華中科技大學

本科生毕业设计

一种基于 Unity3d 的趣味游戏程序设计

院　　系	计算机科学与技术
专业班级	计算机 1805
姓　　名	王钦卓
学　　号	U201814776
指导教师	黄志

2022 年 06 月 01 日

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包括任何其他个人或集体已经发表或撰写的成果作品。本人完全意识到本声明的法律后果由本人承担。

作者签名：　　年　　月　　日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保障、使用学位论文的规定，同意学校保留并向有关学位论文管理部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权省级优秀学士论文评选机构将本学位论文的全部或部分内容编入有关数据进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于 1、保密□，在　　年解密后适用本授权书

2、不保密□　。

（请在以上相应方框内打“√”）

作者签名：　　年　　月　　日

导师签名：　　年　　月　　日

摘 要

近年，共享经济大力发展，全国各地都配置了各种品牌的共享电动车。本趣味游戏程序通过寓教于乐的方式，让享受着共享电动车带来的福利的年轻人们，同时意识到电动车这一载具本身存在的不安定因素，让游玩到这一游戏的大学生乃至还不能骑车的未成年人，都提早认识到交通法律法规的存在，了解到骑电动车佩戴头盔的重要性。

本趣味游戏程序以华中科技大学的校园道路路线为基础，通过 Unity3d 编辑器功能进行场景模型搭建。整个系统分为非玩家角色逻辑模块、玩家电动车模块、交通法律法规检测与惩罚模块、系统界面模块，非玩家角色逻辑模块主要实现非玩家角色的寻路功能，玩家电动车模块主要实现电动车动力学的仿真效果，交通法律法规的检测与惩罚依靠 Unity3d 的碰撞检测实现，系统界面模块有着目的地导航、显示时间、道具金钱、交通罚分与警告等多项功能。

本趣味游戏程序在运行流畅，玩法多样的同时，利用完备的系统引导、贴近现实的交通逻辑、饶有趣味的道具系统和比赛系统，让玩家既能够顺利又愉悦地游玩，也能够深刻认知到各式各样的交通法律法规。

通过这样的虚拟校园电动车趣味驾驶游戏程序，其成果对于推广安全驾驶这一理念有着极为重要的意义。

关键词：虚拟校园；虚拟现实；驾驶演示；动力学；Unity3D

Abstract

In recent years, the sharing economy has grown vigorously, and various brands of shared electric vehicles have been deployed throughout the country. This fun game program allows young people who are enjoying the benefits brought by the shared electric vehicles to realize the unstable factors of the electric vehicle itself through an educational and fun way, so that college students who play this game and even minors, who are not yet able to ride, can be aware of the traffic laws and regulations in advance, and understand the importance of wearing when riding electric vehicles.

This fun game program is based on the campus road route of Huazhong University of Science and Technology, and the scene model is built through the function of the Unity3d editor. The whole system is divided into AI module, player electric vehicle module, detection and punishment of traffic laws and regulations module, system UI module. The AI module is mainly to achieve the non-player character of the pathfinding function, and the player electric vehicle module is mainly to achieve the simulation effect of electric vehicle dynamics. The detection and punishment of traffic laws and regulations module relies on the collision detection of Unity3d. The system UI module has many functions such as destination navigation, displaying things like time, prop money, traffic penalties and warnings.

The fun game program runs smoothly and has various gameplays. At the same time, it uses complete system guidance, realistic traffic logic, interesting props system and competition system , so that players can play smoothly and happily, and can also be profoundly aware of various traffic laws and regulations.

Through such a virtual campus electric vehicle fun driving game program, the results are of great significance for promoting the concept of safe driving.

Keywords: Virtual campus, Virtual reality, Driving scene demonstration, Dynamics, Unity3D

目 录

1 绪 论

本课题以 Unity3d 引擎为基础构建，需要了解 Unity3d 的历史与特性，了解本课题相关的知识与技术。

1.1 课题背景

1.1.1 研究背景和趋势

随着图形学理论与硬件的快速发展，工业界优质的图形图像软件、游戏图形引擎层出不穷。Bishop 等人[1]指出，过去的游戏程序员使用汇编语言来直接建立游戏程序与硬件之间的交互，他们强大的技巧与耐心往往能使产品有着精致的效果，甚至极具娱乐性、令人信服。但随着编程语言和硬件的同时发展，开发者们逐渐发现，游戏之间存在许多重复、可以进行解耦的部分，例如图形、音频以及其他更复杂的输入设备等等。所以无需针对每个游戏项目都为其开发一套对应的底层组件，这样做既不经济也无必要。于是，这些被分离并单独构建再用 GUI（即图形用户界面）进行包装的部分，就被称作“游戏引擎”。

游戏引擎的发展非常迅速。早在文献[1]就提到了场景管理、图形显示、剔除、动态碰撞处理、分层排序以及专门绘制对象这些基础设施。文献[2]是一本介绍编写游戏引擎的方法论的著作，其中就提到了利用 Shaders 制造特殊效果、场景管理树、内存管理、层次细节、面向对象的基础设施、原子化的方法等等从现代的角度来看依旧是常用常新的基础功能。Cozzi 等人[3]在 Eberly[2]的基础上，以如何绘制一个地球这个新奇的角度入手，开拓性地介绍了现代引擎的许多功能的实现。此外，近年的游戏对于多线程的利用也越来越频繁，这也要求了引擎对于多线程的可靠支持[4]。

Unity 诞生于丹麦，在 05 年苹果的全球开发者大会上对外开放，顺应移动端游戏的潮流而逐渐变得炙手可热。它在 2011 年基本完成了对全平台的支持，同时又通过提供部分免费功能的方式来扩大自己的用户量，最终在全世界范围内受到了开发者的广泛应用[5]。Haas[6]系统性地介绍了 Unity 的工作管线及工作流，以及其完备的多平台支持、详尽的文档和 Unity 云功能。近来，人工智能飞速发

展，与之的相关研究也可以部署在 Unity 平台上，Juliani[7]讨论了其可行性。

1.1.2 安全驾驶游戏的研究与实现的途径

首先需要对虚拟校园的场景进行搭建，这一系统的实现又可以分为地形、交通、建筑这三个层次；之后需要研究并实现电动车的车辆动力学系统，让电动车能够具有真实感地在道路上行驶；接着，要对道路上的交规逻辑进行设计，对路上的车辆及路人的寻路 AI（即非玩家角色寻路智能逻辑）进行设计；再次，可以适当增加一些功能来提高游戏的趣味性，例如道具、背包、时间限制、积分比赛、新手帮助、天气系统等等；最后，需要对游戏进行适当的优化，再将其打包成可执行文件。

1.1.3 面临的问题和挑战

（1）虚拟校园系统该如何设计与实现；

（2）如何对驾驶的电动车的车辆动力学结构进行研究与实现；

（3）交通法律法规逻辑该如何设计与实现；

（4）自动寻路 AI 逻辑该如何设计与实现；

（5）如何对游戏进行优化，使渲染过程中尽量减少 DrawCall，以提升游戏的性能；

（6）如何增加额外功能以使游戏具有较高的可玩性。

1.2 国内外研究现状

1.2.1 虚拟校园系统的设计与实现

这一问题并不是本课题研究的重点，但校园作为游戏的基础设施，的的确确是个必须实现的关键点之一。韦杨[8]对这一系统的实现分为三个阶段：首先是利用无人机来对校园的图形图像数据进行采集，确定各个建筑的大致尺寸，再通过实地拍照的方式进一步准备纹理素材的资料。其次，用 3DMAX 建模软件对校园进行建模，并对法线进行调整。最后，将模型导入到 Unity，并进行后续的工作。

其中，在第一阶段里，利用地理信息系统（GIS）对场景进行事先分层是非常有必要的。这个系统可以将校园分为地形层、交通层、建筑层、植被层和公共

设施层，而本课题主要还是聚焦于前两层以及第三层的粗模的建模上。

一个地势较为平坦的学校，其地形层的建模可以简单利用 Unity 自带的地形编辑器来实现，也可以通过无人机拍摄到的地形照片，利用 PhotoShop 软件转换成灰度图并导入到 3DMAX 来直接生成三维地形；道路层的建模可以直接在 3D MAX 中，通过给地形层贴上纹理贴图的方式来完成；对于建筑层，简单建筑可以直接用不同形状的几何体的组合来替代。与韦杨[8]不同的是，冯新玲[9]对于建筑的建模，选择了使用 AutoCAD 软件进行图纸设计，再直接导入到 3DMAX 中。

1.2.2 国内交通安全教育系统的研究与实现

王梅亮等人[10]将这一系统分成五个模块，来实现交通安全教育的功能，它们分别是包含了训练者的个人信息、当前车流量的数据模块，利用建模软件 MAYA 生成的场景模块，声音仿真模块，交通规则模块，以及最重要的交通安全技能训练模块，也就是它的模拟行走模块。

这一文献对于模块的划分具有一定的参考意义，但由于它的模拟主体是路人，因此其交通规则的设计较为朴素，交通安全技能训练模块的车流量设置也略显简单。

文献[11]同样是以人为主体进行模拟，但聚焦于儿童，通过大量的数据分析儿童的行为模式，提出了各种各样的儿童靠近车辆的时候可能做出的举动，并为此设置了各种与之相对应的代码逻辑。这样的设计极具启发性，可以很好地实现寓教于乐的目的。

以上两个文献的模拟主体虽然都是人，但它们对于交通环境的设计、周围车流量的设置，与将车作为主体其实都是相似的。

1.2.3 国外交通仿真系统的研究与实现

国外对于交通仿真系统的研究与实现，主要是为开发自动驾驶汽车而服务。它们往往将引擎与人工智能相结合，通过代理的方式来测试事故以及车祸的发生频率。虽然本课题不涉及深度学习领域，但下述文献在微观，也就是车与车之间的一些函数方法上，对于本课题都具有很好的启发作用。它们宏观上得到的结论也具有相当的意义。

车辆之间的微观跟随模型就是一个很好的切入点。这个模型描述的是两辆车之间超车的一个情况，大多数的研究结果都是基于理论模型、统计数据以及经典物理学。

Han 等人[12]考虑到相邻车辆之间的相对速度，对车辆之间的跟随模型进行修正，建立了耦合映射跟车模型，很好地模拟了交通堵塞的现象，以及为了防止交通堵塞需要做出的调整。

Chen 等人[13]基于数据得出车辆的经验轨迹，并以此为判据来模拟车辆之间的相互运动，又将驾驶员的跟车行为纳入到考察之中，实现了减少交通事故发生概率的效果。

Farhi [14]设计了一个最小加速度模型，根据车流量的不均匀性，以及驾驶期望对于动态和静态交通状况的影响，达成了无关卡状态下将车流量进行分散的效果。

Yu 等人[15]通过增加全速差（FVD）跟驰模型的局部稳定性、渐近稳定性以及李雅普诺夫稳定性，进一步提升了该模型的动态性能。Lu 等人[16]对 FVD 模型进行了扩展，将一般行驶与加速过程的两种不同的延迟速度分开讨论，并考虑到时间延迟对车流稳定性分析的影响，最终实现了抑制交通阻塞的效果。

Li 等人[17]对上述跟驰模型进行改进，很好地考虑了车距对于车速的影响，增加了基于车距对加速度进行调整的功能，从而缩短了各个车辆在不同情况下的启动时间。

而从宏观的角度来看，这些交通仿真系统也极具意义：

Chao 等人[18]提到可以利用车联网的监控技术，及时地感知到驾驶员的情绪并进行反馈，将这些情感数据集成到对交通的模拟上[19]。交通仿真系统的成果可以广泛应用于无人驾驶技术、城市交通决策、道路拥堵管理等等方面，但实际上还存在诸多问题亟待解决，例如将道路上的天气状况纳入考虑[20]。

Liu 等人[21]讨论了计算机图形学中关于交通模拟和动画技术相关的话题，近十年学术界利用逐渐增大的来自不同地区的交通数据库来帮助开发和测试自动驾驶算法，建立了一个兼具宏观、中观和微观的交通流模型。这些数据驱动的运动规划技术将帮助我们创建各种各样的决策模拟器，想必未来将会有越来越多的

令人惊喜与兴奋的交通应用程序。

1.2.4 机动车驾驶模拟系统的研究与实现

汽车、电动车的物理系统，如果想要达到近乎现实的加减速效果及转向效果，就必须对其车辆动力学进行研究与实现。

文献[22]与本课题类似，以摩托车作为主体进行模拟驾驶，并强调了安全帽对于驾驶员的重要性。其中提到了基本的障碍物设计、碰撞检测、镜头控制等内容，对于摩托车的车辆动力学结构较少提及。姜峰等人[23]则指导性地提出了虚拟驾驶系统的四自由度方法论，简单用刚体来模拟汽车的前进与后退，运动中的车轮绕自己的 X 轴旋转，一旦转向，则两个前轮需要绕自己的 Z 轴发生转动。最后通过运动控制卡，让玩家能够在现实中操纵实体方向盘并将运动数据输入到设备之中。

曾林森[24]利用 EasyRoad 进行了高效的道路编辑，通过 3DMAX 对车辆进行建模，利用 iTween.path 来指导其他车辆以及路人的行进路线，这些都对本课题有一定的启发性。值得一提的是，它根据车辆动力学实现了车辆的驱动仿真、悬架动力仿真等等效果。

张茜[25]系统而全面地介绍了一辆汽车的各个组件该如何利用面向对象的方式进行构建，将整个汽车系统分作了行驶、转向、制动、传动、电气、车身等等系统，每一个系统都有明确的流程图以及结构图。同时还将汽车的起步、加减速、倒车等场景都一一作了展示，完善地介绍了一个虚拟汽车系统的实现方式。

1.2.5 寻路 AI 的研究与实现

寻路 AI 对于每一个游戏都是必不可少的，这其中主要分为两种方式：无视碰撞或不会发生碰撞的固定路线，以及对碰撞会产生交互的随机寻路 AI。王志岗[26]介绍了 Unity 的寻路系统，原理就是通过事先进行烘焙，将地形信息存储到文件中，从而让 AI 角色能够感知到可以行走的区域。

史宝明等人[27]提到，普通的随机寻路算法过程中，在转身的过程中往往会出现动作怪异、不连贯等现象，而解决这一问题的方法就是利用欧拉角与四元数的转换，来及时完成角色角度的更新。这种方法大幅度提升了随机寻路算法中角色

的移动的真实性。

Mas'udi 等人[28]提出了通过决策树的方式来规划 NPC 的行动轨迹，其中寻路的部分将交由航点系统以及光线投射来完成。通过这种方式进行决策的 NPC 速度要比单纯由人工智能决策的 NPC 要更快，但是这种方法会略微降低游戏的帧率。

1.3 课题研究的意义、内容和目标

1.3.1 课题研究的意义

近年来，公安部交管局大力部署“一盔一带”行动，依法处罚道路上没有正确佩戴头盔的摩托车、电动车驾驶员，以推进大众对于戴头盔这一意识的认知。更有电动车驾驶员，甚至意识不到自己骑的车属于机动车，应该行驶在机动车道，遵守机动车的交通规则，因而造成的事故、车祸数不胜数。

作为一名大学生，我们设计这一课题就是为了让享受着共享电动车带来的福利的年轻人们，同时意识到电动车这一载具本身存在的不安定因素。通过寓教于乐的方式，让游玩到这一课题的大学生乃至还不能骑车的未成年人，都提早认识到交规的存在，意识到头盔的重要性，明白“道路千万条，安全第一条”这一道理。

1.3.2 课题研究的内容

（1）搭建一个仿华中科技大学校园的虚拟环境；

（2）研究并调试出一个能够在正确的物理规则下运动的电动车；

（3）查找资料，编写代码，设计一个符合现实的交通规则的游戏环境；

（4）添加一些额外功能，例如道具、背包、时间限制、新手帮助等等；

（5）进行游戏优化，将游戏打包。

1.3.3 课题研究的目标

（1）仿华中科技大学校园的虚拟环境能够符合游戏需求；

（2）虚拟电动车能够在正确的物理规则下运动；

（3）游戏过程符合现实的交通规则同时，能够起到寓教于乐的作用；

（4）优化过程中尽量减少 DrawCall，提升游戏性能。

（5）增加额外功能让游戏具有较高的可玩性。

1.4 论文结构

本文的主要内容如下：

第一章首先介绍 Unity3d 软件的研究背景与当下的发展趋势，简要地分析制作一个趣味驾驶游戏兼安全教育游戏需要做哪些准备，有哪些实现途径，又会面临哪些问题与挑战。之后，通过阅读国内外的文献，了解到世界各地在这一课题及其相关课题上所做的研究与技术，例如虚拟校园系统的设计与实现、交通安全教育系统的研究与实现、交通仿真系统的研究与实现、机动车驾驶模拟系统的研究与实现、车辆和路人的寻路 AI 的研究与实现等等。

第二章首先分析系统的功能性需求与非功能性需求，然后分析将会用到的开发工具。同时也要理解这一课题所需的理论及其关键技术，并在第二章里简要地概述这些概念以及它们在本课题里起到了什么作用。之后要设计出一个基本方案，并以这一方案为主干来扩充其他需求的功能。

第三章开始对整个电动车安全驾驶趣味游戏进行具体的系统设计。首先是系统总体设计，要根据第二章的开发工具分析，选择合适的硬件平台、软件平台、开发环境，并对功能模块进行简要划分。之后就是每一个功能模块的具体设计，需要简明扼要地将每一个功能的系统结构都用文字或是结构图来展示出来。

第四章开始对整个电动车安全驾驶趣味游戏进行系统的具体实现。首先是玩家主体电动车和 AI 车辆的动力学系统实现。然后需要对游戏的 UI 首先进行设计。游戏里还会需要一些 NPC（即非玩家角色），那么就需要完成他们的寻路功能。最后再按照现实中的法律法规对游戏逻辑进行设计，完成相应的检测与相应惩罚的实现。

第五章需要对游戏进行测试，这个测试在 Unity 编辑器内即可完成，同时还要有功能测试与性能测试，这也需要用到一些相应的软件。

第六章将对整个课题进行总结，归纳这次课题设计的优点与不足之处，提出一些可能的改进方案，并做一些展望。

2 游戏方案论证

为了更好地完成本课题，首先需要对游戏需求进行分析，对于设计方案进行初步地论证与构建。

2.1 系统需求分析

本游戏系统的需求分为功能性需求与非功能性需求这两个方面。前者需要考察一个基于现实交规的场景里，哪些功能是不可或缺的，哪些功能又可以提高玩家的游戏体验；后者则需要检查游戏对于软件和硬件的友好程度，判断游戏的优化程度。

2.1.1 功能性需求分析

从游戏设计流程上看，首先需要搭建一个实用的校园环境，以供游戏中各种逻辑正常运作的需求；需要研究并实现一个符合物理的车辆动力学系统，且车辆既有电动车也有汽车；需要完成游戏的界面 UI，它可以与游戏的各种逻辑进行对接，帮助玩家了解游戏、学习法律法规知识并顺利地通关；需要实现 AI 路人和 AI 车辆的自动寻路，方便之后的交通逻辑实现；最后需要按照现实里的交通法律法规寻找符合电动车的条目，并设计具有本游戏特色的需求。

（1） 校园环境搭建需求：

① 需求说明：从出发点到目的地，横跨校园的东西与南北，但由于校园的地势较为平坦，且道路朝向较为规整，因此无需特地对地势高低进行分析设计。

② 需求前提：道路贴图、基础建筑模型、限速路标等素材。

③ 需求流程：地图起点为华中科技大学的韵苑学生食堂，终点为华中科技大学的西十二教学楼附近停车场。

④ 需求结果：搭建出可以供电动车、汽车驾驶的道路，以及供校内行人行走的人行道。

（2） 符合物理的车辆动力学系统需求：

① 需求说明：需要四轮的 AI 汽车在道路上行驶，两轮的玩家主体电动

车进行操控，AI 汽车需要最低程度的碰撞检测，玩家的电动车则需要较为复杂的碰撞检测，且电动车在转弯过程中的倾侧效果也需要实现。

② 需求前提：已搭建校园环境供车辆行驶。

③ 需求流程：

A 用户启动电动车，可以通过 WASD 控制电动车的加减速与转弯，通过空格键可以使电动车刹车；

B 用户在骑行的过程中，能够感受到一定的倾侧效果，这种效果在转弯时尤为明显；

C 用户在骑行的过程中，会遇到一些行驶的 AI 汽车，并且需要避免与它们相撞。

④ 需求结果：电动车、AI 汽车的运行情况符合之后的交通规则逻辑的实现需求。

（3） UI 界面需求：

① 需求说明：需要设计一个 UI 界面，与游戏的各种逻辑进行对接，同时帮助玩家了解游戏的操作方法，对游戏的目的地进行导航，最后还要控制游戏的开始与成功与失败界面。

② 需求前提：无。

③ 需求流程：

A 进入游戏，按“p”可以暂停查看游戏说明、操作指南，也可以选择继续游戏、切换游戏模式；

B 左上角显示当前时间，如果时间到达八点，还没有达到目的地，则游戏失败；

C 屏幕左边显示当前的时速、违章分数以及各个严重程度级别的违规次数；

D 屏幕上方有一条水平线，上面有“东 南 西 北”字样以及一个五角星符号，显示出当前的运动方向和目标位置；

E 屏幕右下角有三个半透明框，显示当前背包里的道具；

F 游戏失败时弹出界面，显示重新开始、操作说明、游戏结束等界面按钮；

G 到达目的地后，弹出游戏成功的界面。

④ 需求结果：UI 界面能够很好地帮助玩家了解游戏的玩法，并顺利到达目的地。

（4） AI 汽车以及 AI 行人的寻路需求：

① 需求说明：在各项交通法律法规当中，有很多法规与“避让行人”、“会车”相关，这需要在场景里设计出有移动逻辑的 AI 汽车和 AI 行人。

② 需求前提：已搭建校园环境供 AI 移动。

③ 需求流程：AI 汽车的移动需要在道路上形成闭环，AI 路人可能会横穿马路，在人行横道上有一定概率会看红绿灯。

④ 需求结果：AI 汽车与 AI 行人能符合现实逻辑地在道路上移动。

（5） 交通规则的检测与惩罚逻辑需求：

① 需求说明：电动车在当前的交通法律法规中并没有太多的限制，但需要实现最基本的不超速驾驶、遵守交通信号灯、避让行人等等交通规则。可以设立两个检测系统：初始 12 分的交通违章可扣分数，初始 1500 元的钱包余额，并根据违反的交通规则条目进行扣分或者罚款，并进行警告。

② 需求前提：已搭建好校园环境，道路上有限速标志、交通信号灯，有行驶车辆与来往路人。

③ 需求流程：

A 与车辆或者行人发生碰撞，游戏直接结束；

B 行驶速度超过规定的时速百分之五十以上的，扣 6 分，扣 300 元；

C 路口行驶时没有按交通信号灯的指示来行驶，扣 6 分，扣 300 元；

D 行驶速度超过规定的时速的百分之二十但没有达到百分之五十的，扣 3 分，扣 100 元；

E 没有在规定车道上行驶（例如行驶到了人行道上），扣 3 分，扣 1

00 元；

F 逆行，扣 50 元；

G 经过人行横道时，没有避让行人，扣 50 元；

H 不戴头盔，扣 1 分，扣 50 元；

I 行驶速度超过规定的时速的百分之十但没有达到百分之二十的，扣 50 元；

J 行驶速度超过规定的时速但没有达到百分之十的，口头警告。

④ 需求结果：游戏中的交规功能与现实接轨，同时能够起到良好的寓教于乐的作用。

（6） 难易度选择和趣味模式需求：

① 需求说明：考虑到玩家的游戏习惯不同，为了让大多数人能快速上手游戏，需要进行不同模式的设计。

② 需求前提：游戏逻辑基本完成，只需设计开关来选择逻辑的启用与否。

③ 需求流程：

A 点击暂停之后，可以通过点击切换游戏模式，来选择其他游戏模式；

B 选择“自由模式”，关闭所有交规检测逻辑，玩家可以自由探索华科地图；

C 选择“简单模式”，关闭逆行、信号灯逻辑；

D 选择“普通模式”，启用所有交通检测逻辑；

E 选择“趣味模式”，关闭与 NPC 车辆发生碰撞的交规检测逻辑，并增加几辆 NPC 电动车，玩家可以与这些 NPC 进行趣味比赛，右边会显示一个 UI，按照各个电动车离目的地的距离进行排序。

④ 需求结果：玩家能够充分享受趣味游戏带来的乐趣，同时也能学习到交通法律法规知识。

2.1.2 非功能性需求分析

非功能性需求相比功能性需求，更强调游戏的软硬件适应性、性能问题以及游戏中的 BUG 问题。这与游戏测试与分析关联，需要尝试在各种极限情况下的

游戏逻辑是否正确，以及各种极限情况下的游戏帧率，在多个硬件上对游戏进行测试，让多位玩家进行测试等等。

在大型游戏开发中，这种非功能性的需求往往才是最困难的部分，本课题除了以上功能性需求的实现之外，也要认真考察游戏里存在哪些非功能性需求，并试图解决。

2.2 系统可行性分析

系统可行性可以分为技术可行性、经济可行性与社会可行性三个方面，针对系统可行性的分析是课题必不可少的步骤，以免于设置一些空中楼阁、不可完成的目标，耗费大量的人力物力。

2.2.1 技术可行性分析

本课题基于 Unity3d 来完成项目的实现。Unity 作为游戏引擎，版本日新月异，从 2007 年的 2.0 版本，到 2015 年的 5.0 版本，再到现在以年份为版本号，每个稳定版本之间都会新增相当多的功能。

如今的 Unity 已经具备图形、物理系统、多玩家联网交互、音视频、动画、UI、导航和寻路等等多种多样的模块，这些模块足以支撑本课题的搭建和功能实现。

同时，Unity 开放的资源商店，提供一些免费的模型、贴图供游戏制作者使用，这些模型和贴图也可以满足本课题的环境搭建需求。

2.2.2 经济可行性分析

在 Unity 最初的十年，运营团队主要将其作为付费软件直接售卖。从 2016 年开始，运营团队将其运行模式更改为订阅模式，如图 2-1 所示。其中，免费版用户就已经能享受到 Unity 的很多功能。

本游戏中还可能会涉及到一些模型和贴图的使用，除了利用 Unity 的资源商店、互联网上已有的一些免费 3d 模型、贴图网站以外，如果有一些能极大程度增加游戏模拟感、真实性的付费模型、贴图，本课题也会尝试购买并使用，以增强游戏的游玩体验。

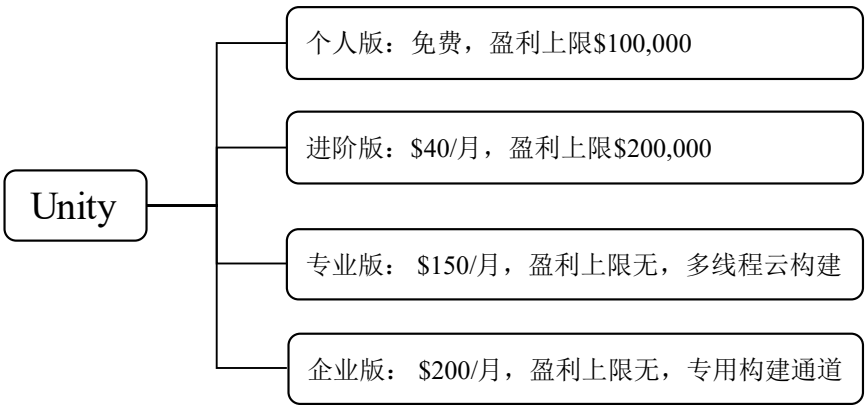

图 2-1 Unity 付费方式

2.2.3 社会可行性分析

本课题设计的趣味游戏同时也是一款电动车安全驾驶模拟游戏，将严格地按照现实生活里的法律法规来设计游戏逻辑，并在游戏里向玩家科普这些法律法规，通过虚拟罚分罚款的模式让玩家知悉遵守交通规则的重要性。

同时，电动车这一载具由于防护措施极少，因此头盔对于每一个驾驶员来说都必不可少。但现实生活里，骑电动车戴头盔却没有作为一种习惯被大众所认知，而在本课题中，如果玩家游玩游戏时没有戴上虚拟头盔，游戏画面会持续发出警告，这有利于玩家在潜意识里形成骑电动车戴头盔的好习惯。

因此，本课题在社会层面有许多好的影响，有着非常不错的社会可行性与实践性。

2.3 关键技术分析

要使用 Unity3d 开发一款游戏，最基本的要求就是明白游戏里的一帧到底经过了哪些事件，如图 2-2 所示。

首先是初始化阶段，引擎会立刻启用已经设定好的预制体和 OnEnable 的游戏实体，进行场景的初次加载。然后，在第一帧之前，需要在 Start()函数里进行游戏各种数据的初始化。

然后就正式进入帧更新阶段：

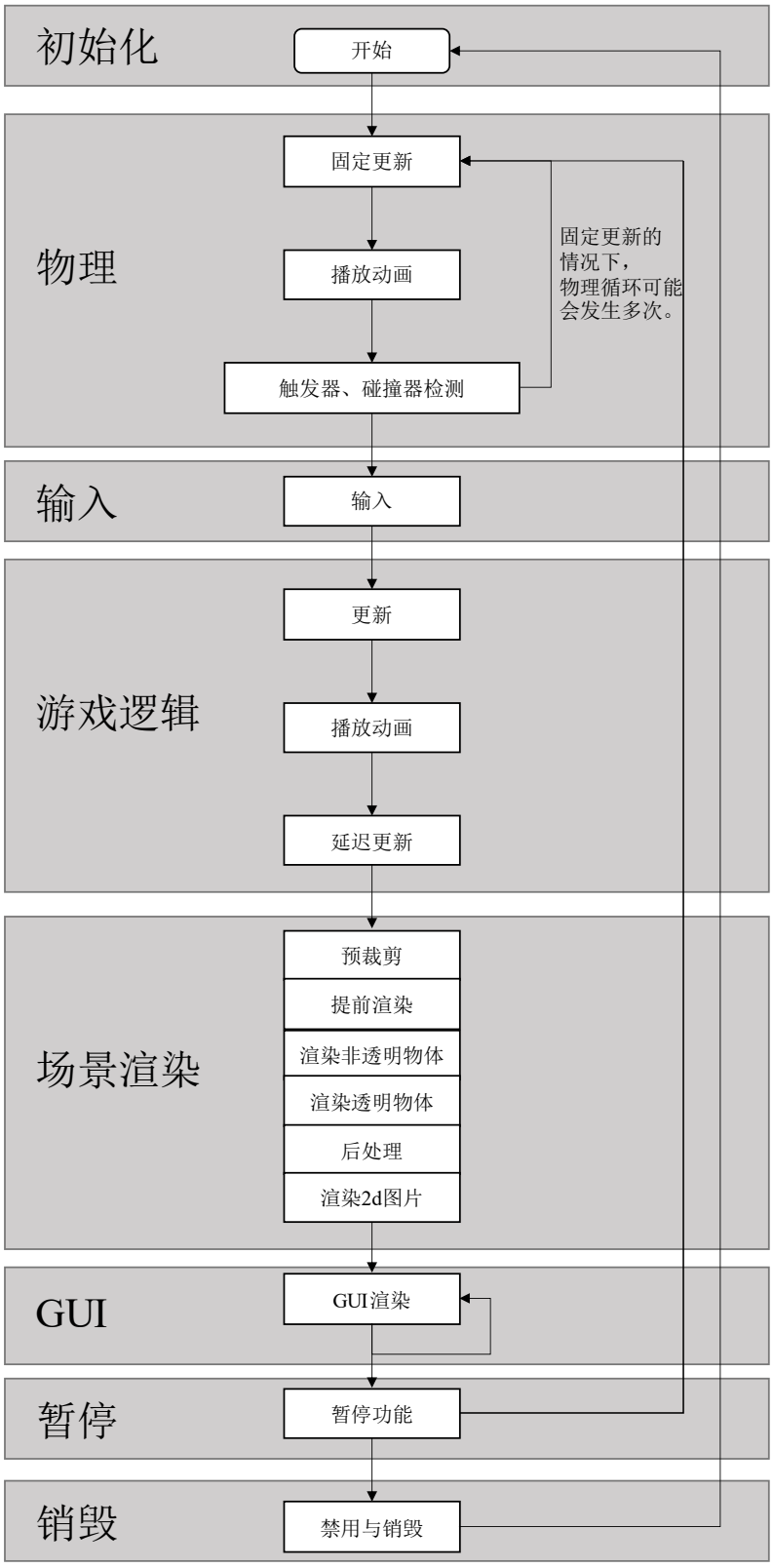

图 2-2 Unity3d 帧事件循环

如果游戏启用了 FixedUpdate()，则在物理系统更新阶段，根据硬件性能的不

同，游戏可能在一帧内反复调用一个物理循环，也可能直接跳过这一循环。与之相对的还有普通的 Update()，以及稍微滞后的 LateUpdate()。最后一个更新的主要特点时，在它更新的时候所有的计算都已经完成，可直接使用一些实时的数据进行更新。

在场景渲染过程中，游戏往往要通过应用阶段、几何阶段以及光栅化阶段，再加之 Unity 引擎的后处理阶段。这一阶段也可以对渲染管线进行自定义的设置。在应用阶段里，游戏需要准备基本的场景数据，对渲染的合批方式进行设置，并将待渲染图元输出到显存。之后两个阶段游戏可以通过着色器来对渲染时的一些效果进行处理，Unity 自带的管线也会完成一些基本的渲染处理。后处理阶段可以利用屏幕空间的数据进行一些操作。

每一帧之间，可以选择游戏是否暂停；退出场景时，将对场景里的所有对象都调用禁用或销毁函数。

2.4 基本方案制定

为了实现趣味游戏系统的开发，首先要搭建相应的开发环境，设计一个正确的开发流程。为了更好地进行功能实现，需要理解游戏引擎的运行流程，并依靠这个运行流程进行系统里各种游戏对象的分层设计。

在游戏对象都建立完毕之后，需要编写各项功能的代码文件并挂载在游戏对象上。代码需要有合理的模块划分，增强各个类的解耦性以便于面向游戏对象实现功能。

在功能基本实现之后，需要在不同的硬件环境里对游戏进行性能测试，需要在各种极限条件下测试游戏功能实现的可靠性。根据测试结果，还需要对功能进行进一步修改，对游戏性能进行进一步优化。

2.5 本章小结

本章分析了系统的功能性需求与非功能性需求，并列举了系统在功能方面可能会需要的接口及方法，在非功能方面可能需要做的优化等等，分析了游戏系统的经济、技术、社会可行性，通过 Unity 一帧内的事件循环，展示了基于 Unity 运行的游戏的本质。最后，制定了本课题的基本设计实现方案。

3 游戏系统设计

本章开始，将依据之前的基本方案，进行游戏系统的总体设计以及部分模块的详细设计。

3.1 系统总体设计

3.1.1 系统开发环境

本课题使用如表 3-1 的硬件环境进行软件开发：

表 3-1 硬件环境配置

硬件环境	配置
处理器	Intel® Core™ i5-8400 CPU
显示适配器	NVIDIA GeForce GTX 1050 Ti

本课题使用如表 3-2 的软件环境进行软件开发：

表 3-2 软件环境配置

软件环境	配置
操作系统	Windows 10 19043.1645
开发平台	Microsoft .Net Framework 版本 4.8.04084
Unity3d 编辑器	Unity 2020.3.32f1c1
代码编辑器	Visual Studio 2019 版本 16.5.4

同时，设计的游戏系统将在 Windows 平台上运行，因此系统的测试环境也直接利用 Unity 自身的游戏测试功能来进行。

3.1.2 系统开发流程设计

如图 3-1 所示，本课题首先要进行数据与素材的收集，根据贴图的内容，可能还需要将部分贴图通过 Photoshop 进行处理以满足课题的需求，之后将这些数据与素材导入到 Unity3d 中并制作虚拟游戏对象。

然后，需要为这些虚拟游戏对象编写具体的实现代码，并挂载在这些对象上面，并在游戏界面上设计 UI，完成与玩家的交互功能。

最后，将整个游戏打包并生成 exe 文件，供玩家游玩。

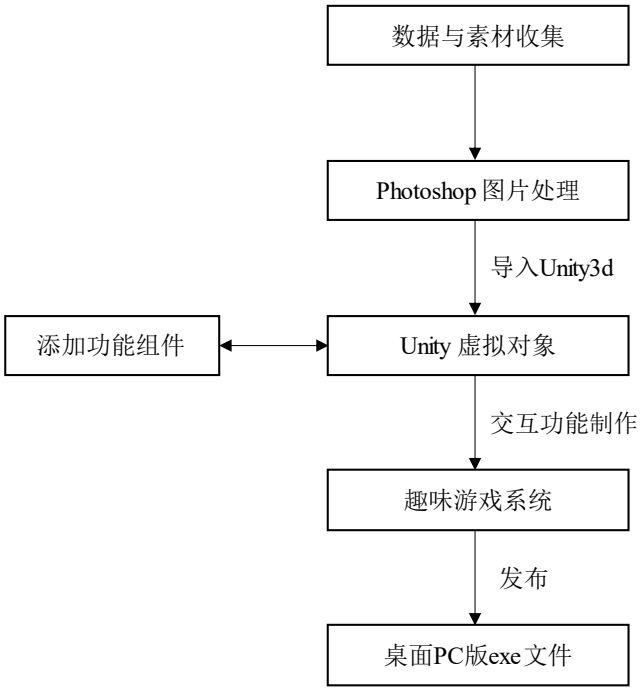

图 3-1 系统开发图

3.1.3 系统运行流程设计

对于整个游戏系统来说，整个游戏的运行过程是帧与帧的叠加，而每一帧则对应一次次事件循环。

如图 3-2 所示，游戏启动之后，首先要进行初始化，对各项游戏数据进行加载，这一过程根据硬件的不同可能会花费 1~5 秒不等。

之后，玩家会看到道路上的电动车，以及屏幕四周的各类 UI，在通过 UI 的指示，对游戏有大致了解之后，玩家便可以通过键盘、鼠标等输入设备来驱使电动车移动。

电动车移动的过程中，开始执行游戏的各项逻辑，例如车辆移动逻辑、交通检测逻辑、交通惩罚逻辑、碰撞检测、UI 实时数据更新、AI 寻路等等。更新的画面从内存输出到显存，在刷新到屏幕上，然后进入下一个帧循环，整个游戏的生命周期持续着这样的过程。

玩家想要结束游戏的时候，可以通过暂停，选择结束游戏，也可以直接关闭游戏窗口。

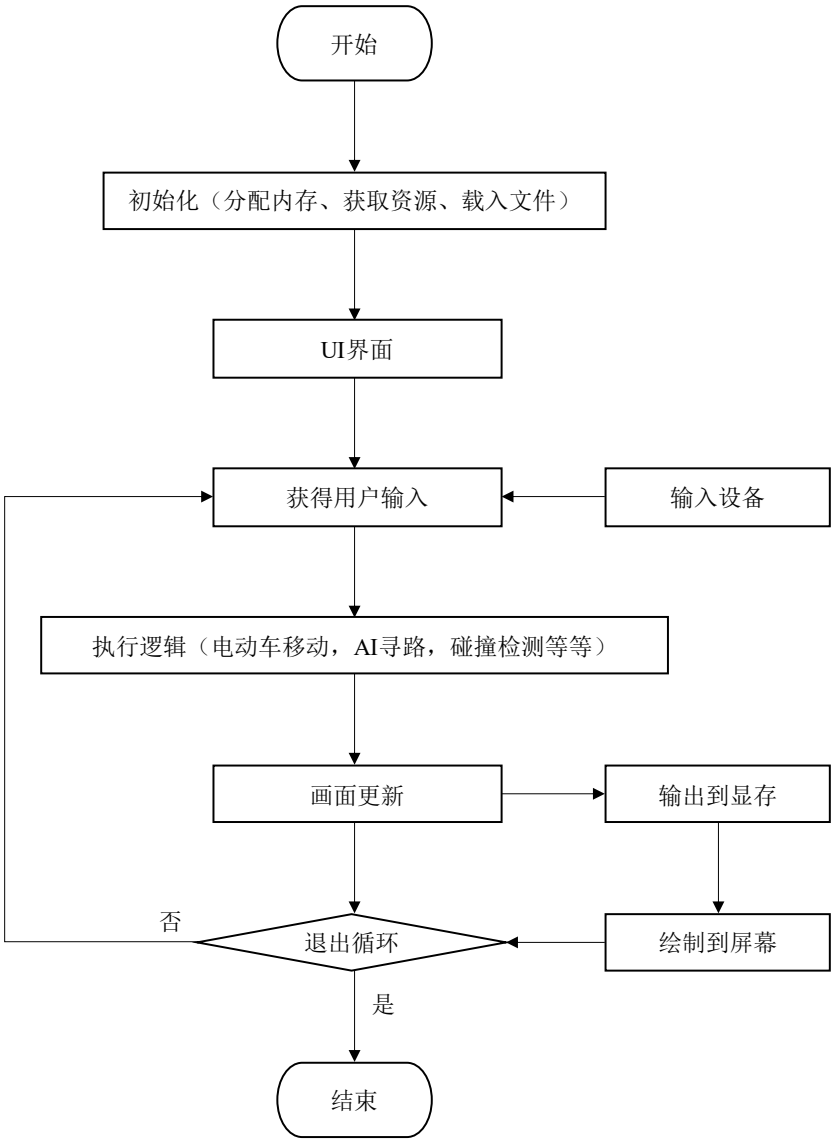

图 3-2 系统运行流程图

3.1.4 系统分层设计

在使用 Unity 编辑器的时候，用到最多的就是 Hierarchy 界面和 Assets 界面。前者存放当前场景的游戏对象，后者则存放这些游戏对象的对应贴图、模型、代码、动画等等资源。

本课题需要合理设计整个软件开发的层次结构，遵循面向对象的原则，以游戏对象的类型为标准进行各类对象模块的划分，使整个 Hierarchy 界面井井有条，方便后续的设计，如图 3-3 所示。

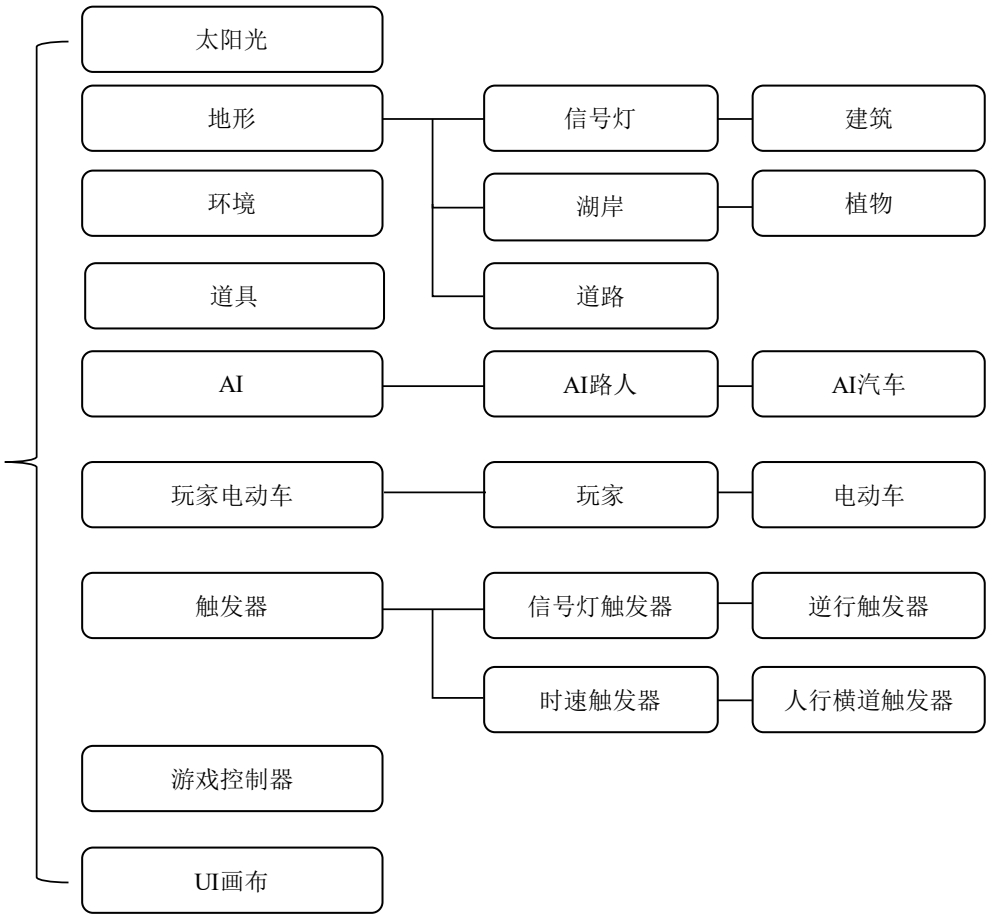

图 3-3 游戏对象层次结构图

环境层次包括学校的建筑、道路、韵苑食堂附近的湖水、植物以及交通信号灯信号牌等等。

道具层次包括加钱道具、加分道具、重置罚分道具等等。他们将零散分布在校园的道路上。

AI 层次有 AI 路人和 AI 汽车，玩家电动车层次包括一个玩家实体和一个电动车实体。

道路上配置了各种形状的立方体，它们没有具体的碰撞实体，而是作为各种游戏逻辑检测的触发器使用，例如信号灯触发器、逆行触发器、时速触发器、人行横道触发器等等。

游戏控制器与 UI 画布作为游戏设计的重要组成部分，将在功能模块设计部分进行具体描述。

3.1.5 系统功能模块划分

本游戏系统可以分为下列模块，如图 3-4 所示。AI 汽车与 AI 行人有一个移动模块与一个寻路模块，他们的移动逻辑也要符合交通规则；玩家电动车挂载移动模块的同时，还要挂载一些交通检测模块，在进行检测的同时将实时数据反馈给交通法规检测模块，后者再继续反馈给 UI 模块，以对玩家进行交通罚分或罚款。道具模块与 UI 模块的背包部分相关联。

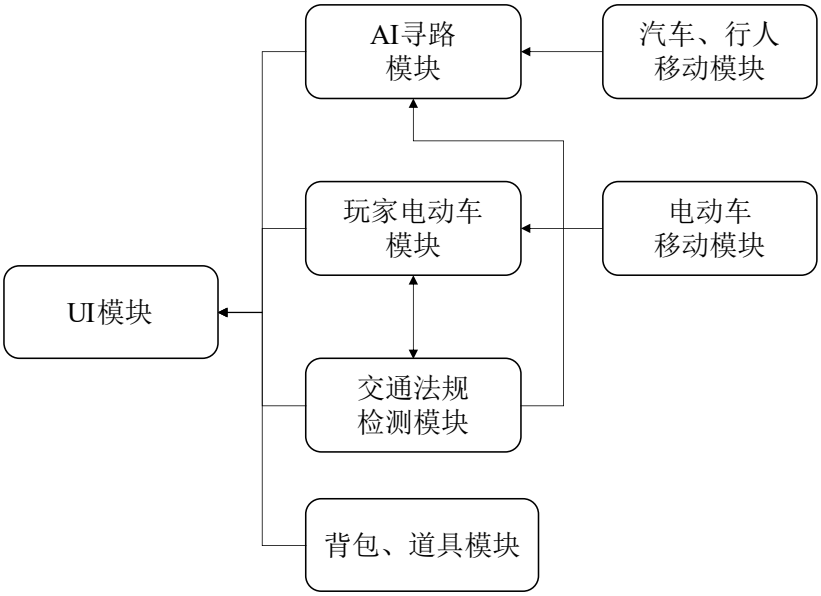

图 3-4 系统功能模块划分图

3.2 功能模块设计

3.2.1 基于 Unity3d 的车辆动力学系统结构设计

车辆一般由动力学系统、灯光系统、音效系统、粒子系统这些部分组成，在本趣味游戏程序里，为完成最基本的交通规则逻辑，首先要设计的就是其动力学系统。

由于本趣味游戏程序中，不仅需要作为玩家主体的电动车在道路上行驶，还需要作为游戏逻辑处理必须要素的 AI 汽车，所以两者的动力学系统都需要进行研究和实现。

（1）基于 Unity3d 的汽车动力学系统结构

如图 3-5 所示，在确保 AI 汽车已经配备了刚体组件的情况下，为减少计算

压力，可以简单使用一个 Box Collider 作为它的车身碰撞器，并保证碰撞器能够包裹住整个车身；四个车轮则使用 Unity3d 系统自带的 Wheel Collider。

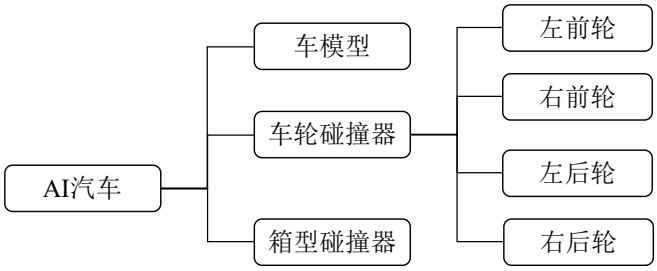

图 3-5 汽车动力学系统结构图

（2）基于 Unity3d 的电动车动力学系统结构

与 AI 汽车的简单模拟不同，电动车作为二轮载具，且对驾驶员的防护装置较少，所以需要配置多个碰撞器以进行更全面的模拟。

如图 3-6 所示：

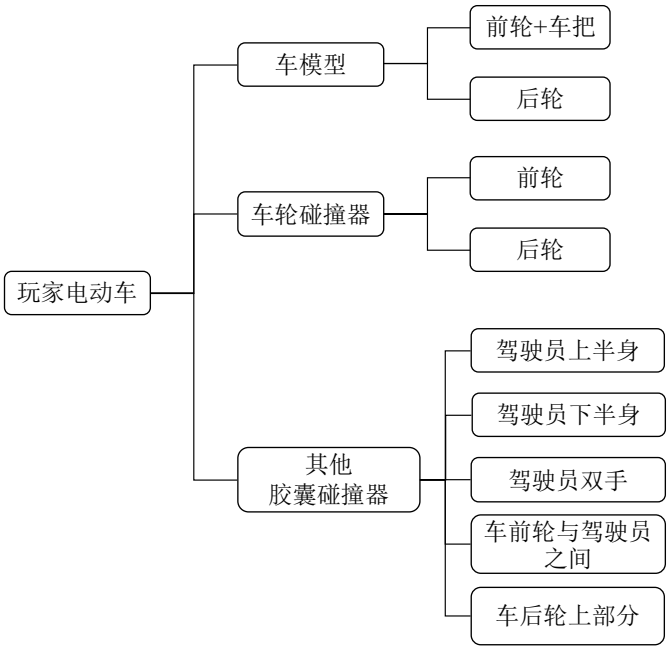

图 3-6 电动车动力学系统结构图

游戏用五个 Capsule Collider 来分别模拟驾驶员的上半身、驾驶员的下半身、驾驶员的双手及车把、车前轮与车把之间一部分、车后轮的上面一部分。

两个轮胎作为动力学模拟的重点，游戏不采用系统自带的 Wheel Collider，

而是使用开源的 WSP Collider。这样可以更方便地获取更多有用参数，来更好地模拟电动车的驾驶行为。

为了模拟驾驶电动车时车把的左右旋转，游戏需要将车把模型也归为车前轮部分的子组件。

为了模拟驾驶电动车时转弯时的左右倾侧，游戏需要创建一个 Stabilizers 组件，通过光线投射以及电动车自身驾驶时的相关数据来调整模型的倾侧幅度。具体实现将在后文进行更详细地描述。

3.2.2 基于 Unity3d 的 UI 界面设计

UI 界面作为玩家与游戏交互时的提示窗口，需要展示很多游戏中的实时数据供玩家作判断，同时还要在合适的时机向玩家发出警告，展示玩家的违规扣分、钱包余额，从而达到帮助玩家更深层次地理解交通规则，遵守交通规则，实现寓教于乐的目的。

UI 界面的设计主要依靠 Unity3d 自身的 UI 编辑功能，其大致分布如图 3-7 所示：

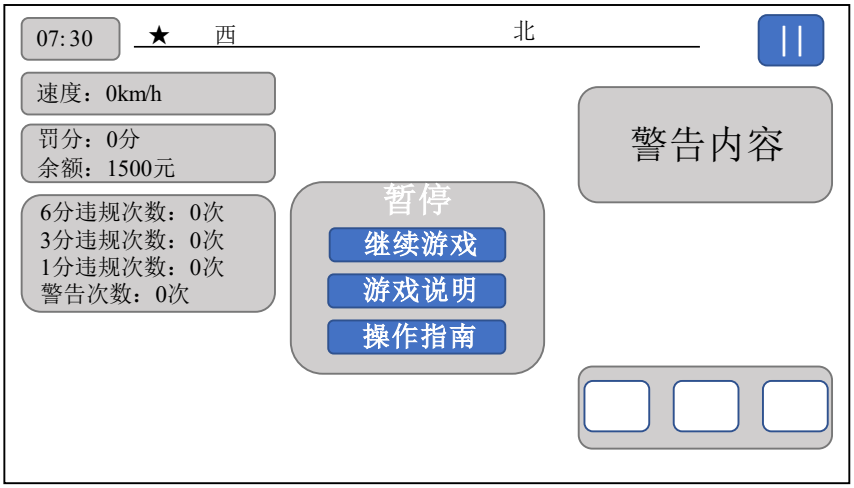

图 3-7 UI 界面设计图

左上角是当前的时间，下面是电动车当前的速度，再下面是玩家当前的交通罚分情况、钱包余额。再下面是违规行为计数。

屏幕上方有一个水平导航线，玩家转动视角的时候，上面会有东、西、南、北以及代表目的地的五角星。点击右上角的暂停之后，会显示出暂停界面，可以

选择继续游戏、游戏说明和操作指南。

发生交通违规行为之后，画面的正上方会显示出警告内容（即图 3-7 中右上角的警告框）。右下角有一个小型背包功能，可以将游戏道路上碰撞获得的道具放入其中。

3.2.3 不同难易度与趣味度的游戏模式设计

游戏需要设计不同难易度与趣味度的游戏模式，来符合各个层次的玩家的需求。如图 3-8 所示。

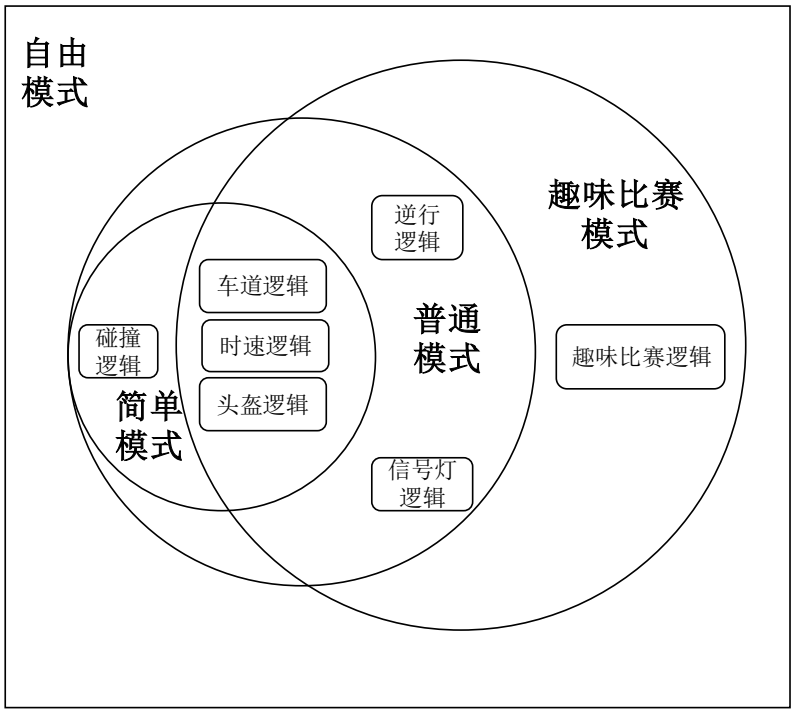

图 3-8 游戏模式设计韦恩图

自由模式下，游戏会关闭所有交通检测逻辑，保留地图上的道具和移动中的 AI 车辆、AI 行人，供玩家自由探索地图；

普通模式下，游戏会开启所有的交通检测逻辑，该模式是游戏的默认模式，玩家可以点开暂停界面，选择切换模式；

简单模式下，游戏将关闭比较容易出错的逆行逻辑、信号灯逻辑，保留比较简单的车道逻辑、时速逻辑、头盔逻辑，让玩家能够通过该模式逐渐熟悉游戏的玩法。

最后是趣味比赛模式，这一模式最为困难，游戏将启用四个与玩家类似的电动车 AI，除了可以与他们之间发生碰撞以外，依旧不能违反交通规则，画面右侧将启用一个排名 UI，显示当前五辆电动车离目的地的排序。道路上的道具也会新增干扰类道具，可能会给玩家或者 NPC 带来负面影响。

3.3 设计中考虑的制约因素

社会因素考量 本趣味游戏依照现实生活中的交通法律法规进行设计，且课题实施过程中充分考虑了社会层面的各项需求与制约，因此整个游戏在游玩过程中不会有诱导游戏成瘾的机制，玩家体验轻松愉快，整个流程安全健康。

法律风险规避 本趣味游戏在设计和实现过程中，使用的开发工具、开发平台均为正版，代码库、游戏模型、游戏贴图均为开源或可供研究使用，整个开发过程无法律风险。

文化风险规避 本趣味游戏在实现过程中代码不存在反动、暴力、低俗词汇或其他不当言论，代码的审查工作在编写的时候同步完成。

3.4 本章小结

本章介绍了本课题的开发环境，设计了开发流程、运行流程，并对游戏系统的各项游戏对象进行了分层设计，之后划分了代码的功能模块，并对车辆动力学模块的系统结构以及 UI 模块的界面进行了分析与设计，最后设计了不同难易度与趣味度的游戏模式，让整个游戏流程能够循序渐进，妙趣横生，寓教于乐。

4 游戏系统实现

本章开始，将依据之前的模块划分，进行各模块的游戏系统的实现。

4.1 车辆动力学系统实现

4.1.1 车辆动力学参数绑定

依照第三章中的设计所述，由于 AI 汽车和玩家主体电动车的动力学系统结构不同，实际的物理参数也很不一样，所以需要分别为其设置各自的合乎物理规律的动力学参数：

（1）AI 汽车的动力学参数绑定

对于 AI 汽车的轮胎，需要设置系统的 Wheel Collider 组件，它的参数绑定如表 4-1 所示：

表 4-1 AI 汽车轮胎参数绑定

参数名称	说明	数值
Mass	质量，单位为千克	20
Radius	轮胎半径，单位为米	0.335
Wheel DampingRate	车轮受到的阻尼大小	0.25
Suspension Distance	悬架在局部空间沿 Y 轴延伸的最大距离	0.2
Force App Point Distance	车轮的受力点。例如，如果此值为 0 的话，那么受力点就在车轮的底部。一般会设置为离车轮质心较低的位置	0.1
Suspension Spring	悬挂系统的弹簧力，该值会影响车辆颠簸时的用时以及车轮的载荷量	70000
Suspension Damper	悬挂系统的阻尼，较小的值会显得弹簧更有弹力	3500
Target Position	悬挂系统静止时的车轮位置，用来描述悬挂系统的缩张程度	0.1

对于 AI 汽车的车身，需要为它设置刚体组件，它的参数绑定如表 4-2 所示：

表 4-2 AI 汽车车身参数绑定

参数名称	说明	数值
Mass	质量，单位为千克	1000
Drag	针对于力的空气阻力	0.1
Angular Drag	针对于扭矩的空气阻力	0.05

（2）玩家主体电动车的动力学参数绑定

对于玩家主体电动车的轮胎，本课题使用开源的 WSP Collider 进行设计，其实现原理与系统自带 Collider 类似，需要绑定的参数有所增加，但这也方便了课题中更多的功能实现，例如电动车的音效。它的关键参数绑定如表 4-3 所示：

表 4-3 电动车轮胎关键参数绑定

参数名称	说明	数值
Wheel Radius	轮胎半径，单位为米	前轮：0.325 后轮：0.335
Wheel Mass	轮胎质量，单位为千克	20
Suspension Length	悬挂长度	0.28
Target	悬挂系统静止时的车轮位置，用来描述悬挂系统的缩张程度	0
Spring	悬挂系统的弹簧力，该值会影响车辆颠簸时的用时以及车轮的载荷量	25000
Damper	悬挂系统的阻尼，较小的值会显得弹簧更有弹力	1000
Max Motor Torque	发动机的最大扭矩，单位为牛·米	前轮：0 后轮：20000
Max Brake Torque	制动的最大扭矩，单位为牛·米	10000
Max Steer Angle	车把的最大转向角度	前轮：30 后轮：0

此外，与 AI 汽车的轮胎碰撞器不同的是，玩家电动车的轮胎碰撞器可以直接与电动车模型进行绑定，使电动车的物理数据可以直接反映在模型的变化上。此外，我们还可以通过设置前侧向的摩擦系数实现道路的模拟。其绑定模型参数及其他参数绑定如表 4-4 所示：

表 4-4 电动车轮胎绑定模型参数及其他参数绑定

参数名称	说明	数值
Rigid Body	与所属的电动车刚体进行绑定	对应模型部件
Steering Transform	与所属的电动车的手把部件绑定	对应模型部件
Suspension Transform	与所属的电动车的悬挂部件绑定	对应模型部件
Wheel Transform	与轮胎部件绑定	对应模型部件
Forward Friction Coefficient	前向摩擦系数（进行道路模拟）	5
Side Friction Coefficient	侧向摩擦系数（进行道路模拟）	5
Surface Friction Coefficient	表面摩擦系数（进行道路模拟）	5
Sweep Type	悬挂系统的碰撞模拟方式，可以选择光线投射/球体/胶囊体三种方式，这里使用胶囊体来模拟	CAPSULE
Friction Model	轮胎的打滑模型，使用标准模型	STANDARD

对于玩家主体电动车，需要为它设置刚体组件，它的参数绑定如表 4-5 所示：

表 4-5 电动车车身参数绑定

参数名称	说明	数值
Mass	质量，单位为千克	250
Drag	针对于力的空气阻力	0.1
Angular Drag	针对于扭矩的空气阻力	0.05

以上是基础的需要进行绑定的参数。为了更好地利用这些参数，本系统对这一车轮碰撞器进行包装，并额外提供下列参数：

Rpm Limit（车轮转速限制，单位为转数每分钟）：20000

Throttle Response（发动机进行加速的快慢）：前轮：0；后轮：10

Steering Response（车把转向速度的快慢）：前轮：10；后轮：0

Brake Response（发动机减速/制动速率的快慢）：前轮：30；后轮：30

以上四个参数是为了更方便进行条件判断，或是调整速度的变化快慢而特意

设置，与车轮碰撞器本身并无关联，具体的实现方式将在后文进行更详细地描述。

4.1.2 电动车输入系统实现

游戏使用 FixedUpdate()作为电动车运动的更新频率，使用 Input.GetAxis("Ve-rtical")作为电动车后轮的加减速控制，使用 Input.GetAxis("Jump")作为电动车的制动控制，使用 Input.GetAxis("Horizontal")作为电动车的转弯控制。

首先游戏需要判断当前车轮的转速是否已经超过了游戏设置的 Rpm Limit，也就是转速上限，如果未超过，那么就可以继续向电动车施加扭矩，否则施加的扭矩直接变成 0。

在对碰撞器进行初始化的阶段，游戏生成了一个球形碰撞器包括车轮，并进行碰撞检测，但由于这个球形碰撞器不是游戏通过编辑器添加的，所以在电动车运动的过程中可以通过如下伪代码来实现球形碰撞器对电动车车轮模型的跟随：

球形碰撞器的位置 = 插值函数(当前位置, 目标位置, 帧时);

之后游戏就可以对当前施加给电动车的扭矩的增减进行插值运算，为了保证增减不会受到硬件自身的帧率影响，游戏需要以 FixedDeltaTime 作为插值的比例变量。同时，为了调节插值速率的快慢，可以将先前设置的 Throttle Response（发动机进行加速的快慢）、Steering Response（车把转向速度的快慢）、Brake Resp-onse（发动机减速/制动速率的快慢）乘上 FixedDeltaTime 作为这个比例变量，只需要调节这三个参数的大小就可以调节摩托车的加减速、转向、制动的快慢，下面仅给出针对发动机情形的伪代码：

*当前发动机扭矩 = 插值函数(当前发动机扭矩, 输入扭矩 * 最大扭矩, 发动机响应参数 * 帧时);*

将插值之后的扭矩输入到车轮碰撞器，游戏就可以对车轮的运动情况进行改变。

需要注意的是，尽管运动状态改变，电动车模型却不会有相应变化，这需要游戏对其进行变换：

以车把的自身局部坐标系的 Y 轴为轴，游戏需要将其模型以及人物的上半身做一些旋转；悬挂系统根据当前的挤压程度会有一定的上下移动；轮胎以自身的 X 轴为轴，按照设定的帧旋转角度进行旋转，要注意的是这个旋转发生在世界空

间而不是局部空间。

游戏已经实现了电动车基本的运动状态及其对应的模型变换，但接下来还有一个重要需求等待游戏实现，那就是电动车在转弯时不同于汽车，会有一定的倾侧效果。

4.1.3 电动车转弯时侧倾效果实现

在前文已经提到，电动车系统结构中包含了一组 Stabilizers 组件，共四个，分布在电动车前轮的左右两侧、后轮的左右两侧。当想要让电动车在转弯时发生倾侧效果的时候，只需给这些稳定器向上或者向下的压力就可以实现。

为了保持电动车在转弯时最基本的稳定性，游戏需要给予车辆一个向后和向下的力。如图 4-1 所示，游戏首先取得当前转向的角度与允许的最大角度的比例，再获取当前电动车的角速度。

然后，游戏根据角速度的 z 轴分量的大小，给予其反方向的阻力；根据角速度 y 轴分量以及当前的转向比例，给予其向下的压力。实现伪代码如下：

```
补偿扭矩 = 后向量 * 角速度z 轴分量 + 下向量 * 角速度y 轴分量 * (1f - Mathf.Abs(转向角));
给刚体施加扭矩(补偿扭矩 * 稳定器数量, 模式：改变速度);
```

速度也是影响电动车稳定性的一个因素，对于电动车来说，速度较小的情况下倾侧程度会迅速攀升，但是速度达到一定程度的时候就又能保持着这样的倾侧程度，因此游戏需要用一个类似对数的曲线来描述这样的变化。用如下伪代码来表示：

```
倾侧参数 = 速度值-倾侧曲线图(速度值);
```

之后游戏便开始遍历相应的 Stabilizers 组件。首先取得每个稳定器的的坐标和下方向矢量，并通过它的局部 z 坐标，来确认它究竟是前轮稳定器还是后轮稳定器。

之后游戏取得当前的制动扭矩与最大制动扭矩的比例，这个比例越大，则这一轮胎相对应的另一边的稳定性会相应增加。对应伪代码如下：

```
制动因子 = (是否是前轮 ? 前轮制动因子 : 后轮制动因子) * 0.8f;
制动参数 = 1f – 制动因子 * 取最小值(速度值 / 3f, 1f);
```

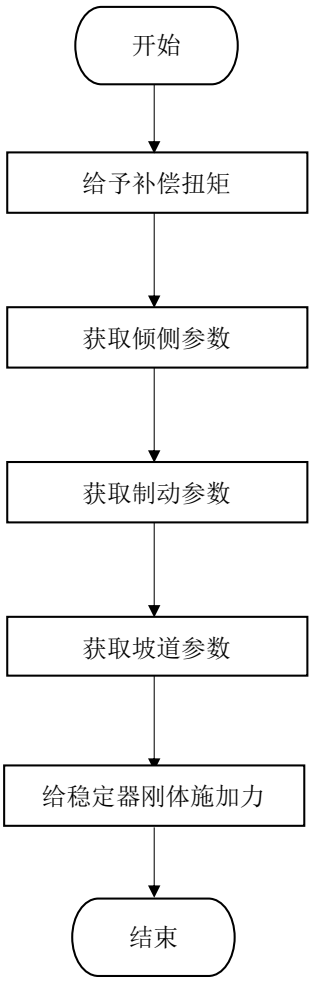

图 4-1 电动车转弯时倾侧效果实现流程图

在上下坡的时候，游戏需要禁用这一倾侧功能，否则电动车会变得非常难以控制，伪代码如下：

```
坡道参数 = 取最大值(0, 光线投射向量 y 分量);
```

最关键的影响倾侧程度的因素还是电动车当前的线速度以及角速度，刚刚游戏已经获得了线速度的影响因素 tiltFactor，将它与当前的转向角度相乘，同时还需要乘以当前遍历的稳定器的本地 x 坐标的符号来确定是给予上压力还是下压力，例如，电动车右转的时候，前轮左稳定器的转向符号为正，本地 x 坐标为负，所以给予上压力，也就是车往右边倾侧。要注意的是，离地面越近，这个系数就越小。

最后游戏将以上系数与稳定器的下向量相乘，作为 AddForceAtPosition()函数

的第一个参数，伪代码如下：

```
错误补偿 = 目标地面距离 – 光线投射距离 – 倾侧参数 * 倾侧因子;
给刚体施加力(稳定器下方向 * -错误补偿 * 稳定器数量 * 制动参数 * 坡道参数, 稳定器位置, 模式：改变速度);
```

这样就能够在考虑了各种情况的影响下，很好地模拟出了电动车转弯时的倾侧效果。

4.1.4 电动车音效实现

电动车的音效实现，与电动车自身的物理数据是紧密联系的。只要将实时的物理数据通过一些映射，转化为音效的放缩倍率数值，就能实现具有真实感的音效，一部分伪代码如下：

```
速度因子 = 取最小值(速度值 ÷ 25, 1);
发动机因子 = 速度因子 * 后轮转速 * 0.5;
撞击地面因子 = 取最小值(y 方向速度, 1);
制动因子 = 前轮转速 + 后轮转速;
```

发动机因子将影响到发动机的音量与音调，撞击地面因子将在车辆有 y 方向分量的速度时且轮胎刚接地时触发撞击音效，制动因子将和速度因子一起共同影响制动时的音量与音调。

4.2 UI 界面实现

UI 界面与游戏对接则依靠游戏的两个实现代码：PenaltyController 和 Menu Controller，也就是惩罚控制器和界面控制器，两个代码挂载在 GameController 这个 GameObject 上，这一游戏对象与电动车的初始位置重合，它也有重置玩家位置的功能，将在后文介绍其具体实现。

4.2.1 游戏界面控制器实现

首先游戏需要一些数据结构，与编辑器里的 UI 对象进行连接以便于对应逻辑功能的调用：

```
[SerializeField] //以下数据结构都需要通过这个特性定义
```

```
private GameObject posInitial; // 与 GameController 连接
private Text TextVelocity; // 电动车当前速度
private GameObject screenPause; // 暂停界面
private GameObject screenGame; // 一些游戏按钮
private GameObject screenHelp; // 帮助指南界面
private GameObject screenExplanation; // 游戏说明界面
private GameObject screenPenalty; // 违规行为界面
private GameObject screenGameOver; // 游戏失败界面
private GameObject screenGameClear; // 游戏成功界面
private GameObject screenVehicle; // 玩家电动车
```

如图 4-2 所示，游戏开始时，screenGame 和 screenPenalty 会启用，其他界面全部隐藏。游戏更新过程中，游戏画面左上角会实时更新车辆的速度，右上角会更新剩余的时间。如果玩家按下键盘上的“p”键，则激活 screenPause，暂停界面上有帮助指南、游戏说明、继续游戏、重新开始游戏等按钮，前两者直接点击就可以进入对应界面，点击按钮的功能可以通过 Unity3d 的编辑器实现。

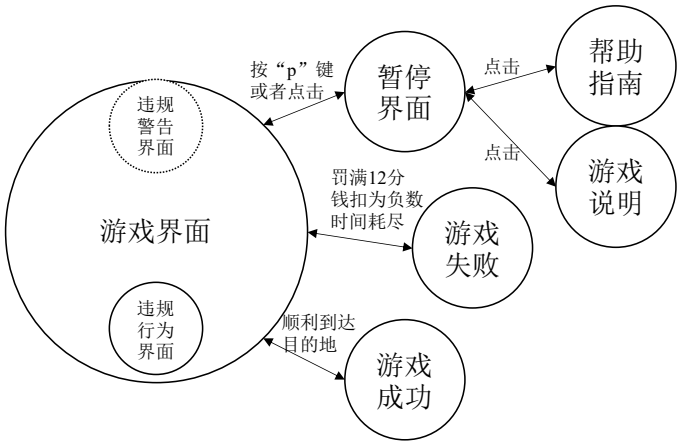

图 4-2 游戏界面状态转移图

想要让游戏进入暂停状态，只需要将 Time.timeScale 这一变量从 1 修改为 0 即可。在编写这一类 UI 切换的函数的时候，游戏可以利用 SetActive()这一函数来实现启用和禁用的功能。暂停页面、帮助指南页面和游戏说明页面的功能实现都要用到这一函数。

重新开始游戏的功能需要游戏将玩家电动车的位置重置，将玩家电动车的运动状态重置，将倒计时和罚分、钱包余额重置。

4.2.2 游戏惩罚控制器实现

首先游戏需要一些数据结构，与编辑器里的 UI 对象进行连接以便于对应逻辑功能的调用：

```
[SerializeField] //以下数据结构都需要通过这个特性定义
private Text TextScores //罚分
private Text TextMoney // 钱包余额
private Text TextLevel6 // 扣 6 分的行为次数
private Text TextLevel3 //扣 3 分的行为次数
private Text TextLevel1 // 扣 1 分的行为次数
private Text TextWarn // 口头警告的行为次数
private GameObject screenShowPenalty // 展示违规警告界面
private Text TextReason // 违规理由
private Text TextLevel // 违规级别
private Text TextFines // 罚款数量
```

每一次更新时，游戏需要更新的是上述数据结构中的上半部分，同时检查当前帧是否发生了违规现象，一旦发生，就跳转到执行方法。

至于如何判断违规现象是否发生，可以在该类代码里添加一个静态方法，名为 ApplyPenalty，通过将判断变量 fined 设为真来实现这一判断。在这一方法里，不同的违规现象也需要对不同的变量进行设置。

执行方法里，游戏需要对相应变量进行罚分、罚款，对相应行为次数进行自增，同时设置上述数据结构的下半部分的文本内容。然后游戏将调用一个时长 3 秒的协程，这个协程同时还会挂载一个警告音效，在屏幕上方显示，向玩家展示刚刚的违规行为。

4.2.3 游戏水平导航仪实现

首先游戏需要创建一个 Waypoint 的数据结构，来存放相应的导航对象，代

码如下所示：

```
public class Waypoint {
    public Transform icon;
    public Transform target;
    public Transform obj;
    public CanvasGroup grp;
}
```

对于水平导航仪，游戏至少需要以下接口：

```
public Transform player; // 玩家
public Transform north; // 北
public Transform south; // 南
public Transform east; // 东
public Transform west; // 西
public Transform star; // 目的地
```

为了实现接口与 Waypoint 这一数据结构的对接，可以通过字典来实现，例如，输入一个“north”字符，获得相应的 Waypoint 类的结构，其中 icon 和 obj 变量对应这些 Transform 对象，target 则对应玩家自己。

接下来的问题就是要如何设置 icon 的位置，可以编写一个 SetPosition 函数来实现：

```
偏移向量 = 转换到局部坐标(玩家位置+导航方向)
偏移角度 = tan(偏移向量.x ÷ 偏移向量.z)
偏移位置 = 右向量 * 宽度 * 偏移角度 ÷ (2π)
屏幕位置 = (偏移位置.x, 屏幕位置.y, 0) // 屏幕位置.y 是游戏预先就已经设置好了的
如果偏移角度小于-1.58 或大于 1.58
    不启用对应图标
```

4.2.4 游戏背包系统实现

道路上分布着一些对玩家游戏有帮助的道具，背包里也会初始化一个头盔，

这些道具也需要放到背包里，所以需要一个简单的、类似趣味赛车的小型 UI 背包，并将它放在屏幕的左下角。

对于道具，游戏需要一个数据结构来存放它的名字、类型、功能说明，如下：

```
public class item {
    private int m_id; // 道具 ID
    private string m_name; // 道具名
    private string m_description; // 道具描述
    private string m_type; // 道具类型
}
```

为了界面的美观，同时督促玩家及时使用道具，游戏只设置三格空间的背包，UI 位于屏幕的右下角。当道具进入这一 UI，鼠标移动到道具上并悬停 0.3 秒以上的时候，需要显示出道具的简介。这一功能可以通过编写一个类，让这个类继承 IpointerEnterHandler 和 IpointerExitHandler 这两个父类，这样游戏就可以使用 pointEntered 这一变量来检测鼠标的移入，timer 这一变量来检测悬停的时间长短。

游戏首先通过 Instantiate()方法复制一个小面板作为道具介绍的背景，设置它的位置在当前道具的上方，然后再设置面板里的具体内容：由于是复制的面板，所以要首先销毁已有内容，再一一设置新添内容里的文字的父子关系、文字内容、文字字体。想要编辑文字格式的话，可以用 string.Format 方法来实现。

对于背包，游戏通过下列数据结构来存放一些重要变量：

```
public class Pack : MonoBehaviour
{
    private GameObject[] cell; // 背包
    public GameObject grid; // UI 格子
    public GameObject helmet; // 玩家自带的头盔实体
}
```

在游戏开始的时候，初始化一个三格的背包，并在背包中初始化一个电动车头盔，放在背包的第一格。按 1 可以激活头盔游戏对象。另外两格为空，按 2、3 可以使用之后这两格之后存放的道具。

如图 4-3 所示，当玩家与道路上的道具触发器发生碰撞后，游戏将会获取到

道具的名字与标签。通过名字游戏查找到对应道具实体，并将该实体的各项信息添加到游戏利用 Instantiate()方法复制的预制体上。然后游戏加载相应的资源图片，并显示到背包格上。最后游戏还需判断当前背包是否已满，如果未满，则将该实体的父体设置为背包，否则不作任何设置。

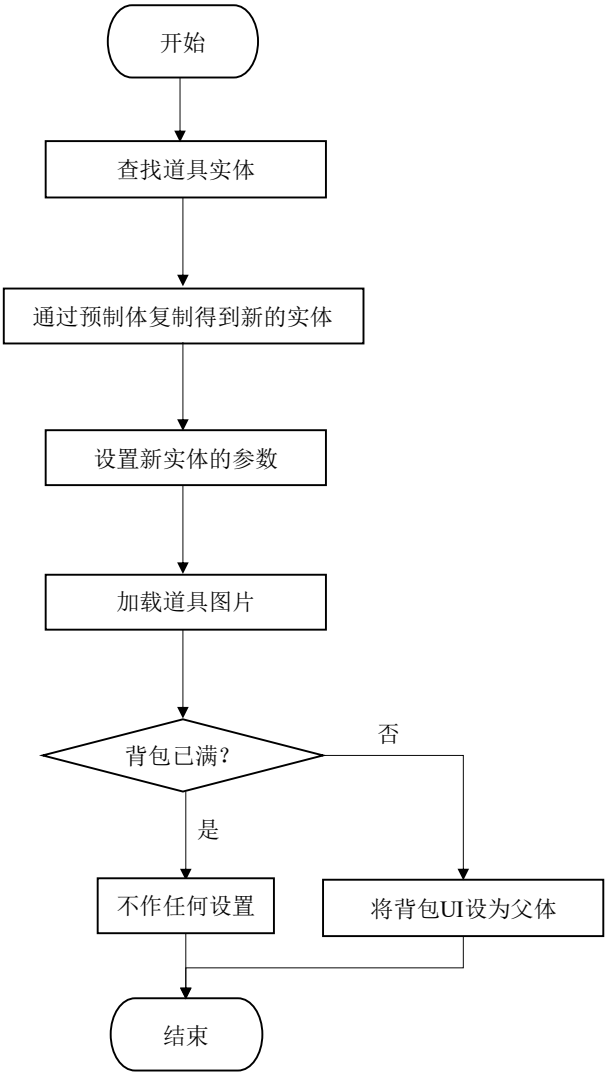

图 4-3 道具拾取流程图

4.3 AI 车辆、行人寻路实现

4.3.1 AI 行人寻路实现

对于 AI 行人，游戏的基本需求是让他们走在人行道上，但有时也可能横穿马路，经过人行横道的时候，如果是红灯将调用 Random.Value()进行随机，行人

有一半的概率会停止行走或者无视红灯继续行走，绿灯则会继续行走。

游戏实现的这一寻路功能将基于 Unity3d 自带的 NavMeshAgent 功能，再通过在道路上添加一些触发器来判断他们是否进入人行横道，通过信号灯类提供的一些接口来判断当前是红灯还是绿灯。

由于人物的移动功能涉及到一些动画实现，游戏直接利用 Unity3d 的 Unity StandardAsset 组件里的人物移动代码实现。因为 AI 只需要基本的行走，所以在游戏中可以禁用蹲伏和跳跃功能。

4.3.2 AI 车辆寻路实现

汽车的驾驶方法与电动车类似，如图 4-4 所示：

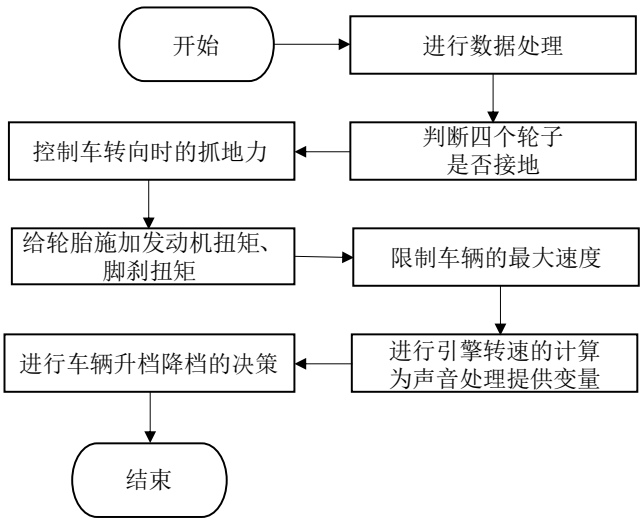

图 4-4 汽车移动流程图

汽车的驾驶应该比 AI 路人的行走决策要谨慎得多，所以游戏不可以简单依靠 Unity3d 自带的 NavMeshAgent 功能，而是自己编写一个判断 AI 汽车驾驶的谨慎程度的方法，如图 4-5 所示：

游戏可以根据目标偏移与当前速度矢量的角度判断谨慎程度，或者根据 AI 汽车与目标之间的距离判断谨慎程度，也可以两者综合在一起来判断谨慎程度。通过一系列参数计算出当前的谨慎因子，再由这一因子决定当前输入车辆的扭矩大小。

AI 汽车的碰撞体相比与它自身要大许多，当有其他 AI 汽车进入它的碰撞体

之时，游戏也要进行回避。游戏首先要判断两者的前后位置，然后让后面的车辆减速。

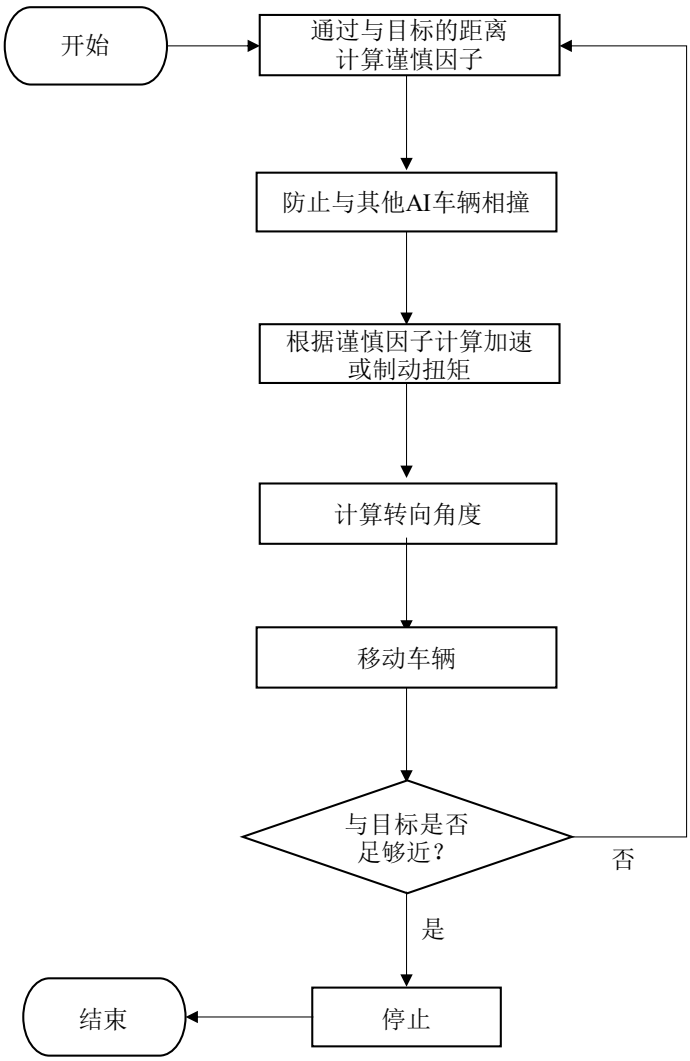

图 4-5 AI 汽车寻路流程图

这一流程图只是在单目标情况下，AI 汽车的行为。但实际上，为了不让 AI 汽车到达目标点之后就停在路边，游戏至少要它们的行驶形成一个闭环。这就需要游戏再在 AI 汽车上挂载一个自动更新脚本，这个脚本能不断更新目标点集合里新的目标点。

在趣味比赛模式下，游戏新增的四个电动车 AI 也需要挂载这一寻路功能，并合理设计他们的目标点集合，这次的集合就无需再形成闭环。

4.4 交通法律法规的检测与惩罚实现

如图 4-6 所示，游戏使用各类触发器来进行法规检测的逻辑判断。

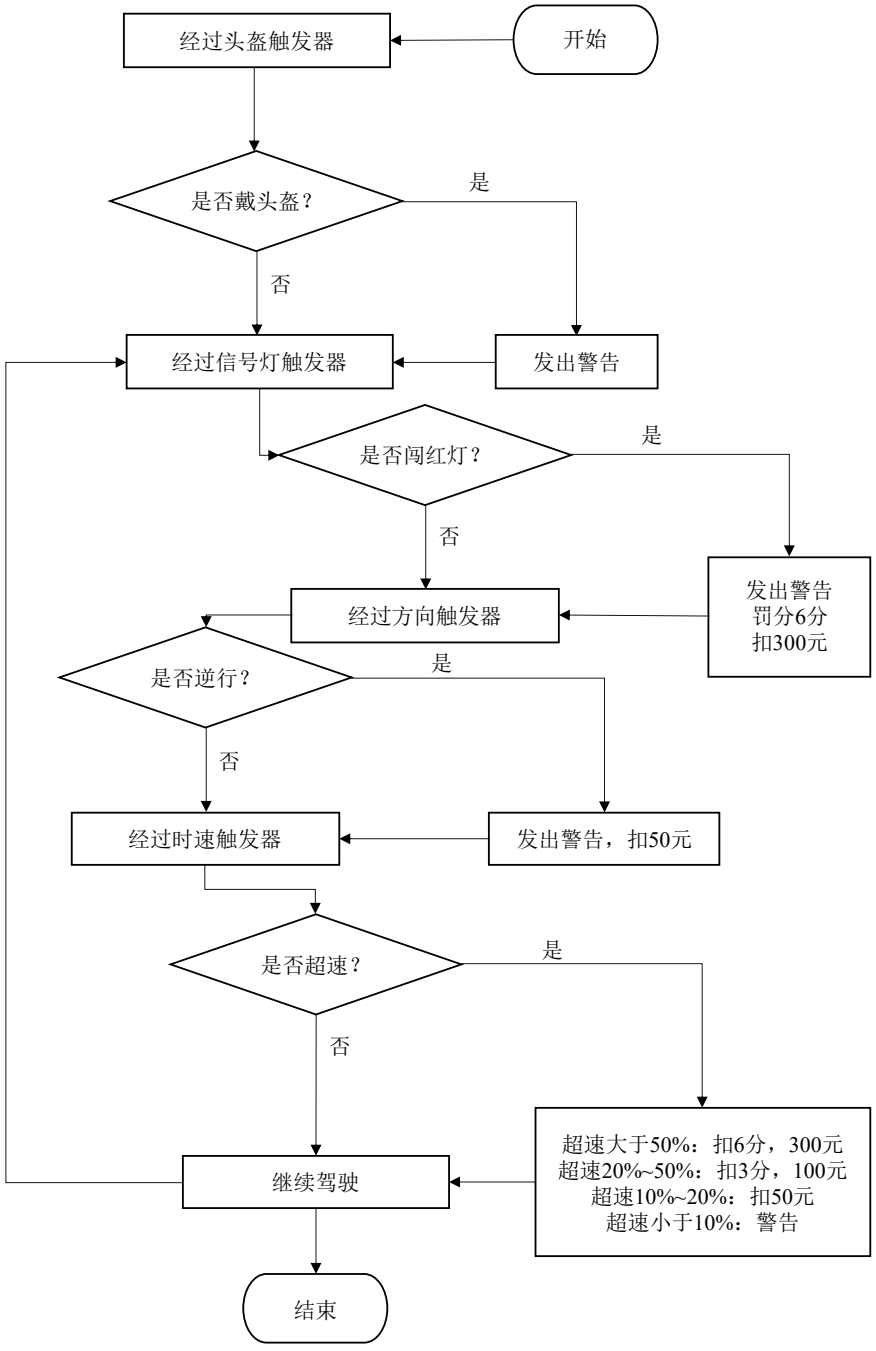

图 4-6 交通法律法规判断流程图

游戏使用碰撞检测来进行这些逻辑的实现。玩家一旦经过各种触发器，并通过触发器的标签，检测到相应的违规行为，游戏都会作出相应的惩罚，并反映在

UI 界面上让玩家知晓。

游戏设置了两个头盔触发器，置于游戏起点与终点附近，经过触发器时，如果玩家的背包里仍然存在没有使用的电动车头盔，将在画面显示警告，并从钱包扣除 50 元；

游戏设置了多个时速触发器，在经过时检测速度是否超过 30km/h，并根据超过的程度进行相应的罚分、罚款；

游戏设置了四个有红绿灯的十字路口，红绿灯的交替我们通过协程来实现，红灯 6 秒，绿灯 6 秒，黄灯 1 秒。违反交通信号灯是严重的违章行为，将罚分 6 分，罚款 300 元；

游戏设置了道路触发器，不可以将电动车行驶到人行道上。也设置了逆行触发器，现实里的道路上会有箭头标识，游戏中的触发器也设置在这些标识上，玩家经过时游戏通过车辆行驶方向的向量与触发器的前向量的点乘来判断玩家是否逆行，如果是，则扣 50 元；

有三种可能性会导致游戏失败，分别是：与 AI 车辆发生碰撞；与 AI 行人发生碰撞；游戏时间耗尽。游戏对玩家电动车的碰撞模型设置较为精密，与 AI 行人擦肩而过不会判定为发生碰撞。在趣味比赛模式下，与 AI 比赛电动车发生碰撞不会导致游戏失败。

4.5 软件开发成本估算

本课题使用 Putnam 模型估算本课题的开发成本，该模型是一种动态多变量模型，能较为准确地得到其中的需求参数。这些参数包括本课题的源代码行数、开发持续时间、开发技术因素。

对于代码行数，本课题使用互联网的开源统计工具进行统计，如表 4-6 所示，有效行数共 2317 行，共 23 个代码文件。除编写代码以外，本游戏共创建并编辑了 54 种游戏对象，编辑器编辑时间可换算为大概 500 行代码的编写时间。

表 4-6 代码统计数据

代码行 2317	注释行 236
空行 391	总行数 2944

对于开发持续时间，本课题从 2022 年 3 月上旬开始，至今约 2.5 个月的时

间，也就是 0.21 年。

对于开发技术因素，本课题基于成熟的 Unity3d 编辑器开发，开发工具和软件库有详细的介绍文档，开发环境较好，因此该因素取值为 8000。

按照 Putnam 模型公式，计算得到工作量为 22.45。

4.6 本章小结

本章对整个电动车安全驾驶趣味游戏进行了系统的具体实现，首先研究了车轮碰撞器的动力学系统，进行参数的绑定，设计了输入系统、音效，实现了其转弯时的倾侧效果。为了更好地服务于游戏逻辑实现，设计了 UI 界面，在道路上规划了 AI 汽车与 AI 路人的行进路线，并对 AI 汽车的寻路方法进行了各种限制处理。依照现实的交通法律法规，实现了各项法规的检测与相应惩罚。最后估算了软件开发的成本。

5 游戏测试与优化

本章开始，将介绍游戏的测试方案、优化方案以及发布步骤。

5.1 系统功能测试

5.1.1 测试方案

对于电动车移动的测试，设计如表 5-1 的测试方案：

表 5-1 电动车移动测试方案

编号	用户输入	预期结果
1	输入 w	电动车加速
2	输入 s	电动车减速
3	速度较小时，输入 a 或 d	电动车转向，车把旋转，玩家身体旋转
4	速度较大时，输入 a 或 d	电动车侧倾转向，车把旋转，玩家身体旋转
5	输入空格	电动车刹车

对于 UI 界面显示的测试，设计如表 5-2 的测试方案：

表 5-2 UI 界面显示测试方案

编号	用户输入或触发条件	预期结果
1	鼠标点击暂停键	游戏暂停
2	输入 p	游戏暂停
3	点击暂停界面的“继续游戏”	游戏继续
4	点击暂停界面的“游戏说明”或“操作指南”	展示游戏说明界面或操作指南界面
5	输入 a 或 d 使电动车转向	上方的水平罗盘 UI 的指向发生变化
6	输入 1 或 2 或 3	使用背包界面 1 号位或 2 号位或 3 号位的道具
7	玩家进行了一段时间的游戏	左上角时间改变

对于交通法律法规检测与惩罚的测试，设计如表 5-3 的测试方案：

表 5-3 交通法律法规测试方案

编号	触发条件	预期结果
1	与车辆或者行人发生碰撞	展示游戏失败界面
2	时速超过 45km/h	罚分 6 分，扣 300 元
3	闯红灯	罚分 6 分，扣 300 元
4	时速超过 36km/h	罚分 3 分，扣 100 元
5	没有在规定车道上行驶	罚分 3 分，扣 100 元
6	逆行	扣 50 元
7	经过人行横道时，没有避让行人	扣 50 元
8	不戴头盔	罚分 1 分，扣 50 元
9	时速超过 33km/h	扣 50 元
10	时速超过 30km/h	警告
11	罚分满 12 分	展示游戏失败画面

5.1.2 测试结果

以上测试结果均顺利通过，下面展示两个典型场景：

如图 5-1 所示，当玩家闯红灯的时候，屏幕上方会显示出警告 UI 界面，同时罚分 6 分，钱包余额从 1500 元变为 1200 元，6 分违规次数也增加 1 次。

图 5-1 闯红灯场景截图

如图 5-2 所示，玩家在转弯的时候，电动车会有向右侧倾的效果。右下角的背包栏里有一个还没有戴上的头盔，玩家需要及时按 1 戴上。

图 5-2 电动车侧倾效果

5.2 系统性能测试

5.2.1 测试方案

本课题使用 Unity 官方提供的 UPR（即全面周期性检测）工具来进行性能测试，该工具无需依赖 Unity 编辑器，它通过扫描打包好的 Unity 工程文件进行资源检测，展示出资源之间的依赖关系以进行冗余排查，还能利用其静态代码检测功能来检查代码的问题隐患。

UPR 还集成了内存快照、RenderDoc 等功能，可以帮助开发者快速定位游戏瓶颈并进行优化。

5.2.2 测试结果

根据 UPR 工具的测试，测试结果如表 5-4 所示：

表 5-4 UPR 测试结果

测试评分 88（优秀）	平均帧率 61.88
ReservedMono 峰值 2.4MB	音频资源峰值 5.6MB
纹理资源峰值 230.4MB（偏高）	DrawCall 峰值 3957 次（偏高）

其中，纹理资源加载的峰值和图形渲染时的 DrawCall 次数偏高，但由于游戏平均帧率为 61.88，且 UPR 工具在进行 Windows 端游戏测试的时候这些值会有所偏高，因此它们并未给游戏造成较大影响。

5.2.3 优化方案

（1）设置 DrawCall 合批顺序；

（2）渲染前进行预处理，加大对不可见游戏对象的裁剪力度；

（3）减少模型的顶点数量；

（4）使用压缩贴图；

（5）对趣味比赛模式下的 AI 电动车进行模型简化处理。

5.3 本章小结

本章对游戏进行了功能测试与性能测试，利用 Unity 官方提供的 UPR 工具对工程进行了详细的性能测试，并有针对性地进行了优化。

6 总结与展望

随着游戏软件与计算机硬件的高速发展，市面上各种各样的模拟器也层出不穷。本课题以设计并实现一个校内趣味电动车模拟驾驶安全游戏为中心，以华中科技大学的校园道路路线为基础，在系统中设计了 AI 模块、玩家电动车模块、交通法律法规检测与惩罚模块、系统 UI 模块，游戏运行流畅，玩法多样，让玩家既能够顺利愉悦地游玩，也能够深刻认知到各种各样的交通法律法规。

在这个过程中，我做了以下几点工作：

（1）深入学习了 Unity3d 引擎的结构以及代码的编写模式，广泛调研了国内外的虚拟校园系统实现方式、交通安全系统实现方式、机动车驾驶仿真实现方式、寻路 AI 实现方式；

（2）分析了本课题的各个方面的可行性，并分析了 Unity3d 引擎的关键技术；

（3）分析了游戏的各项需求，制定了游戏的基本方案，并对游戏总体系统和各个模块进行了详细的设计，使游戏逻辑既符合交通规则，又极富趣味性；

（4）实现了游戏的各项需求对应的功能，并对实现内容进行了各项测试与性能优化。

本课题已有较高的完成度，但依旧可以添加一些高级需求，例如：

（1）可以研究 Unity 的多玩家联网组件，并尝试让游戏实现联机比赛功能；

（2）可以下载一些 3D 建模软件，尝试自己定制贴图或模型，完成相应需求；

（3）可以对 Unity Shader 进行深度研究，在游戏里实现更高质量的实时渲染，例如天气系统。

致 谢

论文完成之际，首先要感谢我的导师黄志老师。他严格地检查我的毕设制作进度，仔细地筛查我的论文的疏漏，使我能够顺利完成毕设。我要感谢党和国家，感谢养育我的父母，感谢计算机学院里辛勤授课的老师们。我还要感谢让我了解到图形学的乐趣的闫令琪老师，为图形学作出杰出贡献的毛星云等网友，让我免费上了一些 Unity 网课的 SIKI 学院的老师，互联网上无数写博客、提供开源代码的无名英雄的网友们。

在我做毕设的过程中，我也遇到了许多困难，我要感谢为我提供欢乐的计算机学院的群友们，感谢经常与我一起吃饭聊天的阿霞、希诺等社团校友们，感谢在此期间一直生产着优质内容以供我观看欣赏的徐云流、王雀孙、山田尚子、高桥哲哉、杉井光等创作者们。

认识与实践，无论何时我都想让自己处于这两个行为的循环里，读万里书，行万里路，在毕业以后也能不断地充实自己，提升自己。

参考文献

[1] Lars Bishop, Dave Eberly, Turner Whitted, et al. Designing a PC game engine[J]. IEEE Computer Graphics and Applications, 1998, 18(1): 46-53.

[2] David Eberly. 3D game engine design: a practical approach to real-time computer graphics[M]. 2nd edition. CRC Press, 2006. 1-1015.

[3] Patrick Cozzi, Kevin Ring. 3D engine design for virtual globes[M]. 1st edition. AK Peters/CRC Press, 2011. 1-490.

[4] Johan Andersson. Parallel Futures of a Game Engine v2.0[C]. in STHLM Game Developer Forum. 2010.

[5] 程彬彬, 王明鑫, 商楠. 浅谈游戏开发平台 Unity3D 的应用与发展前景[J]. 通讯世界, 2016, 14: 236.

[6] John Haas. A history of the unity game engine[D]. Diss.WORCESTER POLYTECHNIC INSTITUTE, 2014.

[7] Arthur Juliani, Vincent-Pierre Berges, Ervin Teng et al. Unity: A general platform for intelligent agents[D]. arXiv preprint arXiv:1809.02627, 2018.

[8] 韦杨. 基于 Unity 的虚拟校园系统的设计与实现[D]. 广西大学, 2020.

[9] 冯新玲. 三维虚拟校园交互漫游系统的设计与实现[D]. 南京理工大学, 2018.

[10] 王梅亮, 顾键萍. 基于 Unity 3D 的交通安全教育和技能训练系统的关键技术研究与实现[J]. 电脑知识与技术, 2015, 11(29): 75-77.

[11] 黄晓玲. 基于虚拟现实技术的儿童交通安全教育系统的研究与实现[J]. 信息技术与信息化, 2019, 2: 135-139.

[12] X. L. Han, C. Y. Jiang, H. X. Ge. A modified coupled map car-following model based on application of intelligent transportation system and control of traffic congestion[J]. Acta Physica Sinica, 2007, 56(8): 4383–4392.

[13] D. Chen, J. Laval, Z. Zheng, et al. A behavioral car-following model that captures traffic oscillations[J]. Transportation Research Part B, 2012, 46(6): 744–761.

[14] N. Farhi. Piecewise linear car-following modeling[J]. Transportation Research Part C, 2012, 25: 100–112.

[15] L. Yu, Z. K. Shi, T. Li. A new car-following model with two delays[J]. Physics Letters A, 2014, 378(4): 348–357.

[16] T. Lu, Z. Liu, T. T. Liu, et al. Virtual vehicle simulation method based on car-following model[J]. Computer Engineering, 2016, 42(6): 305–309.

[17] Y. Li, H. Zhu, M. Cen, et al. On the stability analysis of microscopic traffic car-following model: a case study[J]. Nonlinear Dynamics, 2013, 74(1-2): 335–343.

[18] Chao Qianwen, Li Weizi, Bi Huikun, et al. A survey on visual traffic simulation: Models, evaluations, and applications in autonomous driving[J]. Computer Graphics Forum, 2020, 39(1):287-308.

[19] M. Xu, X. Xie, P. Lv, et al. Crowd behavior simulation with emotional contagion in unexpected multi-hazard situations[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2019, 51(3): 1–15.

[20] M. Xu, H. Wang, S. Chu, et al. Traffic simulation and visual verification in smog[J]. ACM Transactions on Intelligent Systems and Technology (TIST), 2019, 10(1): Article ID 3.

[21] Liu Cuijuan, Zhen Liu, Chai Yanjie, et al. Review of Virtual Traffic Simulation and Its Applications[J]. Journal of Advanced Transportation, 2020, 2020(7): 1-9.

[22] 万政, 李娟, 罗宇飞, 等. 基于 Unity 3D 的摩托车安全驾驶游戏[J]. 电脑知识与技术, 2021, 17(4): 74-76.

[23] 姜峰, 陈艳, 李滨城, 等. 基于 Unity3D 的虚拟驾驶系统的开发与应用[J]. 智能计算机与应用, 2017, 7(4): 39-41.

[24] 曾林森. 基于 Unity3D 的跨平台虚拟驾驶视景仿真研究[D]. 中南大学, 2013.

[25] 张茜. 基于 Unity3D 的汽车功能模拟与驾驶场景演示系统的设计和实现[D]. 东南大学, 2016.

[26] 王志岗. 基于 Unity3D 的自动寻路导航系统的研究[J]. 电脑知识与技术, 2019, 15(36): 209-211.

[27] 史宝明, 贺元香, 李岚. Unity3D 随机寻路算法设计[J]. 吉林师范大学学报（自然科学版）, 2022, 43(1): 128-133.

[28] Mas'udi, Nashrul Azhar, Eriq Muhammad Adams Jonemaro, et al. Development of Non-Player Character for 3D Kart Racing Game Using Decision Tree[J].

Fountain of Informatics Journal 6, 2021, 2: 51-60.