using digital signal processing[J]. IEEE Transactions on Control Systems Technology, 1994, 2(4):P.327-335.

[41] Bellman, Richard Ernest, Dreyfus, Stuart E. Dynamic programming[M]. Dynamic Programming. Dover Publications, Incorporated, 2003.

[42] Tilman, Börgers, and, et al. Learning Through Reinforcement and Replicator Dynamics[J]. Journal of Economic Theory, 1997.

[43] J. Dalton and S. N. Balakrishnan, “A neighboring optimal adaptive critic for missile guidance,” Math. Comput. Modeling, vol. 23, no. 1, pp. 175–188, 1996.

[44] Wei Q , Liu D , Xu Y , et al. Neuro-optimal tracking control for a class of discrete-time nonlinear systems via generalized value iteration adaptive dynamic programming approach[J]. Soft Computing, 2016, 20(2):697-706.

[45] Kirk, Donald E. Optimal control theory[J]. American Mathematical Monthly, 2015, 83(4):261-288.

[46] Evanghelos, Zafiriou. Applied optimal control and estimation by Frank L. Lewis, prentice hall and texas instruments, englewood clgjs, nj, 1992,624 pp.[J]. Aiche Journal, 1994.

[47] Larson R E . A Survey of Dynamic Programming Computational Procedures[J]. IEEE Transactions on Automatic Control, 1968, 12(6):767-774.

[48] Prokhorov D V , Santiago R A , Ii D C W . Adaptive critic designs: A case study for neurocontrol[J]. Neural Networks, 1995, 8(9):1367-1372.

[49] Si J , Wang Y T . Online learning control by association and reinforcement[J]. IEEE Transactions on Neural Networks, 2001, 12(2):264-276.

[50] Wang F Y , Zhang H , Liu D . Adaptive Dynamic Programming: An Introduction[J]. Computational Intelligence Magazine, IEEE, 2009, 4(2):p.39-47.

[51] Enns R , Si J . Helicopter trimming and tracking control using direct neural dynamic programming[J]. IEEE Transactions on Neural Networks, 2003, 14(4):p.929-939.

[52] Camacho E F , Bordons C . Model Predictive Control[M]. Springer London, 2004.

装 订 线

谢 辞

时光荏苒，转眼即逝，在同济大学四年的本科生生涯即将结束，值此毕业设计结题之时，不禁感慨良多。心中既怀着对母校深深的眷恋，也对即将开启的另一段人生充满了好奇和期待。

首先，我要感谢我的导师陈耀副教授和薛亚歌导师，在毕业设计过程中，两位老师每周定时举行会议，为我毕业设计的开题，中期直到如今结题解答了诸多疑惑。两位导师深厚的学术造诣，严谨的态度，以及对学生十分耐心的解答给我的帮助是十分巨大的。他们鼓舞着我在毕业设计的课题上不断发掘，开拓自己的思维，在已经完成仿真平台开发的情况下继续研究控制算法，并给我在论文的撰写方面提出了许多十分宝贵的意见和建议。在此，我再次对老师们在毕业设计期间给我的帮助送上真挚的感谢和敬意。

其次，我要感谢杨晓莹同学对我在毕业设计过程的鼓励和支持，也感谢她和我一起讨论控制算法的研究问题；我还要感谢赖金涛博士在毕业设计撰写过程提供的指导；感谢班级刘剑锋同学在生活上提供的关心和帮助。

感谢父母和家人们对我的关心和支持，在我遇到困难的时候不断地鼓励我，激励我不断前行。最后，感谢百忙之中抽空审阅此文地专家、学者，感谢您们地辛勤劳动和付出！

装 订 线