**编 号：**

**审定成绩：**

|  |  |  |  |
| --- | --- | --- | --- |
| **重庆邮电大学**  **毕业设计（论文）** | | | |
|  | | | |
|  | **中文题目** | **基于Unity3d引擎的飞行射击游戏** |  |
|  | **系统开发** |
| **英文题目** | **Development of a flying shooter game** |
|  | **system based on the Unity3d engine** |
| **学院名称** | **计算机科学与技术学院** |
| **学生姓名** | **付饶** |
| **专 业** | **计算机科学与技术** |
| **班 级** | **04011802** |
| **学 号** | **2018211484** |
| **指导教师** | **魏秉铎 讲师** |
| **答 辩 组**  **负 责 人** | **刘柯 副教授** |
|  | |

**2022年 5 月**

**重庆邮电大学教务处制**

学院本科毕业设计(论文)诚信承诺书

本人郑重承诺：

我向学院呈交的论文《基于Unity3d引擎的飞行射击游戏系统开发》，是本人在指导教师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明并致谢。本人完全意识到本声明的法律结果由本人承担。

年级

专业

班级

承诺人签名

年 月 日

学位论文版权使用授权书

本人完全了解重庆邮电大学有权保留、使用学位论文纸质版和电子版的规定，即学校有权向国家有关部门或机构送交论文，允许论文被查阅和借阅等。本人授权重庆邮电大学可以公布本学位论文的全部或部分内容，可编入有关数据库或信息系统进行检索、分析或评价，可以采用影印、缩印、扫描或拷贝等复制手段保存、汇编本学位论文。

（注：保密的学位论文在解密后适用本授权书。）

|  |  |
| --- | --- |
| 学生签名： | 指导老师签名： |
| 日期： 年 月 日 | 日期： 年 月 日 |

摘要

随着智能手机的发展，很多时候手机游戏受到人们的追捧，但受限于手机的尺寸和运行速度，所以手机无法满足大型游戏的需求。在游戏的过程当中，注意人机结合，因此本文课题是设计一款能够实现人机玩法和多人在线连线的飞行射击游戏。首先通过对本游戏需求进行整体规划，通过规划逐步实现模块化的内容，然后根据内容设计仿真，最后进行相应的功能验证。

本次设计在需求分析的基础上，利用 Unity3d引擎的强大的引擎自带的API功能，构建了3D场景多人联机射击游戏软件的整体框架，该框架包括物理引擎，UI引擎，渲染引擎，网络引擎。仔细的阐述了不同模块之间通过不同的方式连接，并且学习到游戏软件所需要的数据实现。通过U3D作为本游戏的核心引擎，在C#语言等相关编程语言基础支持下，完成对射击游戏功能需求的实现，测试了相关软件的运行情况、系统兼容、后期维护，并且将软件进行真人试玩测试。测试运行的结果和游戏整体运行状况，本论题设计的飞行类射击游戏能够基本实现用户对游戏的需求，用户体验感较好，符合设计的预期。

本次设计主要的工作是游戏系统的设计和实现，其中机器人行为树是本次设计工作的难点。在完成预计设计的基础上，结合开发过程遇到的问题，探讨了进一步完善游戏的思路和开发方向。

**关键词：**Unity3d，射击，游戏，引擎

**Abstract**

With the development of smart phones, mobile games are often sought after by people. However, due to the size and running speed of mobile phones, mobile phones cannot meet the needs of large-scale games. In the process of the game, players pay attention to the combination of man and machine, so this paper is to design a flight shooting game that can realize man-machine play and multi-person online connection. Firstly, through the overall planning of the game requirements, the modular content is gradually realized through planning. Then the simulation is designed according to the content, and finally the corresponding functional verification is carried out.

On the basis of requirement analysis and API function of Unity3d engine's powerful engine, this design constructs the overall framework of 3D scene multiplayer online shooting game software, which includes physical engine, UI engine, rendering engine and network engine. This paper carefully expounds the connection between different modules in different ways, and learns the data implementation required by the game software. Through U3D as the core engine of the game, with the basic support of C# language and other related programming languages, the functional requirements of shooting games are realized. The running status, system compatibility and later maintenance of the relevant software are tested, and the software is tested for real people. The results of the test run and the overall operation of the game show that the flight shooting game designed in this topic can basically meet the needs of users for the game. The user experience is good and meets the design expectations.

The main work is to design and implement the game system, in which the robot behavior tree is the difficulty point. On the basis of completing the expected design, combined with the problems encountered in the development process, this paper discusses the ideas and development direction of further improving the game.

**Keywords**: Unity3d, Shooting, games, engines

目录

[第1章 引言 1](#_Toc105518880)

[1.1 课题背景 1](#_Toc105518881)

[1.2 研究内容 4](#_Toc105518882)

[1.3 国内外研究现状 4](#_Toc105518883)

[1.3.1 国外研究现状 4](#_Toc105518884)

[1.3.2 国内研究现状 4](#_Toc105518885)

[1.4 主要内容和工作安排 7](#_Toc105518886)

[1.5 本文组织结构 8](#_Toc105518887)

[第2章 软件需求分析 9](#_Toc105518888)

[2.1 主要游戏开发引擎介绍 9](#_Toc105518889)

[2.1.1 Unity3D开发界面 10](#_Toc105518890)

[2.2 Unity3D开发框架 13](#_Toc105518891)

[2.3 开发插件 14](#_Toc105518892)

[2.4 系统可行性分析 16](#_Toc105518893)

[2.4.1 系统引擎的可行性 16](#_Toc105518894)

[2.4.2 开发流程的可行性 16](#_Toc105518895)

[2.5 需求分析 17](#_Toc105518896)

[2.5.1 系统功能分析 18](#_Toc105518897)

[2.5.2 系统非功能分析 19](#_Toc105518898)

[2.6 本章小结 19](#_Toc105518899)

[第3章 系统总体设计 21](#_Toc105518900)

[3.1 总体设计 21](#_Toc105518901)

[3.2 系统模块 22](#_Toc105518902)

[3.3 流程概述 23](#_Toc105518903)

[3.4 游戏策划 24](#_Toc105518904)

[3.5 游戏美工 25](#_Toc105518905)

[3.6 游戏性概述 25](#_Toc105518906)

[3.7 总体设计原则 26](#_Toc105518907)

[3.8 具体实施方案 27](#_Toc105518908)

[3.9 重点功能实现概览 27](#_Toc105518909)

[3.10 数据总体设计 31](#_Toc105518910)

[第4章 系统详细设计 33](#_Toc105518911)

[4.1 游戏具体功能 33](#_Toc105518912)

[4.1.1 按键控制 33](#_Toc105518913)

[4.1.2 Gaia地形生成 38](#_Toc105518914)

[4.1.3 多人在线游戏引擎 39](#_Toc105518915)

[4.1.4 雷达功能 42](#_Toc105518916)

[4.1.5 血条 42](#_Toc105518917)

[4.1.6 计分板 43](#_Toc105518918)

[4.1.7 机器人行为树 43](#_Toc105518919)

[4.1.8 后处理效果 44](#_Toc105518920)

[4.2 游戏运行实况 46](#_Toc105518921)

[4.3 本章小结 49](#_Toc105518922)

[第5章 游戏测试 51](#_Toc105518923)

[5.1 测试方法 51](#_Toc105518924)

[5.2 测试流程 52](#_Toc105518925)

[5.3 本游戏项目的测试 54](#_Toc105518926)

[5.4 结果分析 55](#_Toc105518927)

[5.5 游戏可移植性说明 55](#_Toc105518928)

[5.6 本章小结 55](#_Toc105518929)

[第6章 总结与展望 57](#_Toc105518930)

[6.1 总结 57](#_Toc105518931)

[6.2 展望 58](#_Toc105518932)

[参考文献 59](#_Toc105518933)

[致谢 62](#_Toc105518934)

[附录 英文翻译 63](#_Toc105518935)

[一. 英文原文 63](#_Toc105518936)

[Abstract 63](#_Toc105518937)

[I. UNITY3D GAME ENGINE 63](#_Toc105518938)

[II. UNITY3D GAME ENGINE INTERNAL ANALYSIS 64](#_Toc105518939)

[III. THE RESEARCH ON KEY TECHNOLOGIES 68](#_Toc105518940)

[REFERENCES 72](#_Toc105518941)

[二. 英文翻译 73](#_Toc105518942)

[基于Unity3D游戏引擎的关键技术研究 73](#_Toc105518943)

[摘要 73](#_Toc105518944)

[第1章 UNITY3D游戏引擎 73](#_Toc105518945)

[第2章 UNITY3D游戏引擎内部分析 74](#_Toc105518946)

[2.1组件模型 74](#_Toc105518947)

[2.2事件驱动编程 76](#_Toc105518948)

[2.3 类的总结 78](#_Toc105518949)

[第3章 关键技术的研究 78](#_Toc105518950)

[3.1一种随机生成npc的动态算法 78](#_Toc105518951)

[3.2 一种射击算法 80](#_Toc105518952)

[参考文献 81](#_Toc105518953)

# 

# 引言

伴随通信技术和网路技术的逐步革新，电脑端的游戏十分受人喜欢，因此每年电脑端的游戏充分占据了整个游戏大部分。各类电脑游戏：单机游戏、网络游戏、棋牌游戏、养成游戏等等。由于游戏引擎在操作系统上的差异，例如: Unity3D和Unreal。这也带来游戏开发上的问题，不同游戏类型所需要的平台也不尽相同。但是，根据游戏需求开发是核心问题，对于庞大的游戏数据市场，每年需要的更新和维护成本相对较大。我们需要的是对于后台游戏设计方能够充分考虑，通过外加引擎的方式实现有效的节约游戏端维护数据值，真正做到通过软件开发，并且达到多平台兼容的效果。U3D作为一款能够在各平台互相交流传输数据的应用级引擎，能够简洁的实现平台登录页面，以及平台内存在的数据解压，利用该引擎能够实现多平台数据兼容，无论是安卓、苹果、Window、Linux等系统，还是各类游戏终端平台都能够有效的兼容。对于快速革新的经济社会交流模式，包括各类网络平台目前最流行的交流方式，就是实现网上交流。一般大家都会用微博、QQ、微信这类常用的交通沟通软件，但是如果能够实现通过游戏内部交通沟流，那么就能让用户在享受游戏快乐的同时，又能实现沟通交流，这样极大的增加了游戏市场的需求性。因此本论文设计基于U3D引擎现飞行类射击游戏的开发，结合了游戏内部沟通交流方式，实现论题的需求，对引擎模块大力投入研究和深入分析，通过游戏设计实现对引擎的开发，拿到击杀得分，最后就可以获得相应的积分排名。

## 课题背景

游戏可分为单机、网游、手游、端游、网页游几大类，每种游戏都有不同的特色，游戏最需要的就是具有耐玩性，而不是靠数量取胜。当前游戏市场里面，可以说已经存在的游戏占据了大部分现有游戏市场，单从用户满足而言，很多用户已经满足现有的游戏，游戏的扩展面并不是很大，无法实现一款游戏能够受到全世界人们的追捧。随着科学技术的发展和经济全球化影响，国内经常玩游戏的资深玩家，已经无法满足于当前游戏的现状。他们一方面希望通过一款优质的游戏来扩展自己的游戏兴趣，另一方面也希望这个游戏拥有长足的发展空间和注入更多游戏模式，来体现游戏的耐玩性。当前国内游戏需求市场相对稳定，例如《黑神话·悟空》就是一款备受人们欢迎的独立性开发游戏，能够有效地融入中国传统文化里面的英雄元素，能够实现既利于文化传播，又能实现游戏娱乐的需求。这些年发展出来的游戏，很多成为人们口中经久不衰的话题，例如早些年的魔域、传奇、征途，再到后来的梦幻西游、qq飞车、穿越火线，最后到火爆的英雄联盟、绝地求生、守望先锋这类游戏的发展，无不体现出游戏市场正在逐步趋于完善，利用强有力的引擎开发程序，让国内游戏市场得到提升。在前几年的调查报告中，中国游戏市场每年游戏增长比例都在45%上下浮动，从此可以看出国内游戏市场的潜力之大，具体的国内游戏市场发展的趋势如图1.1所示。

图 1.1 2015-2017年中国游戏市场规模发展状况

U3D游戏引擎软件能够实现对常见的小型手机端游戏进行开发，灵活的数据库和相对简洁的戏设计板块，能够加速游戏的开发进程。对U3D而言，作为游戏开发引擎时代的主要首选引擎，它承担着未来引擎开发的重要地位，能够实现多个平台的数据交流。因此，致力于以它作为游戏引擎来实现对游戏市场的游戏发展是非常值得期待的。在整个平台开发游戏的过程当中，能够使开发工作量减少，并且游戏后续维护和更新的投入成本较低，生产商能够集中资产去开发游戏设计板块。对于U3D这个跨平台的游戏交流引擎核心，从平台交流角度而言，并不是非常完善的引擎。但是如果后续研究工作人员对其进行深入优化和研究，这引擎还是会逐渐被完善。数据化市场的逐步发展，能有效带动引擎的跨越式提升，本论文提到的引擎在以往的王者荣耀、和平精英等游戏中都有体现。

中国独立的游戏市场相对需求量巨大，根据国家相关单位调查显示，截止19年为止，用户每天玩游戏超过一个小时以上的将近4/5。而调查全国的游戏在线时间，可以得出在移动端使用游戏的用户数量也很多。国内互联网相关研究调查中心给出的数据显示，在过去十几年里面，电脑端和移动端游戏使用频率呈现逐年上升趋势。如图1.2所示。

图 1.2 2015-2019 年中国游戏市场规模

上图数据表明在国内的游戏需求市场较大，很多带着创新想法的上市公司，都想着争先来到游戏开发领域，实现游戏开发的首要考虑因素就是引擎，因此选择良好的引擎开发工具是很有必要性的。

## 研究内容

本节主要是基于Unity3d引擎的飞行射击游戏系统开发，主要是以U3D为引擎基础，根据对应的飞行射击类游戏基本需求展开分析，然后进行详细的研究思路规划和研究所需软件分析，最后进行相应的软件功能仿真和飞行射击游戏的实际测试来验证达到论文预期效果。

## 国内外研究现状

* + 1. 国外研究现状

当前，在国内外的跨平台游戏领域，Unity3D游戏引擎是各个开发商关注的焦点。原因之一是因为此款引擎的游戏设计能力十分强，游戏画面的效果一点不逊色于老牌游戏引擎效果；另外一个很重要的原因就是它很强大的跨平台应用的功能，它能够实现Android、iOS、PC，XBOX360、PS3的无缝并存。到现在，国内外的跨平台游戏在不到一年时间内已经相继出现了数十款的基于Unity3D的PC端网游，基于 Unity3D这款游戏引擎的手机端游戏更是有数百款之多。

据不完全统计，国外比较早应用Unity3D引擎的Bigpoint Game公司，到目前为止已经相继推出了《龙歌 OL》、《木乃伊 OL》等游戏作品，市场反响很不错。在国外，也有很多游戏厂商应用Unity3D这款游戏引擎开发游戏，美国的Playnet 公司到现在为止已经推出了比较受关注的3D画面的军事网游《激战海陆空》。另外，位于俄国的Nival公司目前也正在研发基于Unity3D这款游戏引擎的新型的Dota类塔防游戏《Prine world》。欧美有一家公司基于Unity3D这款游戏引擎研发的3D魔幻类的MMORPG的网络游戏，也是备受关注。九众互动公司的CEO朱传靖曾经针对Unity3D这款游戏引擎的研发技术提到过，Unity3D这款游戏引擎基于一些特定的技术架构，实现了十分实用的实时生成查看的功能，更加易用，从而简化了开发者开发游戏的流程，使得开发效率大大提升。

* + 1. 国内研究现状

国内通过Unity3D这款引擎研发游戏的进度比较早。例如，我们国内第一款基于这款游戏引擎而研发的网络游戏《将魂》，自2009年初就开始研发，目前已经开始对外测试。Unity3D游戏引擎是由Unity Technologies公司所开发的一个可以让玩家非常轻松的创建例如三维游戏、可视化建筑、实时的三维动画等互动内容的支持多平台的游戏开发工具，是一个十分全面的游戏引擎。Unity3D是一款利用交互图型化开发为首要开发式的游戏引擎。它的编辑器能够运行在Windows下和 Mac OS下，可以将游戏发布到多个平台下，基本涵盖了目前最流行的所有平台。是一款功能十分强大的游戏引擎。

2018年，杨鹏在《基于Unity3D的MmoRPG手机游戏研究》研究了主流3D手机游戏开发技术,针对3D客户端游戏研发引擎作出对比和分析,在Unity3D引擎的基础上给出了完整的手机网络游戏研发流程和方案。然后结合Unity3D自身的特点,给出了客户端游戏开发的整体架构,在架构基础上结合游戏策划的需求,针对客户端的各个功能模块开发进行设计,着重解决开发过程中遇到的难点。同年郭东方在《基于Unity3D坦克战争游戏的设计与实现》,对3D坦克射击游戏进行需求分析,并将本课题拆分为UI系统、战斗系统、成就系统和商城系统等四部分。通过Unity3D引擎实现了游戏场景,使用C#语言编写代码实现了游戏的逻辑处理。采用遮挡剔除技术剔除被其他物体遮挡住而相对当前相机不可见的物体,减少了渲染量。同时通过LOD算法逐层次简化物体的表面细节,减少场景的几何复杂性,提高了绘制算法的效率。此外还使用批处理、图集和对象池等技术,提高了游戏帧率,优化了游戏的性能。

2019年4月，马力在《基于Unity3D旅行故事游戏软件设计与实现》以Unity3D游戏引擎为核心,C#作为编程语言,SQL Server作为数据管理平台,利用VS作为开发工具,开发了一款旅行故事游戏软件原型,实现了角色旅行游戏玩法、植物种植玩法、精灵孵化玩法、玩家登录、玩家充值等功能。同年6月，徐文芳在《基于Unity3D教育游戏的设计与开发 ——以一年级《科学》课程为例》对国内外大型虚拟教育游戏及虚拟学习环境相关项目“Games-to-Teach”项目、“Teaching with Game”项目、Second Life进行案例分析,总结游戏化学习环境中知识内容、故事情节、游戏场景、任务模块整合的设计思路,并以认知发展理论、建构主义学习理论、沉浸理论、体验学习理论为中心指导理论,提出一年级科学课程的虚拟学习游戏的设计原则和策略。然后以该原则和策略画出游戏设计流程图和游戏开发流程图,再进行虚拟游戏环境的设计与实现,并对游戏进行测试和运行及实验。

2020年，朱杰在《基于Unity3D游戏人工智能的研究与应用》中基于Unity3D引擎设计制作一款射击游戏,包括游戏的故事背景,游戏风格,基础的可玩性功能,最重要的是,将本文提出的行为树感知系统设计方法和基于机器学习的设计方法应用于该游戏的NPCs制作过程中。同时对比实验多种训练方式,从数据中分析最优的训练方法。在篮球运动员游戏人工智能的设计制作中,提出了行为树与机器学习相结合方法,将机器学习所得到的策略模型封装成行为树中的某些节点,将二者有机的结合,取长补短。

2021年，唐迪在《基于Unity3D引擎的第一人称射击游戏设计与实现》基于网络使用Unity3D工具开发一套第一人称射击游戏,同时提供功能比较完备的游戏后台管理平台。系统在开发过程中引入ISO2000信息技术服务管理体系,使用UML统一建模语言对系统各个阶段的设计进行描述,强化项目组成员之间对项目的理解。系统采用面向服务的设计思想,在游戏客户端使用C#语言编写脚本完成动作和控制，引入有限状态机、寻路系统、战斗系统等提升游戏的体验。

但是国内在独立游戏开发领域发展并不算好。主要原因在于国内的游戏开发无法得到长期坚持，无法持续更新和吸引游戏玩家，导致游戏无法实现长久发展。更多的游戏制作商，把主要精力投放到如何让游戏玩家充值大量的金额来购买一些相对虚幻的游戏道具，没有从游戏真正的娱乐性趣角度来丰富游戏的内涵。很多具有创意的公司，已经迈出脚步去实现独立游戏的开发，但是受限于当前市场因素，无法确保开发出来的游戏能够投入正常使用，无法降低游戏的基本数据需求。并且对于游戏的开发整个流程而言，不是单人就能完成的，我们需要一个合理的游戏开发团队，从游戏数据的调研和市场分析，到游戏引擎选取，以及游戏各类平台数据融合，包括后期用户在游戏里面感受到的游戏界面以及游戏背景。整个游戏开发过程包括因素较多，因此必须要形成一个能够长期坚持的游戏开发团队，这样才能使独立游戏有意义的长久发展。国内当前缺乏一款独立游戏，一款能够打破国内当前游戏市场平衡的设计，之前仍在设计制作中的黑神话悟空游戏，就是在这个大背景下诞生的产物，如图1.3所示。

图 1.3 黑神话悟空在bilibili视频网站的关注度

## 主要内容和工作安排

论文主要在结合U3D游戏引擎开发软件的基础上，通过实现游戏存在版本的操作灵活优化，来结合游戏多个版本的沟通交流设计。鉴于之前存在的2D游戏模式，如早期的2D飞机大战游戏[1]。本论文选取的游戏开发引擎是U3D，势必将游戏带入一个空间领域状态，能够有效的实现在三维空间的游戏运行。通过玩家自身的游戏操作水平和射击类游戏基础，来实现玩家获得游戏里高分数据的需求。本游戏设计，主要是基于灵活操控战机的基础上，对敌机实施精准打击，这样的操作模式，能够让玩家兼顾左右手的协调，仿佛置身于真正的作战环境。单从飞行角度而言，和传统的飞机赛车相似；单从射击游戏角度而言，和传统的枪战游戏相似，因此对于玩家而言，这款游戏并不难上手。因此对本游戏的设计，充分调研考虑了用户的实际需求，实现用户灵活控制的需求和游戏的耐玩性需求。

## 本文组织结构

论文的内容包括六个章节，其主要内容为：第一章，引言相关的内容。主要介绍了Unity3D游戏引擎相关方面的背景、研究内容的必要性、国内外相关内容研究现状和本论文的主要结构安排；第二章，软件需求分析。主要讲解了用于Unity3D引擎开发的相关软件知识和界面介绍、该引擎的开发框架、系统的可行性分析、系统功能需求分析的相关内容说明；第三章，系统总体设计。针对游戏内容的总体设计、游戏系统模块游戏策划和美工、游戏的总体性设计思路和方案；第四章，系统详细介绍。介绍游戏的具体功能，包括按键控制、Gaia地形生成、多人在线对战、雷达功能、血条、计分板、游戏相关的运行实况分析；第五章，游戏测试。介绍了基于该引擎设计的飞行类设计游戏适合的测试方法、测试流程、项目测试、结果分析、游戏的可移植性说明；第六章，总结和展望。总结了论文所做的工作，以及展望了该游戏引擎设计游戏存在的不足点和后续该论题进行研究的理想方向。

# 软件需求分析

## 主要游戏开发引擎介绍

对于U3D游戏引擎而言，随着科学技术的发展和相对计算机科技能力的提升，对于游戏玩家群体而言，他们更注重游戏的创新点和整个游戏的耐玩性。如何针对游戏玩家的需求，来实现游戏引擎的结合变换设计游戏就显得十分重要。对于拥有巨大资本投入研究游戏引擎的公司而言，能够实现其对自己引擎的优化。对于资金相对薄弱的公司来说，要完成一款属于自己的团队创作的引擎几乎是不可实现的目标。在这样的大背景之下，U3D游戏引擎就产生了，游戏开发团队能够根据当前系统的数据，来实现高效的设计游戏的目的，因此能够把更多的精力集中到游戏耐玩性研究上。



图2.1 Unity 3D

Unity3D引擎是由Unity3D Technologies制作商专门为了实现游戏需求而开发的引擎，因此该引擎的精力主要集中在游戏开发的需求上，同时也兼顾了结构、美学、以及后期动画创作功能。随着该引擎的使用成熟化，从最初游戏引擎只能在mac系统上实现业务，到如今兼顾各种手机和电脑端操作平台，因此该游戏引擎最大的特点就是能够兼容各个设备平台。在通过对该游戏引擎基础上，实现了大量的游戏开发，从而增强丰富了对游戏引擎的资源。而这些丰富的游戏资源库，也促进了该游戏引擎得到各大平台的认可。游戏引擎能够结合当前平台提供的画面参数和相关游戏的程序文件，从生产成本而言，降低了投入生产的资金，并且游戏开发程序具备可移植性，因此即使存在开发问题，也能轻松在网上寻求帮助解决。

## Unity3D开发界面

游戏开发界面进行了高度的集成化，这样的设计方式能够有效的节约游戏设计过程的工作时间，并且U3D游戏引擎初始界面能够提供给游戏制作人员各种各样的人性化设计的功能，可根据游戏制作人不同的需求和设计理念进行开发。具体开发界面见图2.2所示。

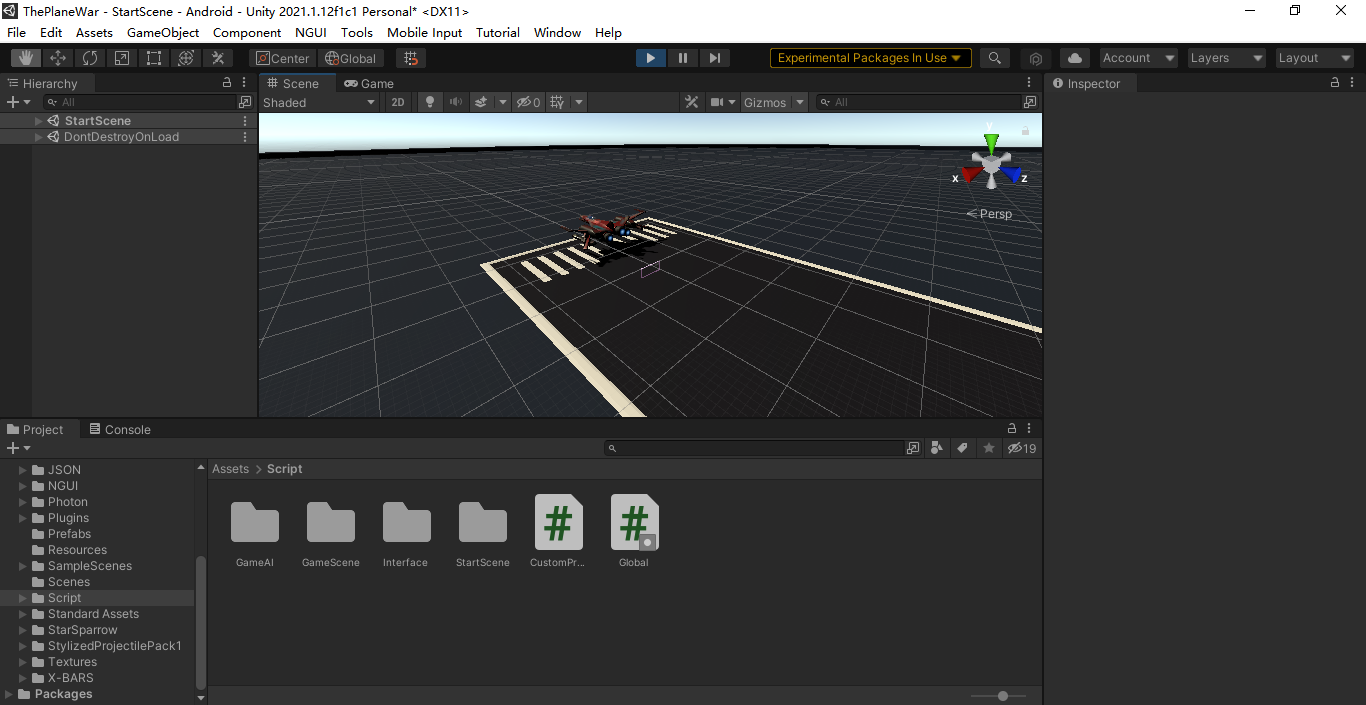


图2.2 Unity3D引擎的开发界面

Unity3D引擎的开发界面包括菜单栏、Hierarchy面板、Project面板、Scene面板、Game View面板和Inspector面板[2]。

1.菜单栏：包含了Unity3D引擎的所有核心主体游戏功能，能够对游戏整个过程存在的需求数值进行操作[3]。例如对游戏工程数据的打开、保存相关数据、输出制作完成的工程数据包。



图2.3 Unity3D引擎菜单栏

2.Hierarchy（层次）面板：可叫做具有层次的面板，能够实现在具体游戏设置场景当中对游戏构建需求的操作。

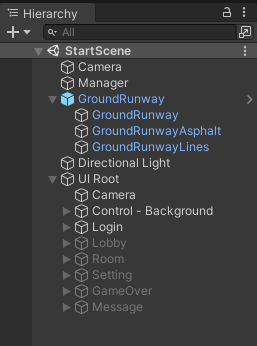


图2.4 Unity3D引擎Hierarchy面板

3.Scene（场景）面板：Scene面板是是对整个游戏内部场景进行构建的主要制作板，能够实现对各种工程和各种转换属性进行微调，站在玩家视角进行画面调制，也包括处理整个游戏场景存在的音效工程控制。

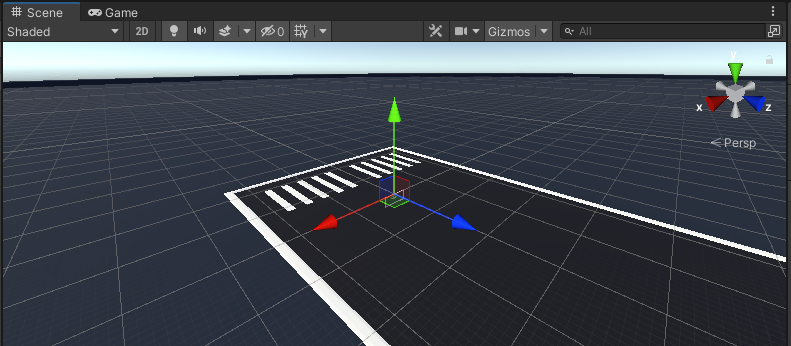


图2.5 Unity3D Scene面板

4.Project（项目）面板：Unity3D引擎的核心控制程序面板，拥有对所有工程程序的控制作用，存放大量利于工程实现的脚本，实现游戏界面比例放缩、可移植化需求。

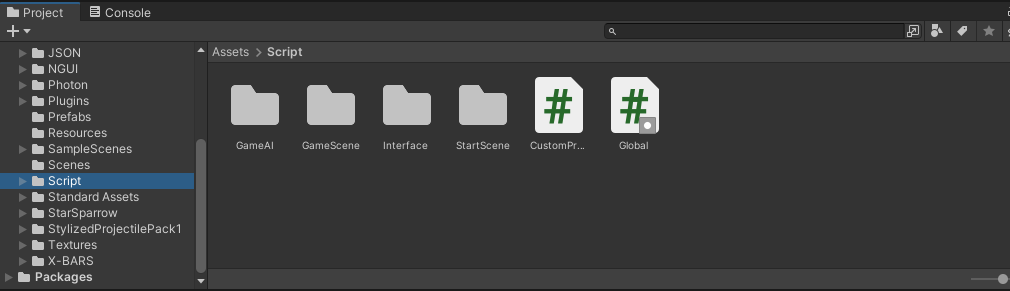


图2.6 Unity3D引擎Project面板

5.Inspector（检视）面板：能够用来体现游戏工程存在的各类数据组件面板，游戏工程项目的存在的逻辑操作都基于此面板实现，所以对于U3D游戏开发过程，需要利用各类游戏组件特征来实现游戏工程的开发需求。

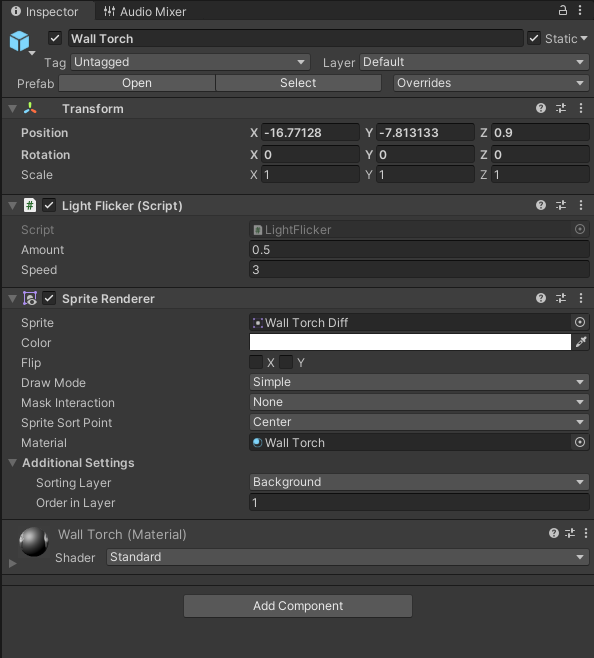


图2.7 Unity3D引擎Inspector面板

6.GameView（游戏视图）面板：通常情况下，是使用游戏内部场景捕捉板块来实现，对用户进去游戏界面感受到的游戏画面提取，真实的模拟游戏场景在游戏工程设计当中的视觉体现。



图2.8 Unity3D GameView面板

## Unity3D开发框架

U3D游戏开发引擎，在结构上可进行多系统划分和组合框架元素划分，通过游戏脚本设计、游戏内部语言开发、以及游戏各类资源和动画音效等，实现游戏开发过程存在的效率需求。游戏开发平台将会实现在基于平台框架的基础上，提升游戏对场景脚本处理数据的能力。本游戏引擎设计需要对其独立封装，并且存在功能操作系统。为了实现开发过程游戏内部场景的完美呈现，在预设游戏操作过程时必须进行模拟化处理，这样才能有效降低后期游戏维护和更新的成本。

Game Object（游戏对象）：在游戏界面内所有存在的物体，都是基于游戏项目内部通过对数据库进行设计，将场景元素灵活排布，实现终端控制面板统一操控元素的分配，最后完成合理的游戏项目实现对游戏对象化处理的操作。

Component（组件或脚本）：对于提到的游戏工程而言，内部核心单元主要是针对游戏内部存在的数据库。在U3D游戏引擎在游戏开发过程中，许多游戏设计人员利用事先已完成的功能封装函数模块，在过去十几年的函数库丰富基础下，能够让游戏开发人员轻松的完成基础需求设计，把更多的精力投放在游戏核心单元，即对游戏可持续性和耐玩性进行集中研究。

Scene（场景）：总的而言就是游戏界面存在的场景，对于一个飞行射击类游戏来说，无非是各个场景的有机组合，利用不同场景的各自的风格特点，通常可以使用网络数据库提供好的场景数据。场景作为游戏项目设计的一个元素，玩家通过对游戏场景的具体需求，来使得游戏开发人员将不同游戏项目需求的库进行完善。在单个游戏场景里面，基于游戏项目库数量的不同，实现远程对数据项目库的渲染，这样的操作最终实现的游戏场景能给玩家一种真实感，也相对节约了开发程序的时间投入。

## 开发插件

Unity3D引擎的主流UI经历过三个时期：



图2.9 Unity3D引擎UI发展

最早的OnGUI因为部分功能相对落后，无法满足当下游戏设计的一些需求，因此在实际开发应用当中，研究人员能够通过UI程序开发基础，实现程序模块的插件投入，能够将实现完成的UI板块界面呈现在用户视野面前。能够有效节约之前对于UI开发所需要的相对可视化操作的抽离需求，很好地提高了此类开发方式的效率。

Unity4.6版本，UGUI被集成到Unity3D引擎[4]，这是由Unity3D平台商为了游戏制作者而设计的UI插件，设计的目标是为了让设计者感到使用轻松、视觉效果强、运算速度快。对于游戏开发人员来说：实现游戏的拓展性好、游戏运行效率好、操作上手难度低以及和U3D引擎在系统设计的兼容良好需求。这款独立的UI设计系统，设计以来就能够实现高速的运行效率，投入市场使用之后，一致受到开发者工作人员的青睐。

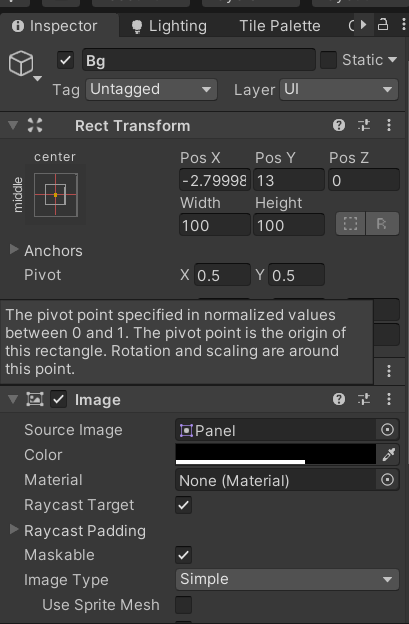


图2.10 UGUI Inspector面板

针对论文需要的U3D游戏引擎，现在进行主要功能介绍，对于核心框架体系和游戏需求的核心引擎技术，包括该引擎在目前国内外研究状况，进行基于开发需求的探究。

## 系统可行性分析

### 系统引擎的可行性

对于u3d游戏引擎而言，基于该引擎开发制作的游戏已经在市场中投入使用，并且占据了游戏市场，游戏界面显示效果良好。像传统的类似神庙逃亡的这种游戏，包括很多小游戏，都有内在需求以及多平台融合的设计理念。因此对于游戏开发必须充分考虑游戏后期的兼容性，对于小游戏而言，更多是在电脑端使用，因此可以优先考虑电脑端平台对游戏开发引擎的需求。结合电脑常用的英特处理器来实现游戏运行设计需求。为了方便游戏设置需求，通常选取Window7或者Linux操作系统来实现，老版本的操作系统更有利于结合U3D集成开发软件系统。当下对游戏软件的开发，结合游戏整体开发的功能需求，利用工作人员的技术水平，在当下成熟的游戏大环境下，针对U3D游戏引擎进行优化处理。便于后期在对游戏整个维护工作中，对游戏引擎板块进行优化即可。当下难题是突破多平台融合游戏引擎使用，这样才利于游戏在后期维护中的需求，能够让游戏维护工作人员只针对一款系统进行优化，就能实现多个平台一同优化的效果。

### 开发流程的可行性

对于论题的软件开发需求，进行目标的可行性规划，从当前软件工作能力需求分析，在整个软件的开发流程中，一般采用迭代、V等模型，基于当前社会市场的需求，选择较为合适的开发模式。在这些模式基础上，会产生多种研发模式，主要是为了结合本论题的游戏引擎要求[5]。本论文使用的是 V开发模型，融合系统迭代的思想，让整个系统实现结构简单，用户体验感好。创造好的用户体验过程，通过用户对游戏的反馈，实现软件开发。具体软件开发流程分析，见图2.11。

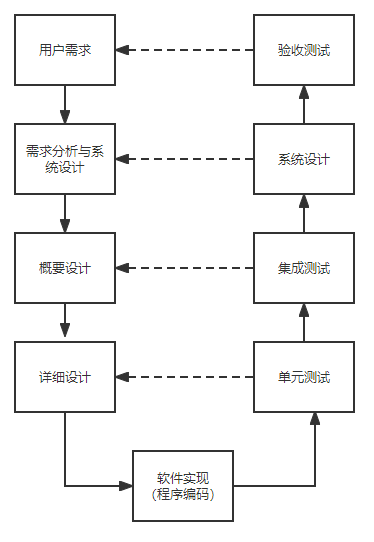


图2.11 开发流程

## 需求分析

对于游戏设计需求过程，设计游戏一般包括市场探究、游戏设计、游戏开发、游戏策划、游戏优化等过程，需要合理地规划游戏前、后期具体完成的工作。在游戏前期策划和后期游戏制作实现这两个方面，并没有绝对的时空区别，游戏前期的策划能够为后期游戏制作提供依据，同理，游戏的制作过程情况也能够反馈到游戏前期策划[6]。只有在两者结合之下，才能够完成对游戏整体设计需求。对游戏的前期规划，必须明确游戏的设计目的，针对的用户群体需求和游戏后期发展方向进行统一规划，将游戏设计理念落实到后期游戏制作当中。游戏制作过程，会充分考虑前期的相关游戏策划内容，合理实现游戏功能需求。前期策划和游戏制作这两大游戏设计模块存在的相互关系，如图2.12。

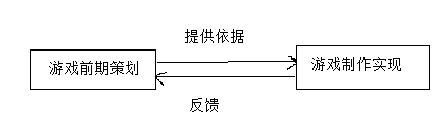


图2.12 游戏开发流程

### 系统功能分析

采用U3D作为游戏引擎实现游戏各类功能需求的开发，能够实现游戏需求相关的各类功能：

1. 控制战机移动：主要体现在玩家可以控制游戏内的战机飞行，完成游戏具体的操作需求。
2. 得分系统：主要用于记录游戏用户，在游戏过程当中通过游戏设定的任务或者击败敌机数量得分，最后显示用户具体分数。
3. 射击效果：主要是针对飞行射击游戏，在攻击系统发射导弹之后，命中敌机或者子弹爆炸之后游戏界面出现的特效。

4、生存和死亡状态：游戏玩家初始状态和重生状态都是操作流程，当玩家受到攻击或者任务失败时，系统进行颜色变换和文体提示，提示用户已失败，等待重生或者重新开始游戏。

5、UI互动：能够完成对游戏内部音效的控制，在结束游戏后，回到游戏初始界面的功能需求。

### 系统非功能分析

针对系统非功能模块的探究，主要是衡量系统模块能否可持续发展，根据相关市场需求的调研，对系统设计模块后期需求进行规划，系统的功能能否通过测试，以及软件存在的功能问题，衡量系统软件质量是否满足市场需求，后期系统更新是否满足市场变换情况，针对游戏维护工作提前做优化处理。

## 本章小结

本章节主要分析游戏整个开发过程，以及开发过程存在的市场需求探究。充分研究本论题设计系统是否存在市场拓展能力，以及系统研究涉及到的资金投入，以及当前软件开发市场的大环境。能否实现市场化对游戏功能和游戏界面的需求。另外对游戏非功能方面进行了探讨，针对游戏每个研究过程做出详细的技术研究，明确软件后期的更新路径，切实的保障游戏的质量。

# 系统总体设计

对于单个游戏项目的开发，一般包括游戏项目准备和具体游戏项目制作阶段[7]。本章节将重点介绍游戏制作阶段的主要工作[8]。以下将从游戏的项目模块开发功能需求进行详细阐述。

## 总体设计

本论文设计的游戏整体是基于U3d引擎进行设计的，整个游戏程序设计包括几大方面：核心程序组，主要用于控制玩家驾驶的飞机进行飞行和飞机发射炮弹；射击效果组，主要用来控制飞机发射炮弹的弹道效果和炮弹击中敌机产生的特效；数据存储，主要用来记录玩家的数据和游戏每局的得分情况；终端互动组，主要是用来联网、联机实现让玩家联机对战。整个游戏的系统设计，主要包括用户端、服务层、逻辑功能层、储存层部分，具体情况见图3.1所示：

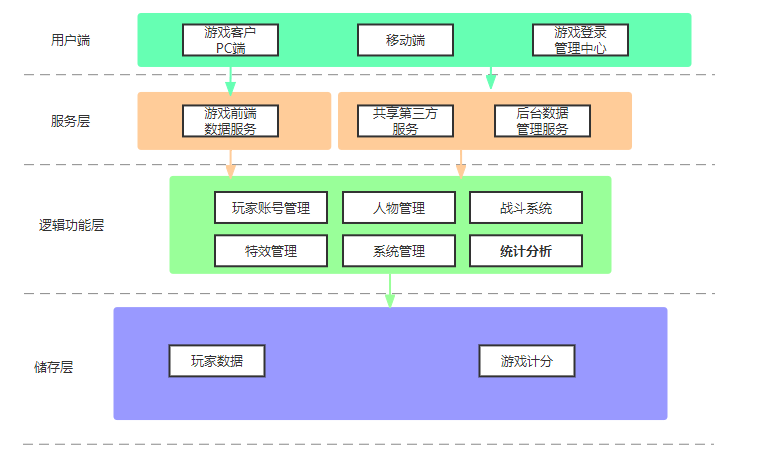


图3.1游戏系统设计图

## 系统模块

游戏大致应用了如下技术：

1. Unity3D游戏引擎
2. .NET Mono C# 游戏脚本
3. Linq 语法糖
4. 协程控制游戏进程
5. 队列控制文字消息提示
6. NGUI和UGUI两套UI界面
7. Gaia 自动生成地形
8. Photon Pun 2 多人在线游戏引擎
9. Photon Voice 2 多人在线游戏语音
10. 飞机飞行与子弹导弹的发射和跟踪物理引擎算法
11. 飞机模型与子弹导弹粒子系统
12. 雷达显示飞机位置
13. 自己和敌方飞机的血条显示和线性变化
14. 计分板杀敌死亡计算以及点击计分板可查看目标飞机第一视角
15. AI机器人行为树实现自动飞行和战斗
16. 手机摇杆和陀螺仪控制飞机偏航角与摄像机视角
17. JSON 保存游戏设置
18. 引擎自带水面材质与天空盒
19. PostProcessing 2 视觉效果强化
20. 基于 Android 手机的图形优化

游戏项目的总体功能如图3.2所示



图3.2游戏功能结构图

## 流程概述

在介绍游戏制作流程前，见图3.3所示的情况。

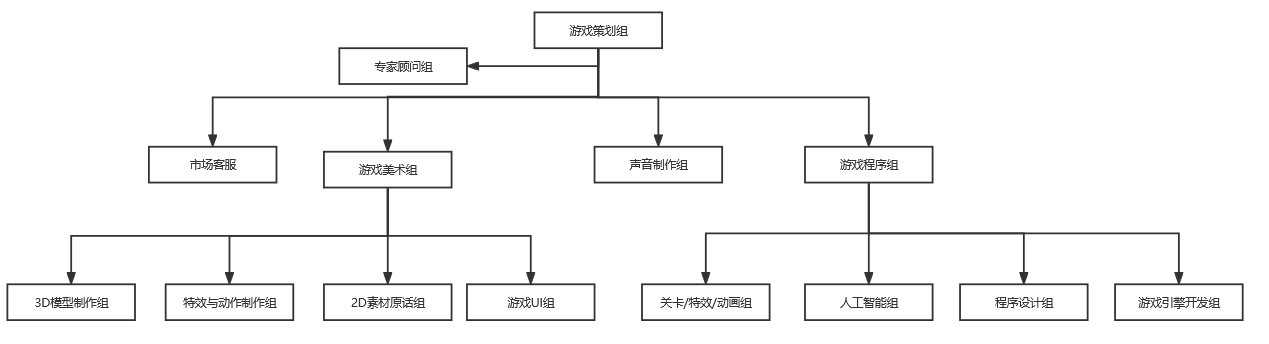


图 3.3游戏制作工作分工

游戏开发过程分工，合理规划游戏各模块需要做的工作。对于游戏主体工作框架可划分为，策划游戏总体需求，优化游戏场景界面，后期游戏更新维护程序[9]。对于游戏策划包括游戏总体设计架构，以及游戏核心内容设计，以及游戏采用的娱乐模式，这些都将成为影响游戏是否具备长期发展能力的因素。对于优化游戏场景界面，要在设计工作中优先考虑玩家在后期游戏过程中对游戏画面的需求，能够给游戏玩家足够丰富的游戏视野。最后，在游戏更新维护方面，能够结合游戏各开发模块进行优化，以减少后续游戏维护工作，利于游戏更新操作。

游戏开发的这三个主要流程，并不是连贯而成，应该独立考虑每个游戏阶段需要实现的效果，进行优先哪个模块规划，刻意划分游戏各模块的工作，将各游戏模块工作有机结合，最后实现游戏的功能，提高用户的游戏体验感。游戏的开发过程，是一个能够通过循序渐进来实现的游戏开发方式，这样才能得到更多玩家的认可，在游戏市场占据一定发展地位。

## 游戏策划

游戏的策划工作：主要包括游戏系统的选取、相关参数值、游戏背景文案这三个方面，对这三个面进行规划。对于游戏策划者，就是站在上帝的视野看待游戏过程，根据游戏的运行情况制定相关规则，策划者就是这个游戏世界的主要控制人[10]。

针对游戏文案方面的策划工作，要是结合当下社会发展背景，融入人们价值理念中的历史元素，给玩家一种亲切的感觉，更能吸引他们投身到游戏当中。作为游戏策划工作人员，其主要任务就是设计游戏的相关文案，利用文案来吸引用户对游戏产生浓厚兴趣。对游戏整个设计层面，合理的游戏策划利于游戏玩家感受到游戏的真实性。有着丰富经验的游戏策划人员，可以协调游戏系统内部各种策划需求，避免游戏偏离主题。

## 游戏美工

对于游戏研发系统来说，这是一个相对枯燥和机械的工作。游戏美工会对游戏画面进行美化，游戏美工也是后期游戏直接呈现在用户面前最直接的产物，要求游戏美工人员要有创新的思想。这通常需要结合日常生活中人们熟悉的场景，把场景灵活的运用到游戏里面，进行游戏结合优化，最后将场景组合，最终形成预期的游戏界面。  
 在游戏素材选取工作中，应该结合游戏设计系统对选取工作的具体安排要求，实现对样本材料的公平筛选，这有助于程序融合提升。对于游戏画面选材的效果，通常结合特殊的游戏特效元素，让设计策划的游戏美工进行可视化模型分析，最后运用相关软件进行建模处理，实现最后游戏产品能够具备高质量水平的美工处理效果。

## 游戏性概述

在日常生活当中游戏的基本就是应该满足用户日常娱乐需求，游戏必须要求达到操作简单、画面真实、带入感强这三点，而且要结合人们日常的价值理念和对美好事物的追求。对于喜欢游戏的玩家，大多数是为了放松自己，即游戏就是他们的娱乐方式之一，想通过游戏来释放心中压抑的情绪。所以也要满足这一部分玩家的需求。

1.交互性：对于游戏，其主要的功能是满足人们的娱乐需求，但是在娱乐之余，用户也很注重可以和其他玩家进行联网交流，这是有区别于单机游戏这种只重视游戏娱乐性而言的。游戏玩家虽然清楚游戏世界并非真实世界，但是更多的真人玩家在线，也能让游戏玩家感受到这就是一个真实的社会交际场所，提高游戏玩家的娱乐兴趣和社会交流能力。因此我也要充分考虑，如何合理的实现游戏内部交流沟通。交互性也体现在当游戏玩家进入游戏界面时，游戏系统预设的指引界面，引导玩家正确使用游戏类各种功能。我要充分考虑，对于已经玩过此游戏的用户，如何快捷引导他们进入游戏，避免出现复杂的游戏介绍而影响用户的体验，甚至让用户直接放弃玩这款游戏的情况。

2.规则性：俗话说得好，无规矩不成方圆。如果一个游戏没有合理的规则，那么对于游戏玩家而言是不公平的，对于游戏后期管理维护也是一个难题，因此可以将游戏策划内容提升到游戏的核心灵魂层面上，强调这是保证游戏正常运行的一个基础。通过制定相关游戏规则来约束玩家，防止出现游戏内部的不公平现象。这样就能像日常生活当中制定的相关法律法规，通过相关法律法规来规定人们的言谈举止。这样更能让游戏用户感受到 ，游戏世界和真实世界一样都是具备一定的规则，这样游戏也能传达出社会正能量，利于国家法治文化建设，只有合理考虑这些因素的游戏，才能够在游戏市场走得更远，才能得到更多的发展机会。

3.耐玩性：游戏的耐玩性主要体现在游戏设置的相关情节和游戏进行过程，从游戏关卡情节到游戏画面，以及游戏后续活动很多方面因素对游戏进行充分分析[11]。通过市场调研结果，来实现对游戏画面、游戏音乐等内容的合理性设计。如果游戏能够根据玩家的兴趣来开启不同的模式和玩法，那么这个游戏就能极大的增强其耐玩性，能吸引更多的游戏玩家参与这个游戏。一个游戏的成功，最具参考性的指标就是游戏用户在线数量和长期在线一款游戏的玩家数量。

## 总体设计原则

本次开发的游戏主要是对于拥有零散时间的游戏玩家和业余时间需求游戏娱乐的玩家，为了满足不同时间需求的人员，因此我在游戏设计层面，本着简洁易懂的原则，设计了离线模式和真人玩家匹配模式。能够有效的解决玩家在短时间内需要游戏娱乐而缺乏真人用户及时匹配的问题。在游戏相关内容上，游戏策划着重对游戏体现的世界价值观集中表现。结合了单机游戏和网络游戏的特征，极大地增强了游戏的耐玩性。

## 具体实施方案

角色设定：考虑到游戏规模是轻量级，注重的是操作简便性和易区分性，选定了红色、蓝色、绿色、黑色、橙色、紫色战机。

音效设定：游戏的背景音乐根据游戏进行情况设计，能够根据玩家在游戏中达到的实际操作情况来播放对应的节奏音乐和音效。

战斗设计：具体的游戏战斗设计，射击类飞行游戏需要考虑空间的射击路线和相对的射击特效。总体而言，飞行射击游戏能够让玩家在游戏当中，感受到各种游戏战斗的特效，从而满足内心驾驶战机的欲望，极大的增强了游戏的真实性。

联机系统设计：联网对战作为网游的灵魂，根据移动端的联网特点，我们采用Unity商城自带的Photon Pun2插件托管服务器实现同步和消息发送的联机模式，采取 2~6 人联机进行对战的形式。

## 重点功能实现概览

此项目的重点功能有三项，分别是多人在线大厅功能、飞机飞行功能和子弹导弹发射功能，以下作相应的简单阐述。

玩家打开游戏后，来到多人在线的大厅，进行游戏开始之前的准备界面，选择合适的房间进入游戏，然后选择本局游戏将要使用的战机和作战场景的地图，根据实际需求添加机器人参与本局游戏。

多人在线大厅的整个流程具体如图3.4所示。

多人在线大厅的用例图如图3.5所示。



图3.4 多人在线大厅模块流程图



图3.5 多人在线大厅模块用例图

点击开始按钮进入游戏，来到飞机飞行的状态，控制飞机飞行的是移动摇杆，利用输入轴的变化情况来改变飞机偏航角；输入轴不变的情况下，计算当前飞机的动力和扭矩，来同步当前飞机状态。根据下一步摇杆输入的指令重复上述操作，如无继续输入指令则程序结束退出到游戏初始界面。具体流程如图3.6所示。



图3.6 飞机飞行模块

然后来到游戏关键的环节，就是遇到敌机进行射击的环节。这必须兼顾控制飞机的方向和控制飞机子弹开关进行射击。遇到需要调整飞机的方向和偏角，才能准确的命中敌机时，需要调用远程伤害判定的程序来对敌机进行状态分析，以得到是否命中敌机的准确数据和是否需要继续射击的必要性 。如图3.7所示。



图3.7 飞机发射子弹导弹模块

## 数据总体设计

项目数据E-R图如图3.8所示。



图3.8 项目数据E-R图

项目数据关系代数如下：

1.房间（房间名，最大房间人数，主机玩家，游戏时间，飞机）

2.飞机（玩家名，模型，血量，目标点，物理参数，是否是机器人，子弹导弹，计分板）

3.子弹导弹（模型种类，模型大小，伤害）

4.计分板（玩家名，击杀，死亡）

# 系统详细设计

## 游戏具体功能

## 按键控制

在Unity3D中，要想实现飞机移动，我们必须知道欧拉角这个概念，如图4.1所示。

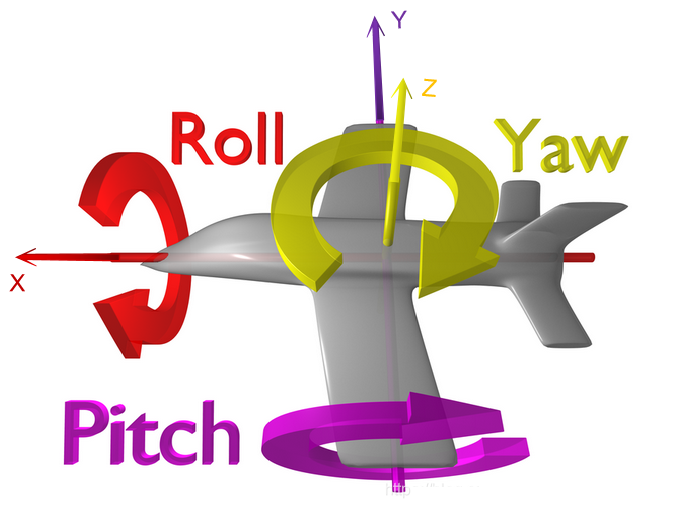


图4.1 欧拉角的图示

在飞机左右移动的时候，我们会旋转模型的Yaw和Roll轴，在飞机上下移动的时候，则会旋转模型的Pitch轴，然后再根据模型的向前向量，乘以一定的速度，飞机就可以灵活的在3D场景里自如的移动了。控制移动代码如图4.2所示。

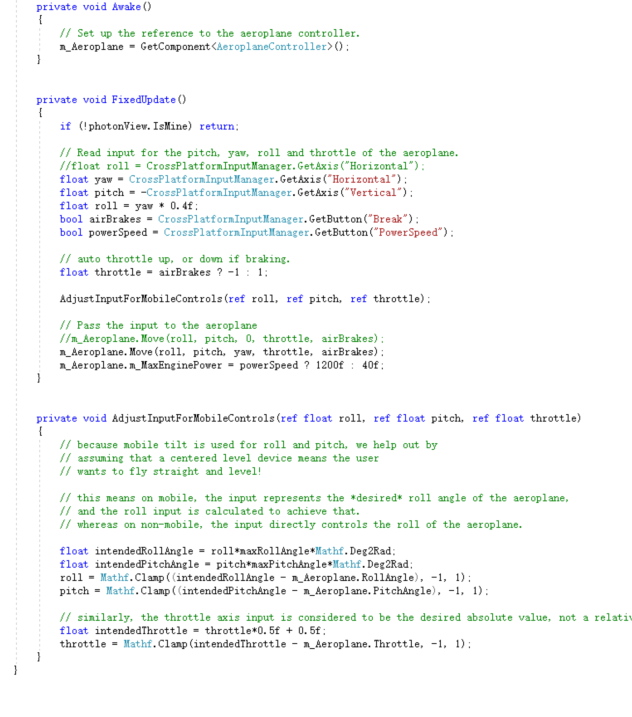


图4.2 控制移动代码

射击代码首先判断了角色现阶段所处的状态，并且使用了Input.GetButton（“Fire”）的代码接收了键盘传来的控制指令从而通过在飞机前方生成一个子弹的预制体再向前施加一个一定大小的力从而实现了射击，如图4.3所示。

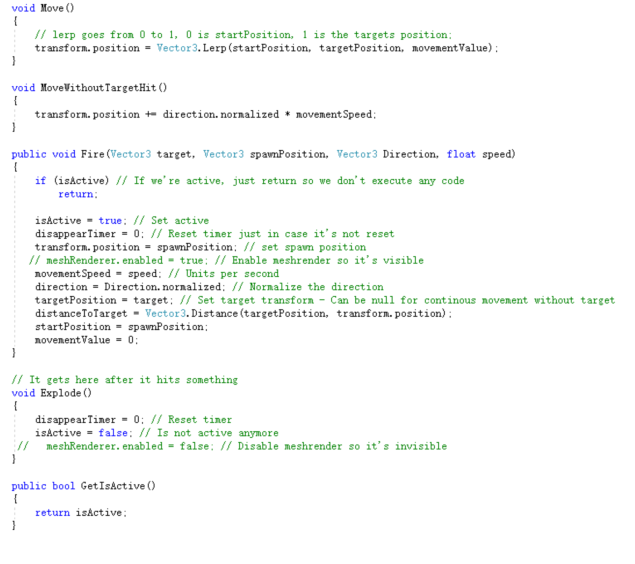


图4.3 子弹代码

导弹控制是通过一个算法计算飞机前方一定距离内是否存在另一个敌方飞机，再以一定的角度向敌方飞机偏转，导弹正确控制和发射操作部分相关代码，如图4.4和图4.5所示。



图4.4 导弹控制代码

文本

描述已自动生成

图4.5 计算前方敌方飞机代码

导弹控制判定如图4.6所示。



图4.6 导弹控制判定图

导弹控制判定树如图4.7所示。



图4.7 导弹控制判定树

## Gaia地形生成

在Gaia插件里，我们首先要对游戏环境进行设置，并创建相应的地形生成器对象，如图4.8所示。

图形用户界面, 文本

描述已自动生成

图4.8 Gaia插件初始化设置

然后我们根据地形示例的相关参数，就可以创建一个地形环境，如图4.9所示。

图片包含 地图

描述已自动生成

图4.9 根据地形示例创建地形环境

## 多人在线游戏引擎

多人在线对战功能使用了Photon Pun 2 插件进行制作，这个插件包含同步，消息发送、语音、房间大厅、离线模式等功能。

同步功能需要用到插件自带的Photon View组件，在玩家飞机模型上挂载此插件后，再挂载Photon Transform View（位置同步）、Photon Animator View（动画同步）、Photon Rigidbody View（物理同步）中的其中一个就可以实现相应功能的网络同步了，在此毕业设计中，我使用的是位置同步，如图4.10所示。

图形用户界面, 应用程序

描述已自动生成

图4.10 飞机网络同步组件

游戏的消息发送机制需要在相关的方法代码上方添加一个PunRPC的注解，然后调用photonView.RPC方法，第一个参数是方法名[12]，第二个参数是需要发送的玩家，可以是某一个玩家，也可以是所有玩家，这样就可以实现远程调用方法，从而实现多人联机对战的相关逻辑了。

带注解的远程播放声音的代码如图4.11所示。

图形用户界面, 文本, 应用程序

描述已自动生成

图4.11 远程播放声音代码

RPC调用的代码如图4.12所示。



图4.12 RPC调用代码

能运用到RPC消息发送的地方有很多，这里重点讲一下子弹攻击敌方飞机时实现的逻辑。

在子弹打中敌方飞机后，首先我就拿到一个5到15的随机值作为伤害值。然后拿到敌方飞机的当前血量和是否无敌的判断，如果敌机已死（血量为0）或者无敌，则不进行任何操作。否则造成伤害，在攻击方玩家显示已击中的准星特效，然后向敌方玩家发送一个播放被击中声音的消息。如果造成伤害后敌方飞机血量为0，则运行死亡的一系列代码，分别是：在攻击方玩家显示击杀的准星特效，播放击杀音效，计分板击杀数增加一个，然后向所有玩家发送一个公屏显示谁击杀了谁的文字的消息，最后向敌方玩家发送死亡的消息，敌方玩家接收到死亡消息后，开始处理死亡相关的代码，如爆炸等。具体代码如图4.13所示。



图4.13 子弹攻击敌方飞机代码

游戏里的语音用到了Photon Voice插件，在按下按钮时，开始录音，松开按钮时，结束录音，实现起来比较简单，这里略过。

房间大厅和离线模式的功能都封装在了PhotonNetwork类中，在代码中实现相关逻辑即可。

## 雷达功能

雷达功能是在玩家的飞机上方一定距离放置一个摄像机，然后在所有玩家的飞机模型中放置一个脚本，只要这个脚本放置的物体就可以在摄像机扫描到的地方显示一个红色的标志。雷达永远指向飞机的正前方，并且可以显示地形和水面，最终的效果如图4.14所示。

图片包含 室内, 桌子, 小, 玻璃

描述已自动生成

图4.14 雷达

## 血条

血条的线性变化是这样实现的：在血条中放置三个不同颜色的血条。最上面一层显示红色，代表当前的血条；中间层显示黄色，代表线性变化；最下层显示白色，代表背景。在受到伤害时，红色的血条立即变化，黄色的血条会通过时间慢慢减少，这是用Unity数学库中的Lerp函数实现的。死亡时的血条线性变化效果如图4.15所示。

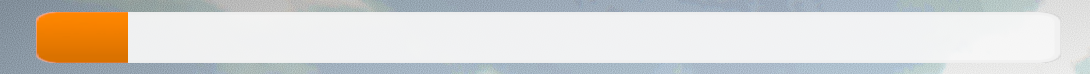


图4.15 死亡时血条线性变化

敌方血条显示是把3D位置转化为2D投影的方式实现的，运用了Unity库自带的WorldToScreenPoint函数，最终的效果如图4.16所示。

飞机飞在空中

低可信度描述已自动生成

图4.16 敌方血条显示

## 计分板

计分板保存了每个玩家，包括机器人的击杀、死亡的次数，点击计分版会切换到对应飞机的第一视角，最终效果如图4.17所示

切换第一视角的功能很简单，只需要关掉自身的摄像机，再打开对应飞机的摄像机，引擎就会帮我们自动切换了。

电脑萤幕画面

描述已自动生成

图4.17 计分板

## 机器人行为树

 机器人行为树是本毕业设计的重点和难点，这里对机器人的大致实现逻辑进行简单总结。如图4.18所示。

卫生间的摆设布局

低可信度描述已自动生成

图4.18 机器人行为树

1. 首先判断机器人是否由主机玩家控制，非主机玩家不会对机器人进行操作；
2. 机器人前十秒寻找随机目标点飞行，并开启无敌模式；
3. 后十秒首先判断机器人是否死亡，如果死亡则一直等待；
4. 然后判断前方一定距离是否有敌机，如果有则指定为目标点飞行并攻击，如果没有则飞向随机目标点；
5. 在飞向随机目标点时一直判断前方是否有飞机，有则定位为目标点，并开始射击和追踪，否则继续飞行；
6. 如果飞到特定的目标点后会随机指定另一个目标点继续开始飞行。

## 后处理效果

在场景内，我使用了Processing Post 2插件对环境进行优化，称为后处理效果，设置面板如图4.19所示。

图形用户界面, 应用程序

描述已自动生成

图4.19 后处理效果设置面板

在水面下方的后处理效果如图4.20所示：

飞机在空中

低可信度描述已自动生成

图4.20 水面下方的后处理效果

## 游戏运行实况

整个游戏的运行过程：从开始界面、游戏大厅界面、房间界面、设置界面、游戏界面、死亡界面、结束游戏界面。

开始界面包括3个选项：黄色的开始游戏按钮、灰白色的游戏设置和退出游戏按钮，详见图4.21。



图4.21 开始界面

大厅界面包括3个选项：黄色的创建房间按钮、随机加入设置和断开连接按钮。创建房间时可以输入房间门和最大限制人数，左边是房间列表，详见图4.22。

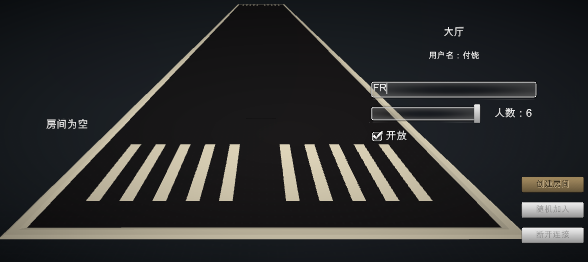


图4.22 大厅界面

房间界面包括3个选项：开始游戏、添加机器人和离开房间按钮，选择地图和是否开放房间勾选框。对其他玩家，可以设置为主机玩家，踢出房间。对机器人，主机玩家可以更换模型，删除机器人。还可以语音对话。详见图4.23。



图4.23 房间界面

设置界面包括几个选项：姓名版颜色的提示，选择子弹按钮、选择导弹按钮、离线模式按钮、启用陀螺仪按钮、确定按钮，详见图4.24。

[图形用户界面

描述已自动生成](https://github.com/664235822/ThePlaneWar/raw/master/img/psc%20(3).jpeg)

图4.24 设置界面

游戏界面包括：左侧血条从左到右依次是退出房间按钮、自杀按钮、结束游戏按钮，三个按钮的下方则是玩家的计分板，右上角是雷达，左下角是摇杆，右下角是具体的操作按钮：X按钮发射子弹，Y按钮发射导弹，B按钮减速，A按钮加速。详见图4.25。



图4.25 游戏界面

死亡后的界面视角切换到场景一方，开始自杀读条，上面提示阵亡飞机玩家信息，详见图4.26。

[图形用户界面, 网站, 日历

描述已自动生成](https://github.com/664235822/ThePlaneWar/raw/master/img/psc%20(6).jpeg)

图4.26 死亡界面

游戏结束界面：将会展示每个玩家（如机器玩家）游戏具体的得分情况、右下角会有一个确认按钮，详见图4.27。

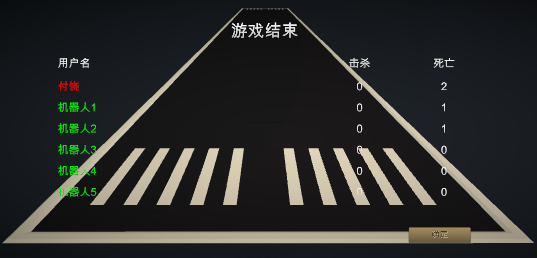


图4.27 游戏结束界面

## 本章小结

在常见的飞机大战项目中，我们见到的都是2D平面的小型项目，3D的飞机大战游戏，只有大型的3A游戏或者高质量的手游才有这样的体量。本项目不仅实现了3D飞机的飞行，还拥有多人在线游戏引擎，能实现最多6名玩家在不同设备上游玩的体验。本项目还提供离线模式和机器人行为树，特别是机器人行为树，是本项目的重点和难点，实现了这样的功能以后，单名玩家，也可以开心地游玩此游戏。项目运用了物理引擎，子弹和导弹的发射都会一定程度的向敌方飞机偏转，攻击敌机的难度就会降低，再加上雷达，加速、减速，都使玩家更加灵活地操作飞机和判断敌机目标。项目的UI界面和渲染都对手机进行了一定程度的优化，不仅支持摇杆操作，也支持陀螺仪。计分板能切换到敌机视角，前方敌机会自动显示生命条，而且血条有线性变化，攻击敌机时准星会偏移并播放音效，有良好的打击感。

总的来说，细节上的打磨和丰富功能的实现是保证此毕业设计质量的立足之本。

# 游戏测试

## 测试方法

在软件测试方面，对即将出产的成品游戏功能进行相关测试，以补全在设计层面存在的缺陷和游戏制作过程存在的不足。其目的是提高用户在游戏当中的体验，游戏测试内容包括对游戏主要架构实现游戏性能的具体参数测试，针对当下游戏架构存在的隐患进行深入分析。例如电脑CPU负荷过重、运行温度过高、存储数据爆满等一系列可能出现影响游戏正常运行的问题和影响游戏连接的外部的网络问题。先根据用户的需求，实现对需求与系统的分析，然后进行总体概要设计，基于总体概要设计的情况进行详细设计，实现对程序的编写[13]。基于程序编写完成并且优化处理之后，进行系统单元测试，以及系统的集成方面测试，对游戏的功能进行验收性总结测试[14]。测试结果当中，如果生产产品存在缺陷，将缺陷报告反馈到相关研发工作人员处，在经过相关人员的讨论纠正之后，共同拟定出新的整改方案。整个软件测试过程类似于迭代思想，通过不断的优化缺陷，来实现系统功能的完整。论文使用的就是结合功能测试和V模型思路的测试路径，具体测试过程[15]，见图5.1。

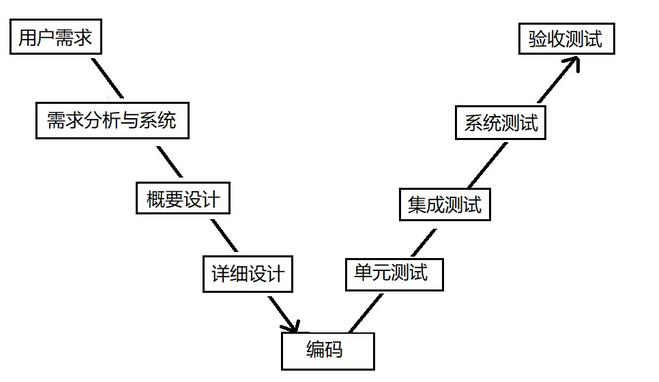


图5.1 V模型

我测试的三个主要方面包括：1、对游戏系统的界面功能测试 2、对游戏系统的声音功能测试 3、对游戏系统的逻辑测试[16]。基于每一项测试数据分析，然后总结出该测试结果[17]。该系统的具体测试过程，见图 5.2 。

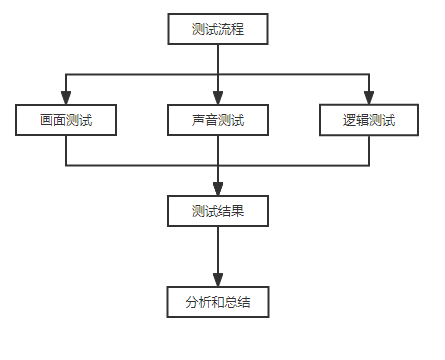


图5.2 系统测试流程图

## 测试流程

作为游戏开发流程里面不可或缺的一个部分，特别是要投入市场使用的游戏更需要经过测试流程[18]：

1. 测试是为了找出设计软件中具有的BUG。

2. 整个测试过程就是根据时间运行的环境进行测试。

3. 每个测试过程都要结合游戏可能投放的领域的背景。

4. 测试过程必须按照软件规划的正规流程运行路径进行，这样才能达到测试的效果[19]。

因为游戏的特色的软件需求，测试包括对游戏运行的平衡性、游戏画面和游戏的实际操作难度[20]。

具体进行游戏测试的流程图如图5.3所示：

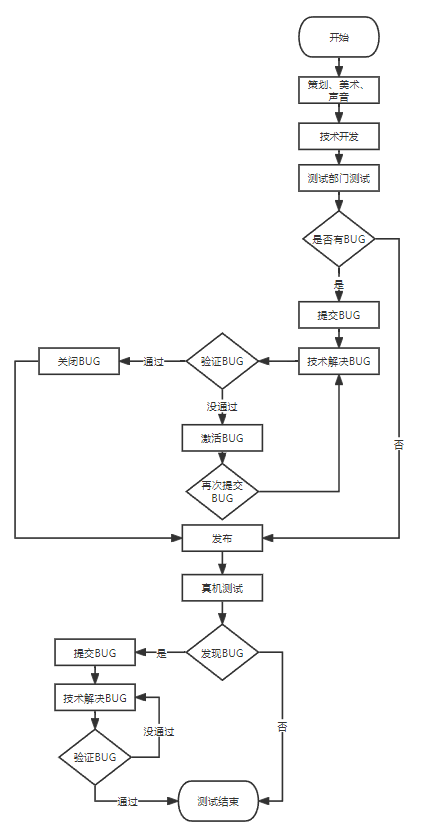


图5.3 游戏测试

## 本游戏项目的测试

因为本游戏大概率不会投入市场大量使用，本游戏只是简单对 Unity3D 引擎实现功能的探究，因此测试内容相对少[21]，从一下11个方面对游戏展开测试工作：

1.游戏安装会不会出现重复安装问题，当前操作系统是否提示玩家程序已经安装到系统中[22]。

2.对游戏数据下载和安装空间需求是否能合理估计。

3.游戏在经过不同系统的安装卸载操作以后，是否影响游戏安装数据资源包[23]。

4.游戏运行界面是否能够正常显示而不受到当前设备的参数影响，能够兼容这些影响。

5.玩家能够正常来到游戏登录界面。

6.游戏的背景音效能否正常的匹配。

7.战机操作是否和预期一样简洁。

8.系统能否准确的判断当前游戏进程情况。

9.玩家中断或者退出游戏以后能不能重新连接。

10.整体游戏运行过程是否参在卡顿现象。

11.能否兼顾不同平台的运行环境。

对上述测试项目进行测试以后，具体情况如下：

1.安装和卸载程序均不会影响游戏软件的安装资源包。

2.游戏界面能够正常的切换。

3.不同显示器读出的游戏画面正常，并且 UI 显示功能正常。

4.游戏背景音效能够匹配。

5.游戏能够自由更换设备使用。

6.战机控制简单和预先设计的效果吻合。

7.游戏实际战的弹道效果正常。

8.游戏能够中途退出和重连，不会丢失数据。

## 结果分析

测试之后也明显感觉游戏仍然存在不足之处，并且通过对程序的修改来应对存在的不足[24]。本论文基于飞行类射击游戏，展开了对游戏的正确测试和游戏显示界面测试。经过测试整体而言是能够满足基本的飞行类射击游戏需求[25]。功能测试发现玩家能够轻松的实现对飞机的控制，以及正确的瞄准和发射炮弹。游戏界面的战机图展示清晰很有动态效果[26]。界面测试的具体情况，见图5.4 所示。

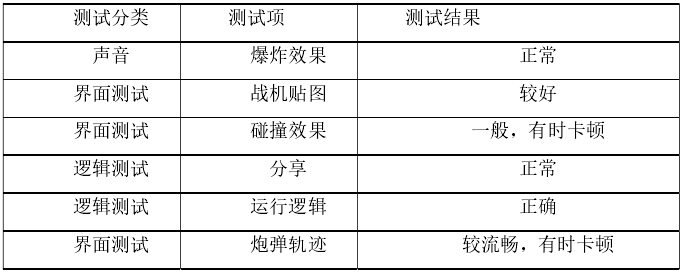


图5.4 界面测试

## 游戏可移植性说明

目前使用 Unity3D 引擎开发的游戏可以发布到 Web Player 平台、PC 平台、IOS平台、Android 平台、BlackBerry 平台、Flash Player、Windows Phone8 平台、Xbox360平台[27]。

## 本章小结

本章节主要通过阐述游戏的策划和设计，以及游戏画面和相关背景音乐程序设计，实现飞行射击游戏的需求，最终将游戏应用到手机端[28]。最主要和核心设计难点就是实现程序的编写任务和具体的测试程序实现，比如：战机方向控制功能模块、射击功能模块等[29]。

# 总结与展望

## 总结

 随着科学技术发展和人工智能领域的成熟，应用于游戏领域的引擎已经逐渐完善。本论文主要基于U3D引擎的基础上实现对飞行射击游戏的设计、相关功能实现和完成游戏功能测试方面的任务[30]。首先针对使用的引擎对游戏研究课题的背景、研究意义和价值进行了探讨，并且梳理了国内外关于这个课题的研究现状。在通过对软件整体的需求，设计相应的项目，实现相关软件分析的系统测试和非系统测试板块的功能需求分析。再对软件需求的总体设计框架进行细分，针对每个模块的任务进行详细的编程实现。最后对生成的主体软件进行功能测试操作，通过对测试结果的数据分析，得知软件已基本实现论文预期的功能需求，软件整体的性能基本符合预期。本论文重点做的研究任务如下：

（1）对游戏软件的引擎需求进行了分析，基于论文设计的飞行类射击游戏所需要的软件相关功能进行分析说明；

（2）选用Unity3D作为本论文设计需求软件的框架体系，对具体模块进行详细划分，并且梳理模块之间的相互联系；

（3）对飞行类射击游戏的需求软件进行了详细的设计，讨论了UI框架下对功能和资源器的要求、以及相关数据所需要的存储结构。基于游戏框架对相关设计类游戏的系统功能进行了详细设计；

（4）在软件技术层面，基于UI的框架结构，实现其文案、图片、背景音效的目标，对飞行类软件的资源管理进行介绍和介绍本游戏的主要玩法。

（5）对飞行类射击游戏的系统功能测试。首先对设计的软件测试，通过测试结果发现软件能够满足事先的功能需求目标。最后对软件的兼容和性能进行测试，测试结果显示所有指标均达预期。

## 展望

实现飞行类设计游戏的需求中，主要是针对设计的软件进行相关功能测试，大部分软件都能够满足设计的基本需求。不过，通常也存在一些不尽人意的地方，本课题后续的研究工作应该抓住如何实现软件优化的方面，所以可以在以下两个方面着手：

维护游戏软件相对困难。游戏存在的BUG一般都是在玩家实际操作过程中发现的，甚至玩家也很难发现游戏存在的BUG。这就要求制作者定期对游戏进行动态运行测试，这样会存在大量的工作量，如何设计出一套高效的测试软件运行状况和发现BUG的软件，已经迫在眉睫。

对于飞行类射击游戏的日常的运行性能测试分析，可以发现三维空间相对运行内存需求较大，因此一些性能较低的设备，可能进入游戏界面都需要好几分钟，这可能会降低游戏玩家的兴趣。因此后期本课题相关研究工作，可以在如何加速游戏软件在低配置的设备上的运行速度进行，找到解决的办法，吸引更多的游戏玩家来参与这类游戏。

这是个拥有一定制作难度的多人联机3D飞机大战小游戏，但很多地方还不够完美，美术资源也不够丰富，这些都需要我好好改善。

对于现在的游戏市场而言，更多的游戏案例，更丰富的学习素材，才能够带动大家投入到游戏的开发与设计中去。最后，我希望未来的游戏行业能够不断发展，人们能够正视游戏作品带来的好处，规避游戏行业现在的一些不良作风，使游戏行业形成一个良性循环。

# 参考文献

1. 徐军,张子墨.基于Unity3d射击游戏的设计及其核心功能实现[J].福建电脑,2018,34(07):111-113.
2. 胡静,胡欣宇.基于Unity3D引擎的游戏设计与开发[J].电子元器件与信息技术,2021,5(02):2096-4455.
3. 程媛.基于Unity3D的飞行小鸟游戏的设计与实现[J].电脑知识与技术,2018,14(30):65-66.
4. 谢希仁.计算机网络（第5版）[M]. 北京: 电子工业出版社, 2008: 349-352.
5. 吴晶晶,戴智超.基于Unity3D的多人在线网络游戏设计与开发[J].计算机系统应用,2017,26(10):133-138.
6. 区泽宇,李晶,魏菊霞等.基于Unity3D游戏的设计与开发[J].无线互联科技,2019,16(23):62-63.
7. 拓颖,沈浩.浅析手机游戏的发展现状及未来趋势[J].甘肃科技,2013,29(05):79-80.
8. 胡胜明,曾煜文,王晰等.基于Unity 3D的电动汽车电气故障仿真游戏设计[J].电脑知识与技术,2022,18(06):92-93.
9. 曾瀚磊,张妍,尹萍等.基于Unity3D的年兽传说游戏设计[J].现代工业经济和信息化,2021,11(10):77-79.
10. 张贝贝.基于Unity3D的音乐游戏制作[J].现代信息科技,2021,5(08):112-114.
11. 刘梦颖,马宏琳.基于Unity的冒险游戏设计与实现[J].河南科技,2021,40(17):30-32.
12. 章国雁.基于游戏引擎的第三人称射击类案例制作解析[J].安徽职业技术学院学报,2021,20(01):29-32.
13. 吴迪生.手机游戏开发技术综述[J].通讯世界,2019,26(05):239-240.
14. 沈旭,梁进大.基于Unity3D的单机竞技游戏设计与开发[J].办公自动化,2018,23(19):47-53.
15. 袁科,高启文,闫永航.基于Unidy3D的角色扮演游戏设计与实现[J].软件导刊,2019,18(08):75-79.
16. 董涛,张瑛.基于Unity3D的第三视角射击类手游设计与实现[J].通讯世界,2019,26(11):98-99.
17. 张俊,廖金巧.基于Unity3D的手机版FPS射击游戏设计与开发[J].赤峰学院学报(自然科学版),2016,32(15):38-39.
18. 陈豪.基于Unity3D的Android飞行类游戏设计[J].电脑知识与技术,2015,11(30):168-170.
19. 谢宏兰.基于Unity3D射击游戏的设计与实现[J].现代信息科技,2019,3(24):89-91+94.

1. [曹敏.基于Unity3D的多用户交互系统设计研究[J].西安文理学院学报(自然科学版),2021,24(02):58-61+114.](https://scholar.cnki.net/home/search?sw=6&sw-input=Lu.%20Zhang" \t "https://schlr.cnki.net/en/Detail/index/GARJ2019/_blank)
2. Gui Ping Lu, Guang Hong Xue, Zong Chen. Design and Implementation of Virtual Interactive Scene Based on Unity 3D[J]. Advanced Materials Research, 2011, 1380(317-319) : 2162-2167.
3. Dayan Shangguan, Xinyuan Huang. Development of Unity3D Network Functionality on Android[J]. International Journal of Hybrid Information Technology, 2016, 9(2) : 359-370.
4. CHOI Hyeoung Wook. Development of the Visualization Prototype of Radar Rainfall Data Using the Unity 3D Engine[J]. Journal of the Korean Association of Geographic Information Studies, 2015, 18(4) : 131-144.
5. Oak Ji Won, Bae Jae Hwan. Development of Smart Multiplatform Game App using UNITY3D Engine for CPR Education[J]. International Journal of Multimedia and Ubiquitous Engineering, 2014, 9(7) : 263-268.
6. Bae Jae Hwan, Kim Ae Hyun. Design and Development of Unity3D Game Engine-Based Smart SNG (Social Network Game)[J]. International Journal of Multimedia and Ubiquitous Engineering, 2014, 9(8) : 261-266.
7. Zhu Xianwen. Behavior tree design of intelligent behavior of non-player character (NPC) based on Unity3D[J].Journal of Intelligent & Fuzzy Systems,2019,37(5): 6071-6079.
8. Lim CJ, Han Won Dae, Guen Jeong Yun. Educational Game Making-Tool Development using Unity3D Engine: Birth of Game[J]. Journal of Korea Game Society, 2014, 14(1) : 29-38.
9. Jae Hwan Bae. Design and Development for Unity3D Game Engine using the Shooting Game[J]. Journal of The Korean Society for Computer Game, 2016, 29(1) : 93-100.
10. Kang Sung Yun, Park Ki Hong. Design and Implementation of Sleigh Game Content based on Physical Force using Unity3D Game Engine[J]. Journal of Digital Contents Society, 2019, 20(12) : 2301-2307.
11. Jiwu Wang, Weixin Zeng. Research on the Realization Method of Augmented Reality based on Unity3D[J]. Journal of Robotics, Networking and Artificial Life, 2019, 6(3) : 195-195.

# 致谢

首先，在毕业设计和毕业论文的完成之际，我要向给予我最大帮助的指导老师表示感谢。完成毕设和论文是一个相对艰难与漫长的过程，当我拿到相对陌生的论文题目时，头脑基本还是一片空白，从查阅相应的资料到后面阅读大量的参考文献，这一路走来实属不易。导师严谨的治学态度和科学的工作方法给了我极大的帮助，而且导师不仅在我毕业的道路上给了我很大的帮助，同时在生活中也给与了我很多关心，让我顺利完成了毕设和论文。再次向我的导师表示感谢。

此外，我还想感谢重庆邮电大学所有给我上过课的和指导过我的老师，特别是计算机科学与技术学院的老师。老师们教给我很多专业知识和宝贵的经验，让我受益终身。感谢重庆邮电大学，给我提供了优越的学习环境，在重庆邮电大学的学习生活也在我的人生历程上留下了难忘和美好的回忆。祝福重庆邮电大学的老师们，希望老师们身体健康、事业顺利。祝重庆邮电大学的同学们前程似锦，未来一片光明。希望重庆邮电大学发展的越来越好，争取早日拿到双一流的称号，在重点领域有所突破，补足国内技术短板，为国家未来科技发展培养大量的技术人才。

最后，我还要感谢我亲爱的同学们。我很庆幸我拥有着能一起进步奋进的室友们和一起相互帮助和鞭策的一群同学们。大学四年的点点滴滴都在我的脑海里一幕幕地呈现，感谢我亲爱的同学们，让我拥有这么多温馨又难忘的大学回忆，让我的大学生活既有意义又多姿多彩。

毕业不是结束，而是新的开始！年轻的我在毕业后就参加工作了，与很多同学一样，我也对我今后的生活充满着期待和憧憬，不过光是期待和憧憬还远远不够，我必须还要付出行动和努力，在我毕业的时候希望我在今后的工作中能在自己的领域实现自己的价值，为社会做出自己的贡献，让我的父母感到骄傲！让爱我的人和我爱的人感到自豪！

# 附录 英文翻译

## 英文原文

Research on Key Technologies Base Unity3D Game Engine

Abstract

Game engine is the core of game development. Unity3D is a game engine that supports the development on multiple platforms including web, mobiles, etc. The main technology characters of Unity3D are introduced firstly. The component model, event-driven model and class relationships in Unity3D are analyzed. Finally, a generating NPCs algorithm and a shooting algorithm are respectively presented to show common key technologies in Unity3D.

Index Terms—game engine, Unity3D, NPC algorithm, shooting algorithm

1. UNITY3D GAME ENGINE

A game engine provides a main framework and common functions for developing games, which is the core of controlling games. Since the first advent of the Doom game engine in 1993, game engine technology has experienced nearly 20 years of evolution. Game engine initially only supported 2D, now fully supports 3D, lifelike images, massively multiplayer online game, artificial intelligence and mobile platforms. Some representative game engines are Quake [1], Unreal Tournament [2], Source [3], BigWorld [4] and CryENGINE [5], etc. The internal implementation techniques in Game engines have some differences, but their ultimate goal is the same to improve the efficiency of game development. To comprehensively grasp the general development ideas of a game, it is necessary for choosing a typical game engine to study deep.

Unity3D is a popular 3D game engine in recent years, which is particularly suitable for independent game developers and small teams. It mainly comprises eight products such as the Unity, Unity Pro, Asset Server, iOS, iOS Pro, Android and Android Pro [6]. Without writing complex codes, programmers can quickly develop a scene by using the visual integrated development environment of Unity3D. In the popular iPhone game list, the games developed by Unity3D take a large proportion, such as Plants vs. Zombies, Ravensword:The Fallen King. In particular, Unity3D also provides the Union and Asset Store selling platforms for game developers.

Unity3D has special advantages in easily programming a game. For example, platform-related operations are encapsulated in its internal, the complex game object-relations are managed by different visual views, and JavaScript, C # or Boo scripting languages are applied to program a game. A script program will be automatically compiled into a .NET DLL file, so the three scripting languages, in essence, have the same performance, their execution speed is 20 times faster than traditional JavaScript. These script languages have good crossplatform ability as well. That means developers can deploy games on different platforms such as Windows, Mac, Xbox 360, PlayStation 3, Wii, iPad, iPhone and Android. In addition, games can run on the Web by installing a plug-in.

Another feature of the Unity3D is that game resources and objects can be imported or exported in the form of a package, which can easily make different game projects share development works. Therefore, using package can greatly improve efficiency in game development. In addition to resource material files, specific functions can be packaged, such as AI, network operation, character control, etc.

1. UNITY3D GAME ENGINE INTERNAL ANALYSIS

A.The Component Model

The component model is applied in the Unity3D game development, which provides a scalable programming architecture. Game function modules can be reused conveniently in this component model. Each entity in a game scene is called as a GameObject. A GameObject represents a game object, which has the characteristics of a container. According to the needs of a game, many different components are added into a GameObject. A component can be taken as a collection with a group of related functions, which can be accessed through an interface. For example, a script is a component and its role is to offer a logical operation on a game object, and the Box Collider component in Unity3D specifically provides a support for game objects collision detection. Unity3D has many predefined components. Programmers can combine some of them to create a featurerich GameObject. The Figure 1 shows the ideas of the component model in Unity3D.

The Figure 2 shows the tree hierarchy for organizing objects in Unity3D game project. A game is composed of one or more scenes, each scene includes one or more GameObjects, and moreover, every GameObject is composed of some components or child GameObjects.

In game development, besides directly using GameObjects predefined in Unity3D, programmers can create an empty GameObject with the information about position, rotation and scale of an object, and then add scripts or other components into it. In order to facilitate the same type of game object management, Unity3D affords the Prefab that is a technology like a template. A Prefab can contain both objects and game resources such as 3D models. When the same type of game objects need to be created, a Prefab can be used in this situation. All GameObjects will be updated simultaneously when its Prefab is changed. The above mechanism of Prefab can greatly improve the maintenance efficiency of a game.

To observe the impact of an object's state on a game, programmers can dynamically change the configuration parameters of a component when the game is running. After the game exits, all its initial parameters will be reset. In order to better grasp the impact of game elements, details of differences between game design and actual running affect can be discovered by using the game view and scene view at the same time.

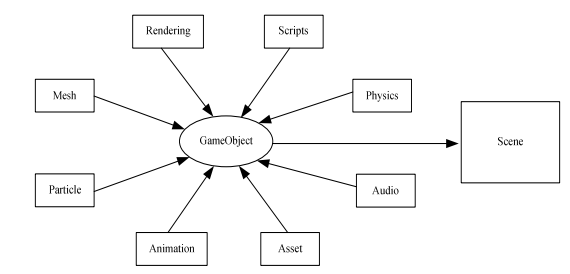


Figure 1 The Unity3D component model

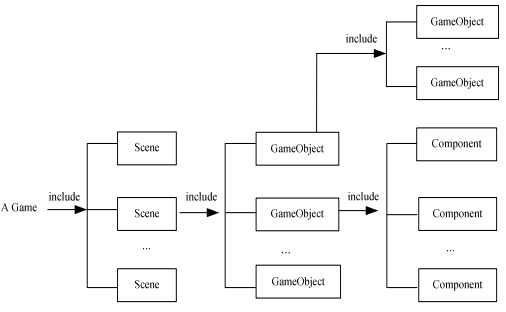


Figure 2 The Unity3D game project hierarchy

B.The Event-Driven Programming

There are a number of scripts in a game. A script is a class controlling the behavior of a game, which should inherit a base class called MonoBehaviour. MonoBehaviour defines common event triggering methods. When a predefined event occurs, an appropriate method will be automatically executed. The Figure 3 lists the important event methods in Unity3D and their execution order relationships.

The Awake() method is executed only once in the life cycle of a script instance, when the script is called. The Awake() method can be used to initialize variables when a game starts. After all GameObjects are initialized, this method will be executed. The execution order of each Awake() method in GameObjects is random. Consequently, it is safe for referencing other GameObjects in the Awake() method without causing a null object reference.

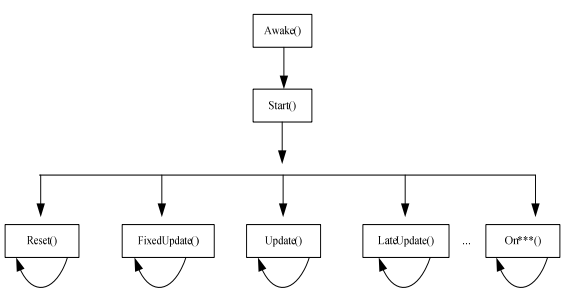


Figure 3 The event-driven model

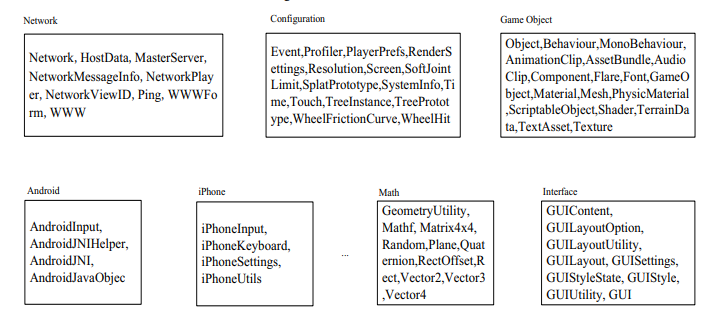


Figure 4 The distribution of main classes in Unity3D

The Start() method is executed only once like the Awake() method, and it can also be used to initialize variables. The difference between the two methods is that the Start() method is executed only when the script instance is enabled. When it needs to order initialization operations, the Start() method can be utilized to delay initializing codes.

After the execution of the Awake() and Start() methods, other event methods can be recycled to trigger in the life cycle of a game. Methods such as Reset(), FixedUpdate(), Update(), LateUpdat() can be executed in multiple times. For example, the Update() method will be called when each frame is rendered. Its implementation number depends on frames per second. It is recommended that logic operations should be put in this method.

1. The Summary of Classes

A game is controlled by programming scripts to access the internal of Unity3D. There are many system classes in Unity3D, which are useful in scripts. There are two kinds of classes. One According to function of classes, nearly 200 runtime classes are categorized into the Figure 4.

1. THE RESEARCH ON KEY TECHNOLOGIES
2. A Dynamic Algorithm of Randomly Generating NPCs

NPC is the abbreviation of non-player character. That the number and location of NPCs are dynamically changed can keep players strong interesting on a game and extend the vitality of the game. NPCs can be designed by the difficulty and logic of a game. The following will propose a dynamic algorithm of randomly generating NPCs, which can be called when some NPCs need to be created in a game.

1. Select a kind of NPC that has not been yet handled, it is named as t;
2. bCreate=0;

3.Determine whether the kind of NPC t should appear:

If(random.value>tRValue && NPCrule==1){

//determine by a probabilistic method, tRValue is the appearing probability for the kind of NPC t.

bCreate=1;

}else if(existNum<tEValue && NPCrule==2){//Existing number of the kind of NPC t is below a certain threshold value tEValue.

bCreate==2;

}else if(sometinghappen && NPCrule ==3){

//determine by an event in which some conditions are satisfied.

bCreate=3;

}

4.If(bCreate>0)

tNum=random.Range(tMinValue,tMaxValue);//determine how many NPCs of the kind of t will be created.

1. While(tNum>0){
2. Set the location of a NPC of type t:

Position=reference.transform.position+Vector3(x,y,z);//x,y and z are randomly generated;

1. Copy a NPC of type t from the template by using the instantiate method and set the behavior of its action;
2. tNum--;}
3. Determine whether it is necessary to create other kinds of NPCs,if it is necessary then goto step 1,else end this algorithm.

Now, let’s analyze the algorithm. The random operations are applied in the step 3, 4 and 6 to control the number and location of new NPCs. The step 3 determines whether certain types of NPCs should be generated by a variety of rules. In the step 4, difficulty of a game can be adjusted by regulating tMinValue and tMaxValue. The numbers of NPCs are dynamic by the corresponding kind of NPC. In the step 6, the protagonist or the pre-set coordinates in a scene can be taken as a reference point. An empty GameObject with special deployment mark can be designed as a reference for the location of NPCs.

It often requires creating a number of NPCs in a game. We introduce a simple solution of enabling them to have perfect distribution.

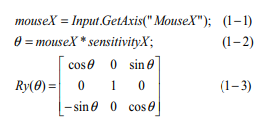
1. Manually deploying a number of referencing coordinates for NPCs in the scene by special strategies. The distribution of references should have a well density.
2. Classifying these references by setting the tag properties in Unity 3D to prepare basic data for deployment. Programmers can quickly find a group of GameObjects with a certain tag by the FindGameObjectsWithTag method.
3. Deploying NPCs by these references and deployment strategies.
4. A Shooting Algorithm

There are mainly two ways of firing bullets in game development. One is that a true 3D object is modeled by decorating a bullet with texture and color. This approach needs to control the flight path of a bullet and destroy the bullet when it is out of bounds. Another method is to use ray tracing technology. That the light collides with an object means a bullet hits the object. In order to get more realistic effects, cartridge cases are pop-upped or sparks are twinkled at the gunpoint when bullets are fired. Moreover, drawing light between the launch point and target point can more realistically simulate the firing of bullets.

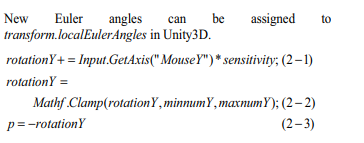
In a 3D shooting game, a sight is usually provided to players. Players can move a mouse to control it for firing. The principle of this technique is to place a graphic sight in the center of the screen. Moving the mouse is equal to move the camera, which will cause the illusion of the sight moving in the screen. In fact, the coordinates of the camera are real target coordinates in game codes.

In a first-person game, a main camera is set as a sub-object of the protagonist [7], so they can move together as a whole. There are four directions in the mouse movement, namely left, right, up and down movement. The following will discuss how to calculate them.

The mouse moving left or right is equivalent that the camera rotates the Y axis. The following formula is about the way of calculating a 3\*3 rotation matrix Ry(). is the horizontal movement angle of the mouse, which can be got through the magnitude and direction of mouse horizontal movement. Input.GetAxis("Mouse X" ) method in Unity3D can help programmers do that in a simple way. This method will return a value between -1 to 1. The sensitivityX represents the sensitivity value for the mouse, which is usually assigned to 15. To the formula (1-3), the Rotate method in Unity3D can be directly applied to rotate an angle.



Finally, we discuss the calculation principle about moving the mouse up or down. A camera moving up or down is equivalent that the head of a person (parent node) rises up or down. Moving values are in a fixed angular range. For example, angles are between -60 degrees to 60 degrees. From the mathematical point of view, it relates to the rotation of the parent's transformation and actually it can be expressed by Euler angles [8]. Euler angles represent three angles around the X, Y, Z axis rotation, which are used to simulate the rotation of local coordinates. The advantage of applying Euler angles is more intuitive, easy to understand and control than a matrix. The initial value of rotationY in formula (2-1) is 0, later it cumulates the magnitude for mouse moving up and down. The p in formula (2-3) is the Euler angle rotating around the X axis.



REFERENCES

1. Quake. http://www.idsoftware.com/games/quake/.
2. Unreal Tournament. http://www.unrealtournament.com.
3. Source. http://source.valvesoftware.com/.
4. BigWorld. http://www.bigworldtech.com/index/index.php
5. CryENGINE . http://www.crytek.com/cryengine.
6. Unity. <http://unity3d.com.>
7. Graham McAllister. Creating a First Person Shooter (FPS). http://download.unity3d.com/support/resources /files/FPS\_Tutorial\_1.pdf.

[8] Fletcher Dunn, lan Parberry, 3D Math Primer for Graphics and Game Development. Tsinghua University Press, 2005

## 英文翻译

基于Unity3D游戏引擎的关键技术研究

摘要

游戏引擎是游戏开发的核心。Unity3D是一个游戏引擎，支持在多个平台 (包括web，手机等) 上进行开发。首先介绍了Unity3D的主要技术特点。分析了Unity3D中的组件模型，事件驱动模型和类关系。最后，分别提出了生成NPCs算法和射击算法，以展示Unity3D中常见的关键技术。

**关键词：**游戏引擎，Unity3D，NPC算法，射击算法

1. UNITY3D游戏引擎

一个游戏引擎提供了开发游戏的主要框架和通用功能，这是控制游戏的核心。自毁灭战士游戏引擎1993年首次问世以来，游戏引擎技术经历了近20年的演变。游戏引擎最初只支持2D，现在完全支持3D、栩栩如生的图像、大型多人在线游戏、人工智能和移动平台。一些具有代表性的游戏引擎有雷神之锤[1] 、虚幻锦标赛[2] 、Source [3] 、BigWorld [4] 和CryENGINE [5] 等。游戏引擎中的内部实现技术有一些不同，但它们的最终目标是相同的，以提高游戏开发的效率。要全面把握游戏的总体开发思路，就需要选择一个典型的游戏引擎进行深入研究。

Unity3D是近年来流行的3D游戏引擎，特别适合独立游戏开发商和小型团队使用。它主要包括Unity，Unity Pro，Asset Server，iOS，iOS Pro，Android和Android Pro等八种产品[6]。无需编写复杂代码，程序员可以使用Unity3D的可视化集成开发环境快速开发场景。在广受欢迎的iPhone游戏列表中，Unity3D开发的游戏占据了很大的比例，如《植物大战僵尸》、《乌鸦之剑: 堕落之王》。特别是，Unity3D还为游戏开发商提供了联盟和资产商店销售平台。

Unity3D在轻松编程游戏方面具有特殊优势。例如，平台相关的操作被封装在其内部，复杂的游戏对象关系由不同的视觉视图管理，并且JavaScript，C # 或Boo脚本语言被应用于编程游戏。一个脚本程序会自动编译成A.NET DLL文件，因此三种脚本语言在本质上具有相同的性能，它们的执行速度比传统JavaScript快20倍。这些脚本语言也具有良好的跨平台能力。这意味着开发人员可以在Windows，Mac，Xbox 360，PlayStation 3，Wii，iPad，iPhone和Android等不同平台上部署游戏。此外，游戏可以通过安装插件在网络上运行。

Unity3D的另一个功能是可以以包的形式导入或导出游戏资源和对象，这可以轻松地使不同的游戏项目共享开发作品。因此，使用package可以大大提高游戏开发中的效率。除了资源材料文件外，还可以打包特定功能，例如AI，网络操作，字符控制等。

第2章 UNITY3D游戏引擎内部分析

2.1组件模型

组件模型应用于Unity3D游戏开发，它提供了可扩展的编程体系结构。游戏功能模块可以在该组件模型中方便地重用。游戏场景中的每个实体都被称为游戏对象。一个GameObject代表一个游戏对象，它具有容器的特性。根据游戏的需要，在一个GameObject中添加了许多不同的组件。可以将组件视为具有一组相关功能的集合，可以通过接口访问这些功能。例如，脚本是一个组件，其作用是对游戏对象提供逻辑操作，Unity3D中的Box Collider组件专门为游戏对象碰撞检测提供了支持。Unity3D有许多预定义的组件。程序员可以结合其中的一些来创建一个featurerich GameObject。图1显示了Unity3D中组件模型的思想。

图2显示了用于在Unity3D游戏项目中组织对象的树层次结构。一个游戏由一个或多个场景组成，每个场景包括一个或多个游戏对象，而且，每个游戏对象都由一些组件或子游戏对象组成。

在游戏开发中，除了直接使用Unity3D中预定义的GameObjects之外，程序员还可以创建一个空的GameObject，其中包含有关对象的位置，旋转和缩放的信息，然后在其中添加脚本或其他组件。为了促进相同类型的游戏对象管理，Unity3D提供了类似模板的技术。预制件可以同时包含对象和游戏资源，例如3D模型。当需要创建相同类型的游戏对象时，可以在这种情况下使用预制件。所有GameObjects将在其预置更改时同时更新。预制件的上述机制可以大大提高游戏的维护效率。

为了观察对象的状态对游戏的影响，程序员可以在游戏运行时动态更改组件的配置参数。游戏退出后，其所有初始参数将被重置。为了更好地把握游戏元素的影响，可以通过同时使用游戏视图和场景视图来发现游戏设计与实际运行效果之间的差异。

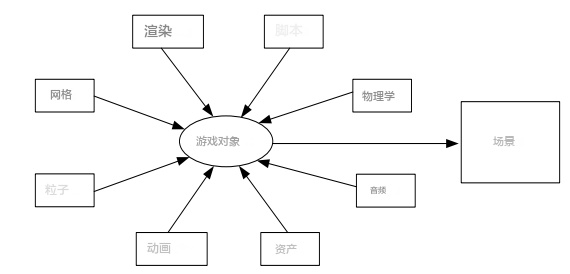


图1 Unity3D组件模型

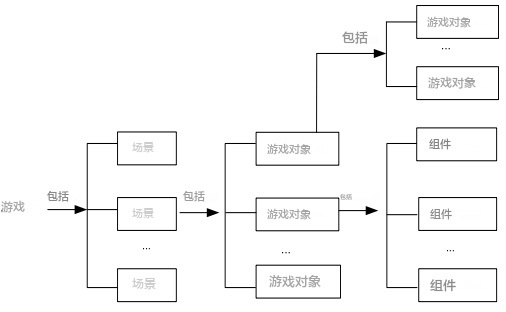


图2 Unity3D游戏项目层次结构

2.2事件驱动编程

游戏中有许多脚本。脚本是控制游戏行为的类，它应该继承一个称为monobehavior的基类。Monobehaviobehaviour定义了常见的事件触发方法。当预定义的事件发生时，将自动执行适当的方法。图3列出了Unity3D中的重要事件方法及其执行顺序关系。

当调用脚本时，Awake() 方法在脚本实例的生命周期中仅执行一次。当游戏开始时，可以使用Awake() 方法初始化变量。在所有GameObjects被初始化后，这个方法将被执行。GameObjects中每个Awake() 方法的执行顺序是随机的。因此，在Awake() 方法中引用其他gameobject是安全的，而不会导致空对象引用。

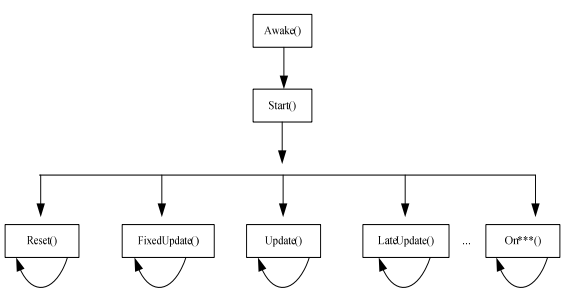


图3事件驱动模型



图4 Unity3D中主要类的分布

Start() 方法和Awake() 方法一样只执行一次，也可以用来初始化变量。两种方法之间的区别在于，仅在启用脚本实例时才执行Start() 方法。当它需要命令初始化操作时，可以利用Start() 方法延迟初始化代码。

在执行Awake() 和Start() 方法之后，可以回收其他事件方法以在游戏的生命周期中触发。Reset() 、FixedUpdate() 、Update() 、lateuppat () 等方法可以多次执行。例如，在渲染每个帧时将调用Update() 方法。它的实现数量取决于每秒的帧。建议在这种方法中放入逻辑运算。

2.3 类的总结

游戏由编程脚本控制，以访问Unity3D的内部。Unity3D中有许多系统类，这些类在脚本中很有用。有两种课程。根据类的功能，将近200个运行时类分类到图4中。

第3章 关键技术的研究

3.1一种随机生成npc的动态算法

NPC是非玩家角色的缩写。Npc的数量和位置是动态变化的，可以让玩家对游戏保持强烈的兴趣，并延长游戏的活力。Npc可以通过游戏的难度和逻辑来设计。以下将提出一种随机生成npc的动态算法，当游戏中需要创建一些npc时，可以调用该算法。

1.选择一种尚未处理的NPC，命名为t；

2. bCreate=0;

3.判断是否应该出现NPC t的种类:

如果 (random.value> tRValue & & NPCrule = = 1){

// 通过概率方法确定tRValue是NPC t的出现概率。

bCreate = 1;

} else if(existNum<tEValue & & NPCrule = = 2){// NPC t的种类的现有数量低于某个阈值tEValue。

bCreate = = 2；

} 否则，如果 (某些事情发生 & & NPCrule = = 3){

// 由满足某些条件的事件确定。

bCreate = 3;

}

4. if(bCreate>0)

tNum = random.Range(tMinValue，tMaxValue);// 确定将创建多少种t的npc。

5. While(tNum>0){

6.设置类型为t的NPC的位置:

Position = reference.transform.position + Vector3(x，y，z);// x，y和z是随机生成的；

7.使用实例化方法从模板中复制类型为t的NPC，并设置其动作的行为；

8.tNum --;}

9.确定是否有必要创建其他类型的npc，如果有必要，则转到步骤1，否则结束此算法。

现在，让我们分析一下算法。在步骤3、4和6中应用随机操作来控制新npc的数量和位置。步骤3确定是否应通过各种规则生成某些类型的npc。在步骤4中，可以通过调节tMinValue和tMaxValue来调整游戏的难度。NPC的数量是由相应类型的NPC动态的。在步骤6中，可以将场景中的主角或预先设置的坐标作为参考点。可以设计带有特殊部署标记的空GameObject作为npc位置的参考。

它通常需要在游戏中创建许多npc。我们介绍了一个简单的解决方案，使它们具有完美的分布。

1.通过特殊策略为场景中的npc手动部署许多参考坐标。参考文献的分布应具有井密度。

2.通过在Unity 3D中设置标记属性来对这些引用进行分类，以准备用于部署的基本数据。程序员可以通过FindGameObjectsWithTag方法快速找到一组具有一定标签的游戏对象。

3.通过这些参考文献和部署策略部署npc。

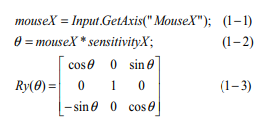
3.2 一种射击算法

游戏开发中主要有两种发射子弹的方式。一种是通过用纹理和颜色装饰子弹来建模真正的3D对象。这种方法需要控制子弹的飞行路径，并在子弹出界时将其摧毁。另一种方法是使用光线追踪技术。光线与物体碰撞意味着子弹击中了物体。为了获得更逼真的效果，弹壳会弹出或在发射子弹时在枪口处闪烁火花。而且，在发射点和目标点之间画光可以更真实地模拟子弹的发射。

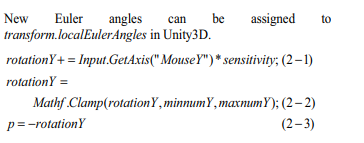
在3D射击游戏中，通常会向玩家提供瞄准具。玩家可以移动鼠标来控制它进行射击。这种技术的原理是在屏幕中央放置一个图形瞄准器。移动鼠标等于移动相机，这会导致视线在屏幕中移动的错觉。实际上，摄像机的坐标是游戏代码中的真实目标坐标。

在第一人称游戏中，一个主摄像机被设定为主角的子对象[7]，因此它们可以作为一个整体一起移动。鼠标移动有四个方向，分别是左、右、上、下移动。下面将讨论如何计算它们。

鼠标向左或向右移动相当于相机旋转y轴。下面的公式是关于计算一个3\*3旋转矩阵Ry() 的方式。是鼠标的水平移动角度，可以通过鼠标水平移动的幅度和方向得到。Unity3D中的Input.GetAxis (“Mouse X”) 方法可以帮助程序员以一种简单的方式做到这一点。此方法将返回-1到1之间的值。sensitivityX表示鼠标的灵敏度值，通常将其分配为15。对于公式 (1-3)，可以直接应用Unity3D中的旋转方法来旋转角度。



最后，我们讨论了上下移动鼠标的计算原理。相机向上或向下移动相当于人 (父节点) 的头部向上或向下上升。移动值在固定的角度范围内。例如，角度在-60度到60度之间。从数学的角度来看，它与父变换的旋转有关，实际上可以用欧拉角表示[8]。欧拉角表示围绕X，Y，z轴旋转的三个角度，用于模拟局部坐标的旋转。应用欧拉角的优点是比矩阵更直观，易于理解和控制。式 (2-1) 中rotationY的初始值为0，后来累加了鼠标上下移动的幅度。式 (2-3) 中的p是绕x轴旋转的欧拉角。



参考文献

[1] Quake. http://www.idsoftware.com/games/quake/.

[2] Unreal Tournament. http://www.unrealtournament.com.

[3] Source. http://source.valvesoftware.com/.

[4] BigWorld. http://www.bigworldtech.com/index/index.php

[5] CryENGINE . http://www.crytek.com/cryengine.

[6] Unity. <http://unity3d.com.>

[7]Graham McAllister. Creating a First Person Shooter (FPS). http://download.unity3d.com/support/resources /files/FPS\_Tutorial\_1.pdf.

[8] Fletcher Dunn, lan Parberry, 3D Math Primer for Graphics and Game Development. Tsinghua University Press, 2005