西南科技大学

Southwest university of science and technology

本科毕业设计（论文）

裸眼 3D 技术
在沉浸式体感游戏中的应用研究

学院名称	计算机科学与技术学院
专业名称	软件工程
学生姓名	周和繁
学号	20121241
指导教师	吴亚东（教授）

二〇一六年六月

裸眼 3D 技术在沉浸式体感游戏中的应用研究

摘要：随着裸眼3D 显示技术的逐渐成熟，结合自然体感和裸眼3D 视觉的沉浸式交互展示系统成为应用热点。本文详细介绍了柱状透镜式裸眼3D 显示技术的原理以及多视点裸眼3D 的实现方案，针对体感交互设计了动作识别算法。设计实现了一款可局域网联机的裸眼3D 体感飞行设计游戏系统。通过捕捉识别人体手臂展开左右倾斜，身体前后倾斜等肢体动作，实现了对飞船的控制。游戏中系统结果表明，结合体感交互的裸眼3D 展示游戏增强了沉浸感，而且显著增强游戏吸引力。

关键词：裸眼3D；体感游戏；Kinect；Unity；局域网联机

The Design and Application of Immersing Game Based On Auto Stereoscopic 3d and Gesture Interaction

Abstract： With Autostereoscopic 3D display technology rapidly developing, the immersive interactive display system which combined with natural somatosensory and Autostereoscopic 3D vision has become a hot spot. In this paper, the principle of 3D display technology and the implementation of multi-view Autostereoscopic 3D was detailed introduced, in addition, a action recognition algorithm is designed for body interaction in system. Finally, a multi-player LAN Autostereoscopic 3D Somatosensory flight game was developed. By capturing the human body action to recognize tilt of body, human body can control of the spacecraft by posture. The result shows that the combination of the Autostereoscopic 3D interactive display game enhanced Somatosensory, but also enhance the attractiveness of the game significantly

Key words: Stereoscopic 3d；Motion-somatosensory game；Kinect；Unity；Online game

目 录

第 1 章 绪论

1.1 概述

随着科技的不断发展，人类对影音体验的需求不断扩张，从1939年推出的黑白电视机，到1954年推出 RCA 彩色电视机。从早期的默片电影，到1952年第一部3D 长片，直至2009年规模空前、技术最先进的3D 电影《阿凡达》，让全世界用户充分体验到了3D 立体视觉所带来的视觉震撼感受，但这种体验只能在电影院才能体验，因此裸眼 3D 显示器的关注度和推广力度也不断的提升。从3D 显示技术从诞生到现在，经过长达70多年的研究，3D 显示技术不断突破技术难点，直至今日，目前市面上的立体显示设备主要有头盔显示器、3D 立体眼镜、裸眼立体显示器以及手执式观测器[1]。

人眼看到立体影像的主要原理是双目视差是形成立体视觉，即要看到立体图像必须使左右眼分别看到有一定差别的图像[2]。通过特定的装置将输入的两路图像分别转换为左右眼图像，两个眼睛各自看到物体具有不同的空间信息，我们的大脑使用这些信息来判断物体的距离，产生一个错觉，感到两张图片被从屏幕上拔出来，并通过延长线投射在屏幕的前面。在两个眼睛的光线交汇的地方，就看到一个立体的图像[3]。

电子游戏从其诞生以来，一直备受广大玩家的追捧。同时游戏行业也不断去陈推新，如今电子游戏已经发展成了文化产业中的支柱型产业。虽然游戏内容日新月异，但是游戏的输入方式却依然无法脱离鼠标键盘等简单的交互方式，玩家首先需要把想要的操作转化为鼠标键盘对应的操作才能与游戏交互，这样增加了游戏的操作难度，同时也让游戏与玩家产生了隔阂。为了让玩家获得更自然的游戏体验，世界著名科技公司 Microsoft 退出了 Kinect 体感设备，微软的 Kinect 不需要使用任何控制器，它依靠相机捕捉三维空间中玩家的运动[4]，让玩家与游戏有了更自然的交互方式。

因而，裸眼3D 技术和体感交互技术成为未来几年中值得研究的一个重要方向，并且随着 Kinect 的升级版 Kinect One 的到来并发布了 Unity 的 SDK 开发包[5]。而 Unity 本身强大的跨平台优势特性，也使得它更为容易的能够实现一次开发、多次发布。同时 Unity 相较其他游戏引擎而言，物理系统、粒子系统、动画系统、及输入控制几大系统的高度集成化和大量可扩展、便于管理维护的插件也更加方便用户开发[6]，尤其是 Unity5.0之后加入了 Enligten 的光照系统，提升了 Unity 光影表现的短板。总而言之，通过 Unity 游戏引擎来开发体感游戏具有平台和技术支持上的双重便捷性[7]。

1.2 课题背景及意义

裸眼 3D 是影像行业最前沿的高新技术，它的出现和发展改变了传统平面图像带给人们的视觉疲惫感，以其生动的表现力，优美高雅的环境感染力以及强大的视觉冲击力感染了观者[8]。它是图像制作领域的一场革命，更是未来世界的主流发展趋势。因此，裸眼 3D 的探索与研究对于未来文化创意产业的发展具有重大意义。

而裸眼3D 技术在沉浸式体感游戏中的应用研究则是将先进数字视听技术与全新交互方式进行结合的一种尝试，将裸眼 3D 影像配合体感互动体验方式，力求打破常规交互方式的局限，为玩家提供一种更为新颖、有趣、真实的游戏体验。本文主要探究裸眼 3D 与体感交互结合的游戏设计方法，力求在给游戏的形式上注入新的活力，给玩家更加真实的操控体验与视觉冲击。

1.3 国内外研究现状

将裸眼技术运用到体感游戏是一个同时要求软硬件能完美结合的综合性课题，本来体感游戏的开发就需要了解 Kinect 的工作原理，熟悉 SDK 提供的 API 接口，如何识别用户的一次输入手势，还需要较强的3D 数学能力。通过计算 Kinect 获取到的人体关节点坐标之间的相对位置，对算法能力也有比较高的要求。尤其是裸眼模块，需要根据裸眼屏幕的硬件特性实现专门的裸眼图片合成算法，对每一帧获取的图像处理[9]。因而相比于开发游戏内容，对实现裸眼3D 技术的发展更优促进意义的地方在于如何设计更自然的交互手势，实现姿势识别算法，提高动作响应时间和精确度，以及如何设计裸眼合图算法，实现更好的裸眼3D 效果。

下面将分几个方面来讨论裸眼3D 应用的研发重难点：

（1）在游戏的内容题材上，作为体感游戏需要游戏场景比较适合场景互动，游戏的玩法需要适配体感操作。为了达到更好的裸眼3D 效果，场景内的模型需要菱角分明；

（2）游戏品质上要求较高，一般的3D 场景大中型的游戏都比较系消耗内存，而因为裸眼算法的约束，需要用八个摄像机同时渲染，在渲染压力上是普通游戏的八倍，这就需要游戏场景上设计上扬长避短，重点突出裸眼的效果；

（3）硬件设备要求较高，开发成本高：单纯的裸眼渲染应用已经要求计算机配置不低于2.5GH 主频、2G 内存，并且还需要加入需要大量计算量的 Kinect，再加上

Unity 引擎开发游戏本身对机器的要求，三者结合来看，裸眼3D 体感游戏开发过程中的性能要限制较高；

从以上几个方面综合而言，开发眼3D 体感游戏的限制条件比较多。加之裸眼技术也在近几年才真正成熟起来[10]，裸眼应用的研究空间还有很多可以提升。尽管如此，国内外在裸眼技术及其应用裸眼体感游戏上已经取得了一定的研究成果。

在前在国外的各种展会上发布的裸眼3D 产品有很多种，主要以裸眼3D 液晶显示器、裸眼3D 电视机、裸眼3D 手机、裸眼3D 游戏机等为代表。从技术上来看，国外的企业研究机构对裸眼3D 技术主要以光屏障式（Barrier）、柱状透镜(Lenticular Lens)技术为主[11][12][13]，而在近几年的全球展会中裸眼3D 产品也成为各大企业对未来市场规划的重点。相继推出了具有企业特色的裸眼3D 产品。关于裸眼3D 的国外企业产品研究情况见表1-1。

表1-1 裸眼 3D 国外企业研究总结表

企业名称	研究方向	研究内容	发布产品	
			年份	产品
夏普	裸眼3D 手机、裸眼显示器	光屏障式	2010	裸眼3D 手机 SH8158U
			2011	裸眼3D 手机 SH8298U
			2013	与飞利浦合作 85 寸 8K 显示器 Fractional Lenticular Lens
LG	裸眼显示器	分光式显示	2011	25 寸显示器 Flatron DX2000
			2012	25寸高清显示器 D2500N-PN
东芝	裸眼显示器、3D 笔记本	柱状透镜	2011	Qosmio F750 笔记本
			2013	21英寸4K 医用裸眼3D 显示器
飞利浦	裸眼显示器	柱状透视	2013	与夏普合作85寸8K 显示器 Fractional Lenticular Lens
			2013	4K*2K 的全高清液晶面板 PFL8830系列
HTC	裸眼3D 手机	立体显示	2011	HTC EVO-3D
三星	裸眼显示器		2014	55寸 Glassless 3D UHD TV
索尼	裸眼显示器、3D 摄像机	光屏障式摄像机	2011	24 寸裸眼显示器
			2012	3D 摄像机 TD20E
任天堂	游戏掌机	光屏障式	2011	任天堂3DS
松下	裸眼显示器	光屏障式	2012	103寸等离子显示器
			2012	145寸8K Super Hi Vision TV

富士	3D 摄像机	摄像机	2009	富士 FinePix REAL 3D W1
			2010	富士 3D W3

国内的裸眼3D 研究主要以裸眼3D 设备的硬件开发为主，国内已有的设备款式和功能相对单一，造成国内裸眼3D 产业长时间局限于视频广告、商品展示等领域，缺乏对内容表现和制作上的创新，在内容表现上主要以 2D/3D 内容转化为主[14]，而这种传统的裸眼 3D 展示内容制作流程复杂繁琐、限定因素过多，形成了如今国内裸眼 3D 市场“有先进的技术产品，却没有适合表现的内容的尴尬局面。关于裸眼3D 国内企业研究总结情况见表1-2。

表1-2 裸眼 3D 国内企业研究总结表

企业名称	研究方向	研究内容	开发产品
长虹	裸眼显示器	光屏障式	21.5英寸、48英寸、55英寸、82英寸裸眼3D 显示器
			21.5英寸、48英寸、55英寸、82英寸4K 裸眼3D 显示器
重庆卓美华视光电有限公司	立体显示 游戏互动	柱状透镜	裸眼3D 笔记本
			21.5英寸、46寸英寸、82英寸裸眼3D 广告机
广州市朗辰电子科技有限公司	立体显示 软件开发 影像制作	柱状透镜 光屏障式	58、86英寸4K 超高清显示器
			24、32、46、55、65、82英寸1080高清显示器
			对媒体广告系统
浙江天禄光电有限公司	立体显示2D 内容制作3D 内容制作	光屏障式	裸眼立体3D 四核手 N3D-D500
			65寸教学触摸一体机
			裸眼立体平板电脑
			裸眼立体拼屏3*3
			40寸裸眼立体显示器

1.4 本论文结构安排

本论文的结构安排如下：

第1章 绪论：主要分析了裸眼体感游戏的主要研究内容、研究涉及到的技术方面以及当前主要采用的开发手段，该课题的项目背景以及研究意义，国内外在该课题上取得的研究成果。

第2章 裸眼3D 的技术基础：主要介绍了人眼立体视觉形成的原理以及基于该原

理的裸眼3D 显示技术，并详细介绍了柱状透镜式裸眼3D 的原理，最后还介绍了多视点裸眼3D 技术。

第3章 体感游戏的技术基础：主要介绍了游戏开发中的组织分工、开发体感游戏过程中的技术要点，游戏美术制作的基本流程以及游戏动画的主要制作方法，游戏策划的使命与职责。最后还介绍了体感游戏接入 Kinect SDK 需要做的准备工作。

第4章 裸眼3D 体感游戏系统方案设计：主要介绍了系统的总体框架，并具体阐述了游戏美术设计；飞船控制系统；裸眼模块实现；网络通信系统；Kinect 的数据获取和工作识别。

第5章 系统实现与结果分析：主要展现了系统中各个功能的实现效果，对实现的结果进行了分析与总结，验证了该系统的可靠性、稳定性以及性能的消耗情况，总结了该系统存在的不足之处与可以改进的方面。

第 2 章 裸眼 3D 体感游戏系统需求分析

本系统将把裸眼 3D 技术应用到一款 3D 体感游戏中，游戏内容主要是局域网联机的飞行射击。游戏采用个人竞技模式，即其他所有的玩家都是自己的敌人，在游戏中玩家要通过肢体动作操作自己的飞船避过障碍物和其他玩家的激光炮，还要找到机会发射激光炮击落其他玩家的飞机。

2.1 项目目标

系统应实现的功能有：

（1） 实现裸眼 3D 显示，玩家在不佩戴任何特殊设备的情况下，能看到明显的立体效果。

（2） 实现体感交互，玩家通过自然人体姿势与游戏交互。

（3） 实现局域网联机功能，玩家能通过局域网一起游戏。

2.2 游戏规则

玩家可在同一局域网下加入其它玩家建立的房间，也可以自己建立房间等待其它玩家的加入，每个房间最多允许 4 个人加入，当房间内的所有玩家都已经准备好之后游戏开始。在游戏中，其它玩家都是自己的敌人，玩家的目的控制飞船射击以击落其它玩家的飞船。每个玩家的飞船初始有 6 滴血，每被击中一次减少一滴血，当血量为 0 时飞船被击落，飞船被击落 3S 后飞船重生。当有玩家总共击落 3 艘飞船后，游戏结束。

2.3 涉众分析

本系统的本质是一款游戏，所以其涉众只有玩家和开发人员和玩家两类，具体的涉众分析见表 2-1。

表2-1 系统涉众分析

编号	涉众名称	对系统的期望
1	玩家	希望看到裸眼3D 游戏效果，游戏有较高的可玩性，能通过肢体与游戏交互。
2	开发人员	能够满足玩家对系统的需求，实现稳定、流畅的游戏。

2.4 游戏非功能性需求

软件的非功能性需求是允许软件以满足用户的业务需求部分以外的要求。非功能性需求的是一个很容易被忽视的模块，但它对一个软件的开发和维护是非常重要的。游戏的本质也是软件，系统非功能性需求应包括系统性能，可靠性，可维护性，健壮性等方面的问题。本游戏的非功能性需求定义如下：

（1）易用性：界面美观，交互姿势符合自然人体动作。

（2）可扩展性：游戏代码结构设计合理，方便扩展加入更多玩法设计。

（3）可靠性：游戏运行稳定，努力不发生故障。

（4）健壮性：游戏在发生故障时应有合理的处理和容错机制。

2.5 游戏功能性需求分析

本游戏的主要用例描述如下，用例图如图 2-1 所示。

（1） 设置显示模式：玩家设置游戏是以裸眼 3D 显示模式还是普通模式运行游戏。

（2） 设置控制模式：玩家设置游戏的操作方式是体感操作或是键盘鼠标操作。

（3） 背景音乐设置：玩家设置是否打开背景音乐。

（4） 控制飞船飞行：玩家操作飞船左右转向，加速，以及爬升和俯冲。

（5） 控制飞船射击：玩家操作飞船瞄准射击其它飞船。

（6） 查看游戏结算：游戏结束后，玩家查看击杀排行榜。

（7） 建立房间：玩家建立房间等待其它玩家加入。

（8） 加入房间：玩家选择房间加入。

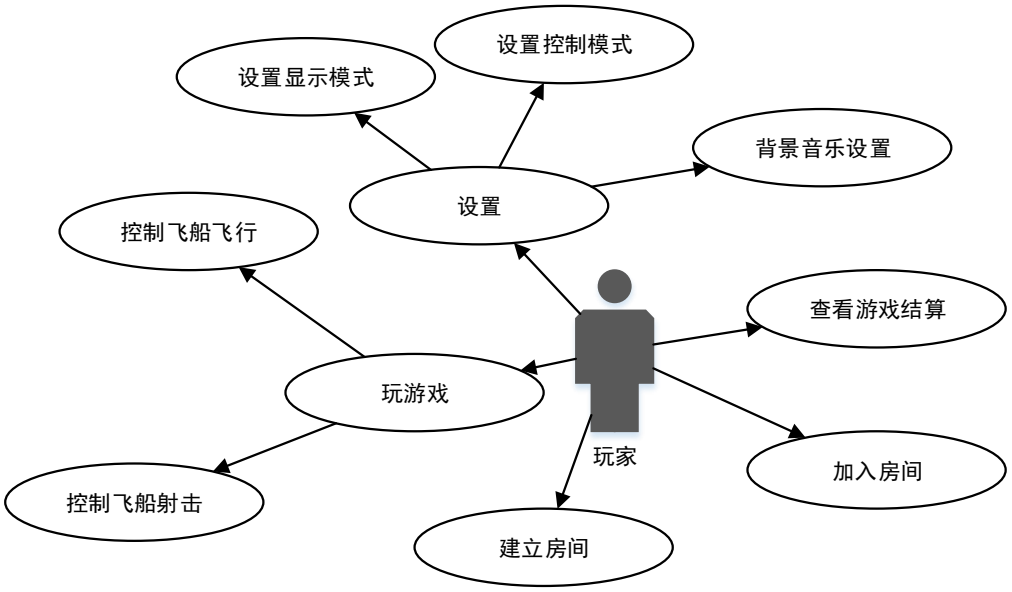


图 2-1 游戏用例图

2.6 业务流程

游戏的业务流程如图 2-2 所示，玩家点击游戏 exe 程序，运行游戏，游戏打开后首先是主选项界面，玩家可旋转加入其它玩家建的游戏房间也可以自己创建游戏房间等待加入，当房间内所有玩家都准备后，跳转到游戏界面，玩家开始游戏，游戏结束后可选择回到主选项界面重新开始游戏也可以退出游戏。

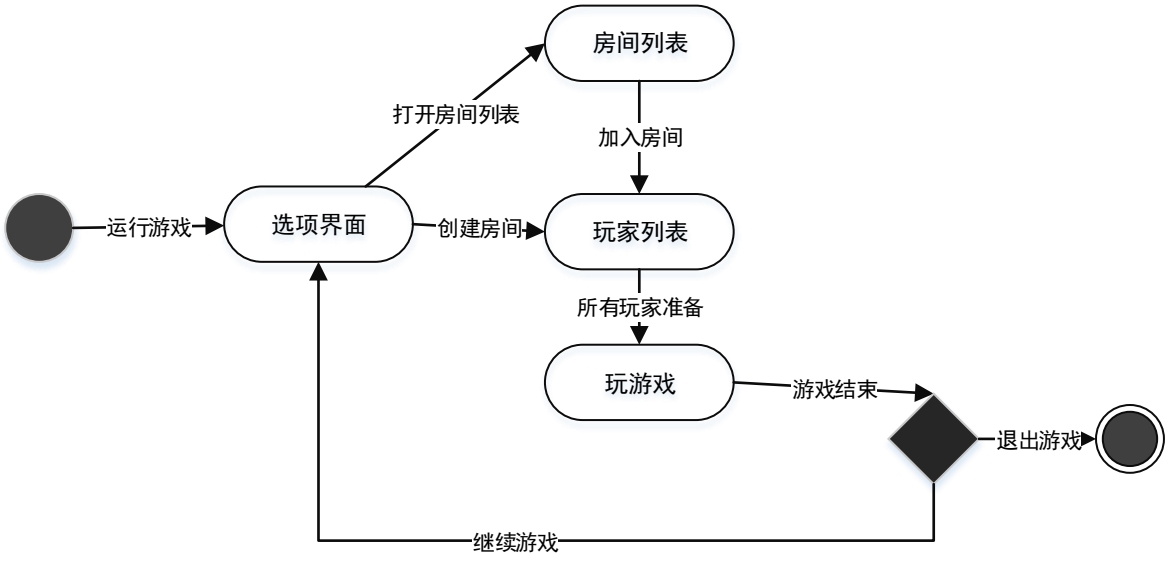


图 2-2 业务流程图

2.7 系统功能设计

本游戏主要分为 5 个模块，如图 2-3 所示，分为裸眼模块、网络模块、体感交互模块、界面模块、游戏逻辑模块。裸眼模块主要处理关于裸眼图像的获取和最终裸眼图像的合成，同时还要包括为游戏提供切换显示模式的功能，网络模块主要处理处理网络相关逻辑。体感交互模块主要处理体感姿势的识别，获取到玩家的操作意图提交到游戏逻辑中，并且需要实现空气鼠标模拟，为方便界面事件，需要兼容 UGUI 的事件系统。界面模块主要处理游戏中各种界面的界面响应和界面刷新，游戏逻辑模块主要根据体感模块传入的玩家操作控制飞船的运动，以及射击，更重要的该模块还要负责游戏的伤害计算。

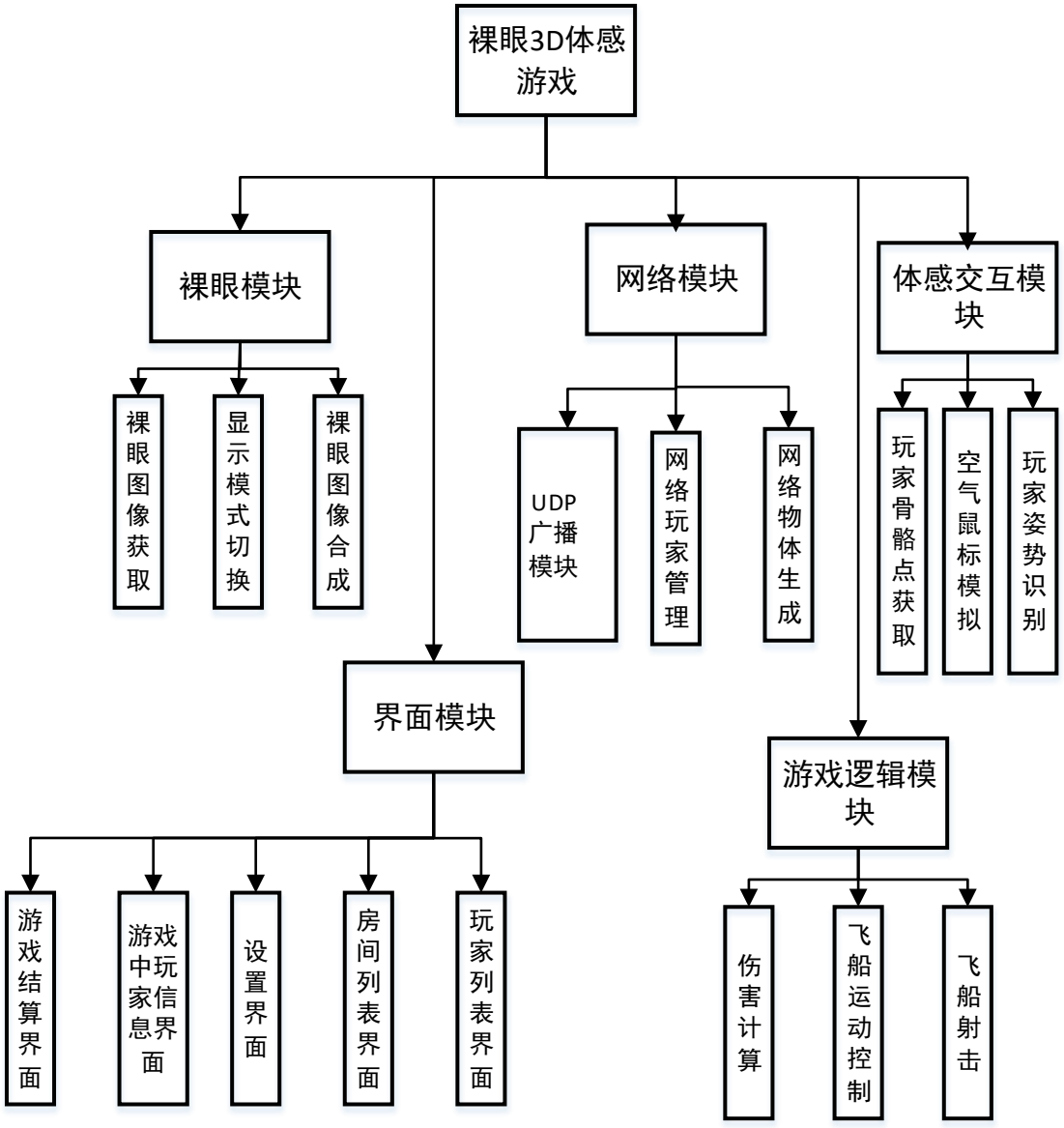


图 2-3 系统功能模块图

2.8 分析类图

经过上一节的系统功能分析，得出如图 2-4 和图 2-5 的一共 19 个类，由于游戏采用的是 Unity 开发，而 Unity 采用的组件式结构，组件式结构提倡类之间采用组件结合而不是继承，这样可以降低类之间的耦合性，提高类之间的灵活性，因此本游戏中的类大多只有关联关系，并没有太多继承关系，下面将分别介绍着些类。

界面管理基类：由于界面类主要处理界面的按钮逻辑，公共方法只抽出了关闭界面和打开界面两个方法。

设置界面管理类：主要包括对显示模式按钮和控制模式按钮两个按钮的响应操作。

玩家列表界面控制类：该类主要的功能是显示房间类的玩家信息。

房间信息类：为一个房间的实体类，包括房间名，房主，并且提供了加入该房间的方法。

房间列表界面管理类：主要负责房间信息的显示和刷新，并且还要处理加入房间按钮的响应。

UDP 广播类：工具类，封装 UDP 广播相关操作。

房间管理类：提供广播的相关函数，并且通过广播消息维护局域网中的房间列表，当房间列表变化时驱动房间列表界面刷新。

游戏信息界面管理类：监听玩家信息改变事件，当玩家信息改变时，刷新玩家的信息，主要包括玩家血量，玩家击杀数。

摄像机控制类：加到游戏主摄像机上的组件，主要功能为控制摄像机跟随玩家的飞船。

飞船运动控制类：通过玩家控制类驱动，当玩家运动状态改变，控制飞船物体运动和发射子弹。

玩家控制类：保存玩家的各种信息，包括姓名，血量，攻击力，击杀数，死亡数，玩家状态，飞船的类型。监听输入管理类的输入事件并驱动飞船控制类来控制飞船物

体运动，当接收到射击事件后，实例化子弹。该类还提供进入房间和准备开始游戏的方法。

子弹控制类：子弹物体上的组件，当子弹被实例化后，控制子弹飞行，并且当碰撞到其他物体后将碰撞到的物体和子弹的所有者发送给网络游戏管理类进行伤害判断。

Kinect 控制类：封装 Kinect 操作，从 KinectSDK 获取到人体关节点数据，为姿态识别类提供 Kinect 的操作方法。

裸眼显示类：封装裸眼显示的操作，控制裸眼相机的排列，并且负责合成裸眼图像，为设置界面提供切换显示模式方法。

键盘输入类：监听键盘输入，将有意义的键盘输入转化为操作事件提供给输入管理类。

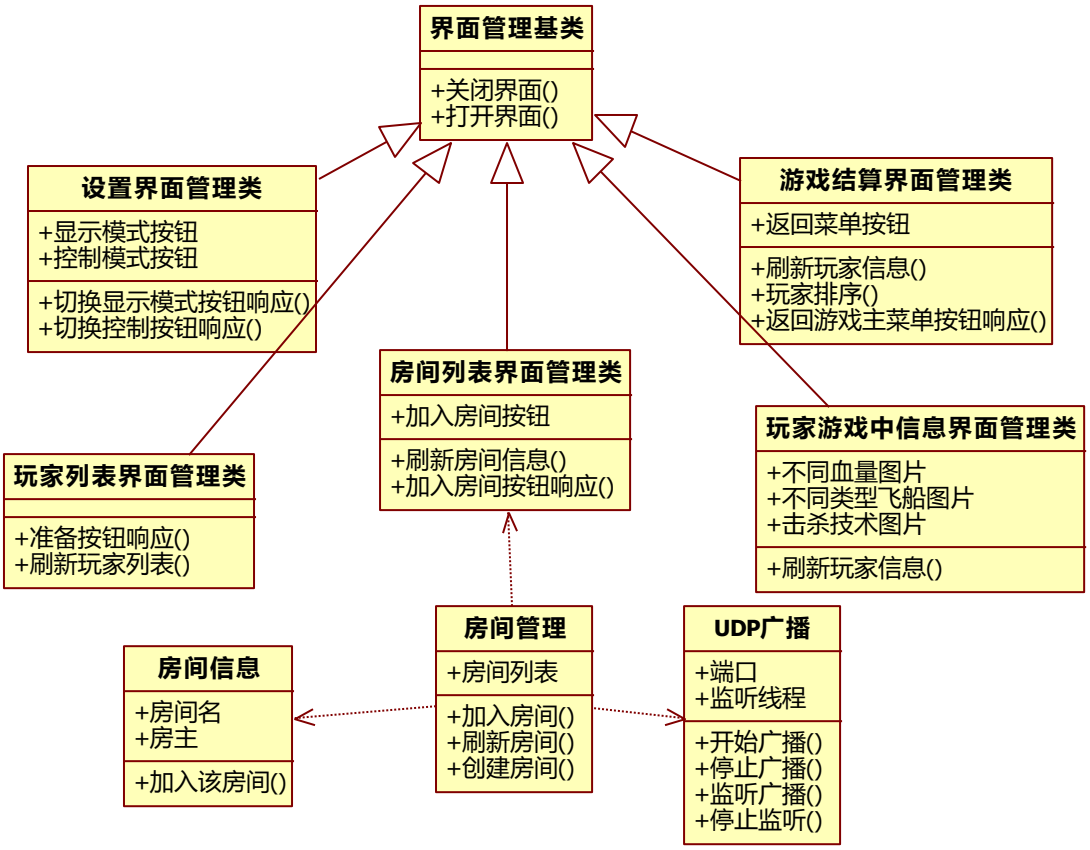


图 2-4 类间的关系 1

人体姿势识别类：通过人体骨骼数据识别人体动作，将有意义的动作映射为玩家操作时间提供给输入管理类。

输入管理类：提供玩家的输入接口给游戏逻辑，同时提供切换输入方式的方法。

网络游戏管理：管理整个游戏的运行，采用单例模式设计，负责网络联接的管理，网络物体的创建和同步，同时还要维护一个当前房间中的玩家列表。

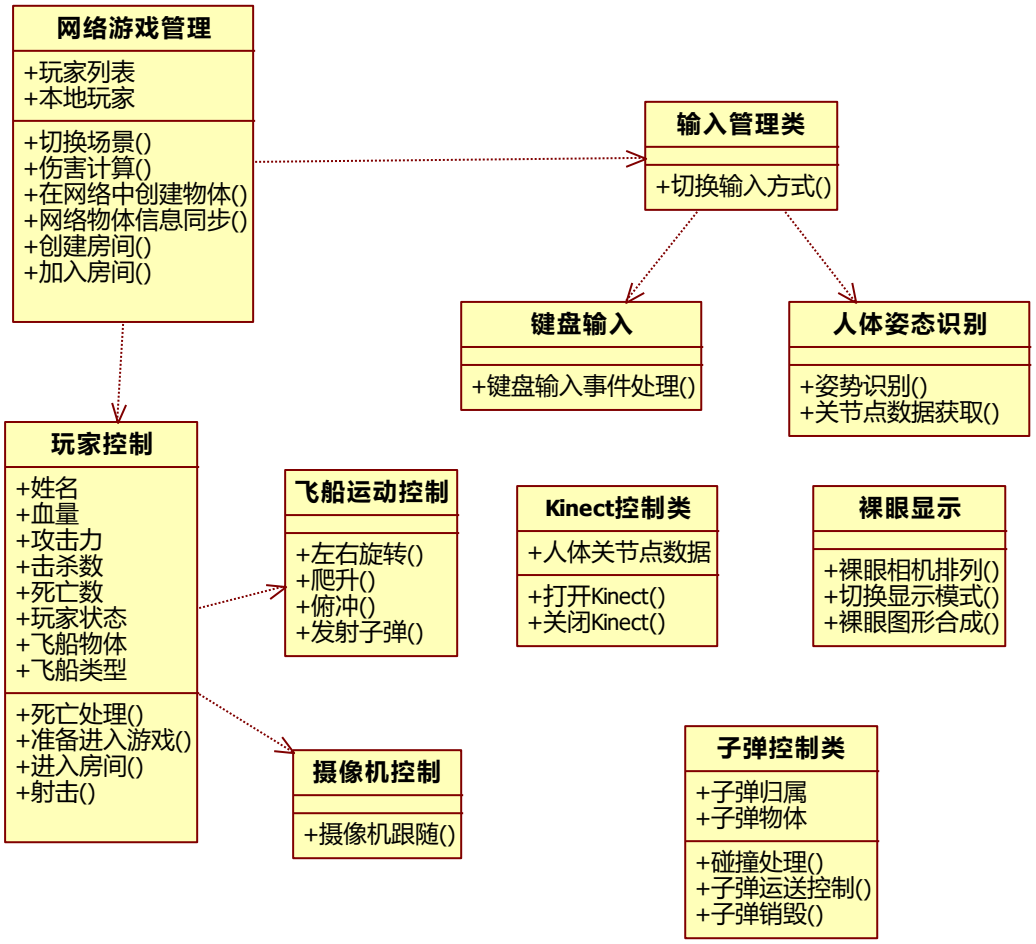


图 2-5 类间的关系 2

2.9 本章小结

为有效地避免游戏架构失误，保证游戏的质量，本章首先描述了游戏的规则，然后进行了非功能和功能需求分析，提出了本系统必须实现的功能点和非功能要求，接着根据游戏规则提出了游戏的业务流程，然后对游戏进行用例分析，通过用例图分析出了游戏的功能模块，最后根据功能模块分析出了游戏的类图。本章对系统的需求进行了详细的分析，为之后的系统设计和开发做好了准备。

第 3 章 裸眼 3D 的技术基础

在了解裸眼 3D 屏幕是如何呈现立体影像之前，有必要先了解人类在生产活动中是通过什么途径来获得物体的深度感与如何判断出观测物体的远近这两个问题。一般而言是由多个深度线索组合来正确的判断出空间中的物体相对位置，深度线索包含了双眼视差、人眼的调节性，移动视差、透视、观测物体间大小关系、物体材质等。其中以双眼视差较能正确的判断出深度信息，双眼视差原理是由于双眼间隔 65mm[15]，因此两眼说看到的影像会稍微不同，此两眼所接受到的影像经由脑部融合而产生 3D 深度信息[16]。本章将从立体视觉基本概念开始讲述，然后介绍基于双眼视差原理的裸眼 3D 技术原理。

3.1 人眼立体视觉形成原理

早在 1839 年，英国著名的科学 家温特斯顿就在思考一个问题——“人类观察到的世界为什么是立体的？”进过一系列研究发现：因为人长着两只眼睛。人双眼大约相隔 6.5 厘米，观察物体(如一排重叠的保龄球瓶)时，两只眼睛从不同的位置和角度注视着物体，左眼看到左侧，右眼看到右侧。这排球瓶同时在视网膜上成像，而人类的大脑可以通过对比这两副不同的“影像”自动区分出物体的距离远近，从而产生强烈的立体感。引起这种立体感觉的效应叫做“视觉位移”。用两只眼睛同时观察一个物体时物体上每一点对两只眼睛都有一个张角。物体离双眼越近，其上每一点对双眼的张角越大，视差位移也越大，如图 3-1 所示。

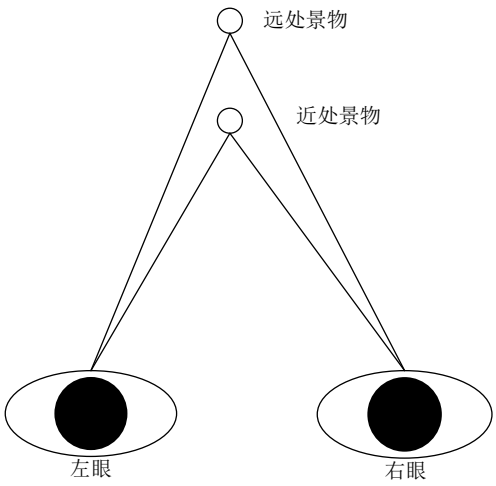


图 3-1 人眼视差图

正是这种视差位移，使人能区别物体的远近，并获得有深度的立体感。对于远离的物体，两眼的视线几乎是平行的，视差位移接近于零，所以很难判断这个物体的距离，更不会对它产生立体感觉了，夜望星空会感觉到天上所有的星星似乎都在同一球面上，分不清远近，这就是视差位移为零造成的结果。当然，只有一只眼的话，也就无所谓视差位移了，其结果也是无法产生立体感。例如，闭上一只眼睛去做穿针引线的细活，往往看上去好像线已经穿过针孔了，其实是从边上过去的，并没有穿进去。而现在所看到的图片、电影、玩的游戏都是平面景物，虽然图像效果非常逼真，但由于双眼看到的图像完全相同，自然就没有立体感可言。如果要从一幅平面的图像中获得立体感，那么这幅平面的图像中就必须包含具有一定视差的两幅图像的信息，再通过适当的方法和工具分别传送到左右眼睛。

3.2 裸眼式 3D 显示技术

裸眼立体显示技术就是指在不需要任何辅助设备观看就能够获得立体视觉效果的立体显示技术，其主要是通过使用光学器件在显示屏幕上改变双眼视图的走向，使人的双眼能分别看到对应的立体视图。因为具有不固定的图像通道及图像分割装置，一般称这个实现立体显示的方法称之为空分法。裸眼 3D 显示都运用了上节所说的双眼视差位移的原理来产生立体视觉，实现技术主要有光屏障技术、柱状透镜技术和指向光源技术三种[17]。柱状透镜式裸眼 3D 技术因其亮度不受影响而得到广泛研究，下面将会详细介绍柱状透镜式裸眼 3D 成像原理。

3.2.1 柱状透镜式裸眼 3D 原理

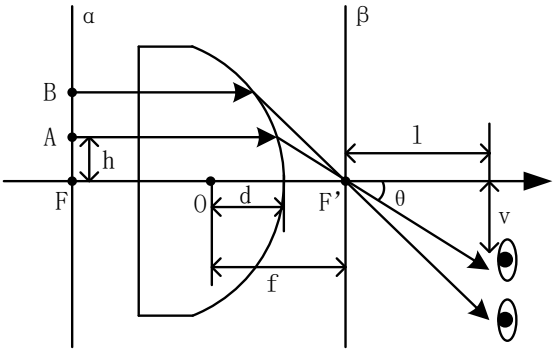


图 3-2 平凸透镜光路图

柱状透镜 3D 技术的原理是在液晶显示屏的前面加上一层柱状透镜，该透镜称为柱状光栅，使液晶屏的像平面位于透镜的焦平面上，这样在每个柱透镜下面的图像的

像素被分成几个子像素，这样透镜就能以不同的方向投影每个子像素。于是双眼从不同的角度观看显示屏，就看到不同的子像素。因为柱状透镜使用的是平凸透镜，其光学特性可以描述为下图 3-2 所示。

图 3-2 中透镜光轴为 x 轴方向，F 和 F' 为第一、第二焦点，O 为透镜光心，α 为第一焦平面，β 为第二焦平面，透镜焦距为 f，光心与透镜顶点的距离为 d。设像素点 A 与光轴距离为 h，经透镜后折射光线进入人眼，人眼与透镜第二焦平面距离为 l，与主光轴距离为 v，设光线经透镜出射点在光轴上的投影距离顶点为 δ，根据几何关系有公式 3.1、3.2。

$$\delta = r - \sqrt{r^2 - h^2} \tag{3.1}$$

$$\tan\theta = \frac{h}{f - d + \delta} = \frac{\mathrm{v}}{l} \tag{3.2}$$

由此推导出公式 3-3。

$$\frac{v}{l} = \frac{h}{f - d + r - \sqrt{r^2 - h^2}} \tag{3.3}$$

公式 3.3 中 f,d,r 都是常数， v 与 l 的比值只与像素点与主光轴的距离 有关，即 的坐标关系由 决定，随着 的变化，像素点光线的传播路线也在发生改变，可观看到该点像素图像的位置 也发生改变。屏幕像素点位于透镜焦平面上，根据凸透镜光学特性，不同位置的像素点经过透镜后将会发射出不同方向的光线。

3.2.2 多视点裸眼 3D 技术

由多条平行放置的柱状透镜构成了裸眼 3D 透镜屏幕，透镜焦平面上的像素发光点经过透镜折射后在空间形成固定的可见区域，从而将左眼图像与右眼图像分开。在一个透镜宽度后面放置多个像素点光源就可以在空间多个位置形成不同角度的视点画面，观看者在不同的观看位置只要同时看到这多个视角画面中的两个就可以感受到深度信息，这种技术被称为多视点技术[18]。多视点示意图如图 3-3 所示。

从图 3-3 可以看出，对于柱状透镜光栅焦平面上的像素点，其发出的光线经过透镜后在空间形成固定位置的可视点，如图上形成的 4 个视点。只要人的左右眼睛分别接收到两个不同视点的图像就可以产生裸眼效果。

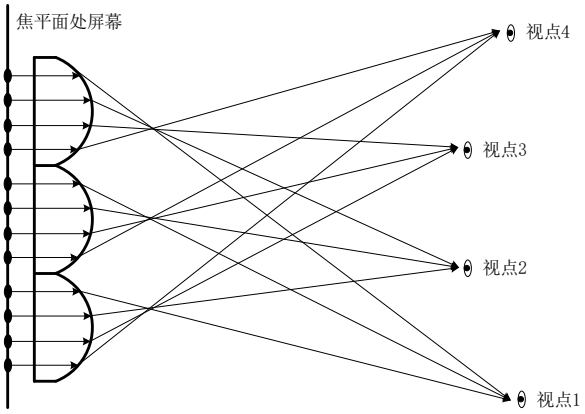


图 3-3 柱透镜光栅成像

对人眼立体感的产生贡献最大的是水平视差[19]，为了得到较好的 3D 效果，往往采用多幅具有水平视差的图像合成图像，而在竖直方向上不做多视点处理。多视点技术提供了自由的观看位置与角度，同时因为裸眼视差都是在水平方向，所以人眼实际接收到的图像分辨率会在水平方向严重降低而竖直方向没有变化，导致分辨率失调。缓解分辨率失调的问题可以将柱状透镜倾斜放置，降低的分辨率被分摊到水平和竖直方向，从而人眼接收到的分辨率不至于太低。

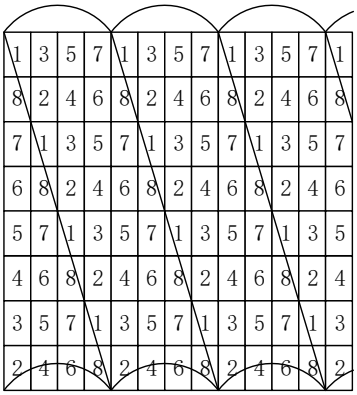


图 3-4 8 视点裸眼透镜摆放与像素索引排列

以图 3-4 所示的 8 视点裸眼屏幕为例，将柱状透镜倾斜放置，与水平面成一定夹角，该夹角由像素点物理宽高决定，屏幕像素的排列也交错放置，在透镜倾斜方向依次排列 1 到 8 视点对应的像素，每个透镜宽度区域内包含了完整的 8 个视点的像素点。经过倾斜放置，可以计算得到水平方向分辨率变成原图像的 1/4，垂直方向分辨率变为原来的 1/2。存在一种坏的情况，当人左眼看到视点 8 的图像，右眼看到视点 2 的图像，根据 8 视点排列原理，此时画面左右颠倒，同时模拟两眼的相机间距最大，好比于眼睛瞳距过大，人眼很难聚焦因而无法获得立体图像。

3.3 本章小结

本章首先简要介绍了人类形成立体视觉的原理，然后概述了基于视差位移原理的裸眼 3D 技术，并单独介绍了柱状透镜式裸眼 3D 原理。最后，通过柱状透镜式展开介绍了多视点裸眼 3D 技术。本章介绍的裸眼 3D 原理仅仅停留在理论表层，并未做深入研究，是因为本论文是讨论裸眼 3D 技术的应用研究，深入研究原理已经超出研究范畴，本章只是为后面在 Unity 中实现裸眼效果提供理论支持。具体如何在游戏中运用裸眼 3D 将在后续章节中介绍。

第 4 章 体感游戏的技术基础

体感游戏突破了传统电子游戏需要通过鼠标键盘或者游戏手柄按键对游戏设备的操作，直接通过摄像头捕捉玩家的动作，识别和分析肢体动作或手势作为游戏的输入，从而使玩家能够直接和系统进行交互，实现前所未有的交互体验，不仅能够让用户直接与系统进行“会话”，还能够在轻松愉快的游戏体验当中达到锻炼身体的目的，自上市以来就成为一种促进身心健康的互动娱乐方式大受推崇[20]。目前开发体感游戏硬件设备方面一般采用微软的 Kinect 传感器，游戏引擎上一般采用发展较为成熟的 3D 引擎，如 Unity，Bullet 等。3D 建模与一般的 3D 游戏要求并无特殊差异，一般采用 3DMax、Maya、Blender 等建模工具。游戏的内容设计上，3D 的体感游戏需要在特定的 3D 的场景中完成，一般的体感游戏的框架结构如图 4-1。

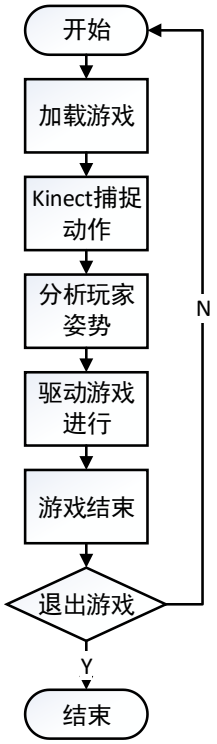


图 4-1 体感游戏框架结构

尽管体感游戏的开发在普通游戏的基础上增加了传感器接入以及硬件调试的步骤，但是就游戏开发本身而言，体感游戏的开发流程与开发方法与普通游戏并没有太大差别。因此，以下部分主要针对通过 Unity 开发 3D 游戏的一些方法与技术点进行讨论和分析。随着游戏类型的不同，所包含的子系统也不尽相同，但是在开发过程中还是都有一些共性。需要注意的是，这并非针对于商业游戏只是通用游戏的开发流程，

因为涉及到游戏运营的问题，作为产品的游戏即便是在游戏推广阶段也是分步骤、分情况的。Unity3D 游戏基本开发流程如图 4-2 所示。

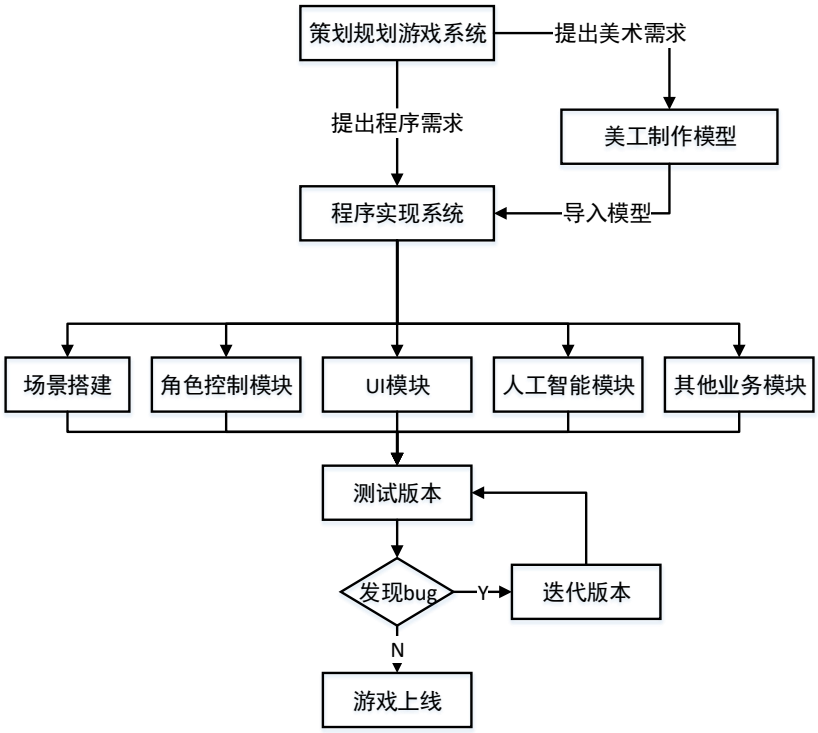


图 4-2 3D 游戏的基本开发流程

4.1 游戏开发的组织分工

从上一节介绍的游戏开发流程可以以看到，游戏的画面依赖于大量的美术资源，游戏可玩性依赖于关键性的玩法设计和数值系统，这也是其不同于其他软件开发最大的不同。由于游戏开发涉及到不同专业领域的技术知识，因此往往一个能相互协调的团队显得异常重要。一个独立完整的游戏项目，一般需要 6 大类人员构成[21]，简述其相应的职责：

（1）制作人：全面掌握了从研发到运营整个体系知识的游戏项目的总负责人，把控游戏开发的进度以及游戏产品的质量把控；

（2）游戏的编码人员：根据游戏类型的不同，角色构成可能存在差别，如果是网络游戏，则要细分为服务器端和客户端，并且各个端都需要相应的核心程序员去架构游戏系统，即主程序员；

（3）游戏美工人员：相对于普通软件开发项目，游戏开发中对界面美化的要相对更高，因为游戏中涉及大量的美术资源，如何“占用更少的内存去做更多更棒的事”，

也使得美工在游戏开发当中担当举足轻重的角色；

（4）游戏策划：以创建者和维护者的身份参与到游戏的世界，将想法和设计传递给程序和美术。设计游戏世界中的角色，并赋予他们性格和灵魂。在游戏世界中添加各种有趣的故事和事件，丰富整个游戏世界的内容。 调节游戏中的变量和数值，使游戏世界平衡稳定，制作丰富多彩的游戏技能和战斗系统，设计前人没有想过的游戏玩法和系统，带给玩家前所未有的快乐。

（5）游戏测试：测试游戏，寻找错误和缺陷。在问题跟踪数据库中编写测试报告，执行测试流程。编写编辑测试案例，对问题处理情况进行查证，监控活跃事件。提供游戏趣味性及平衡性的反馈意见，负责控制游戏整体质量，同其他团队成员就有关测试问题进行交流与合作。

（6）游戏运营：全方位了解游戏开发本身的技术优势、适用平台、题材、IP、核心玩法、开发进度计划、收费点设计、收费方式、竞争对手情况，还要根据游戏测试期间获得的信息，和媒介、市场、渠道的同事开会，讨论出游戏的宣传点和市场投放计划，在游戏上线后关注下载和注册登录流程，考虑如何留住更多的用户。游戏进入平稳运营期要关心每一个版本更新和每一次活动的用户反馈，出现问题第一时间解决，必要的时候对用户做补偿。关注游戏的市场数据并实时调整修缮，保持游戏的市场活力；

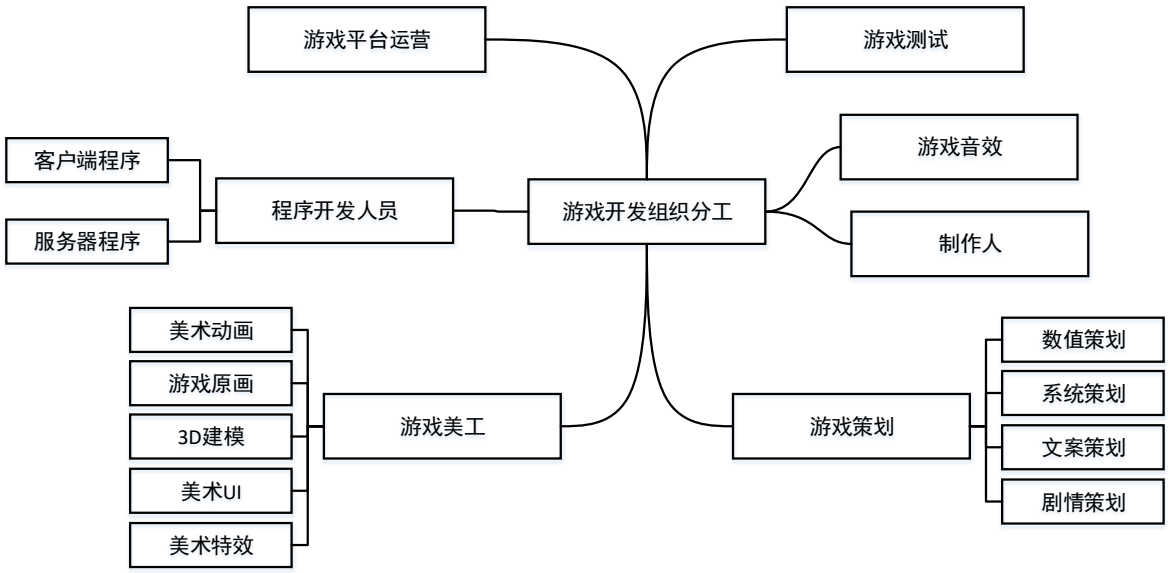


图 4-3 3D 游戏开发的详细组织分工

以上简要概括了游戏开发过程中各个角色的主要职能，一般来讲，一个游戏往往可能包含多个复杂的系统。详细来说，会有更多的角色需要参与到游戏开发当中，图4-3 为游戏开发的详细组织分工。从图中能够看出，游戏的开发并非一件易事，它需要诸多人员的协调配合，才能够完成一款精品游戏。

4.2 体感游戏开发的技术要点

体感游戏的开发即要考虑 Kinect 的运行环境，也需要考虑 Unity 的运行环境，幸运的是 Unity 采用跨平台的.NET 运行环境 Mono 作为运行环境，可以通过动态链接库的方式封装好 Kinect 接口提供给 Unity 使用。在介绍体感技术前，先介绍 Unity 相关技术。

4.2.1 Unity 开发中组件的编写

Unity 的核心就是一切皆 Component。在传统的结构设计中一般会使用“派生”来描述对象之间的关系。子类通过派生父类，来获得父类的功能。在设计游戏对象时，会根据游戏本身的需要而为游戏对象添加各种功能支持，比如渲染，碰撞，刚体，粒子系统等等。这些通用功能为了能够为各种派生类提供服务，都必须实现到基类中。这样就导致了游戏对象基类变得非常庞大臃肿，即难使用，又难维护。基于组件的对象模型就是把所有需要提供给游戏对象的基础功能都独立成单独的组件模块(Component)，一个具体的游戏对象可以将它需要的功能模块组合到一起使用。所有功能不再是父类中的接口，而变成子对象实例，为游戏对象提供服务。这样既保证了功能代码的可重用性，又增加了整个对象体系的模块化和灵活度。在 Unity 中，GameObject 除了作为 Component 的容器之外，基本上没有其他功能。所有需要的功能都要通过组合 Component 来实现。脚本本身也是 Component，用来在 GameObject 上通过控制其他 Component 来实现自定义的功能。

默认的脚本对象都继承自 MonoBehaviour 类。在 MonoBehaviour 类中，有很多重要的方法，按照执行调度的次序依次为 Awake()，Start()，Update()，LateUpdate()，FixedUpdate()，OnTriggerEnter()，Disable()等等，其中，Awake 最先被执行，用于做一些初始化的准备工作，并且 Awake 与 Start 都只会被执行一次。Update 是在 Start 被调用之后、每一帧渲染之前被执行，Update 中如果做太多无意义的操作则会很耗性能，降低帧率，因此不适合用于处理计算量大和逻辑复杂的操作。而 LateUpdate 主要用于渲染后处理。而相对的，FixedUpdate 每个物理时间同步一次，具体的时间

间隔取决于所处的平台，因此可用来处理一些实时性要求一般的事件。OnTriggerEnter 则在需要碰撞触发的时候被调用，常被用作碰撞检测[22]。最后 Disable 在对象被销毁时调用。

4.2.2 位移和旋转

在 3D 游戏当中，位移与旋转是最基本、核心的要求之一。在 Unity 中直接或间接与位移和旋转变换相关联的组件主要有：Transform 组件、RigidBody 组件、NavMeshAgent 组件、CharacterController 组件等等。各大组件见有功能重合部分，但又有细微的区别，如果不能够理清他们的关系，很难很好的控制物体的运动。表 4-1 列出了位移组件间的区别。

表4-1 Unity 中位移相关组件

组件	函数/属性	描述
Transform 组件	Translate 函数	向某方向移动物体多少距离或者相对某物体移动
	Position 属性	在世界空间坐标 Transform 的位置
RigidBody 组件	Velocity 属性	刚体的速度向量
	AddForce 函数	添加一个力到刚体。作为结果刚体将开始移动。
	MovePosition 函数	移动刚体到指定点
NavMeshAgent 组件	SetDestination 函数	设置自动寻路的目标点
CharacterController 组件	Move 函数	一个更加复杂的运动函数，每次都绝对运动
	SimpleMove 函数	以一定的速度移动角色

旋转在 3D 中是比较复杂的，一般描述旋转有三种方式：旋转矩阵、四元数和欧拉角[23]。其中四元数与旋转矩阵相比其只由 4 个数组成，而旋转矩阵需要 9 个数，四元数占用内存小于旋转矩阵，并且四元数不会有欧拉角的万向锁问题。除以上几点之外，四元数更容易插值，更容易做规范化和正交化。但四元数也有其缺点，四元数是从矩阵变换而来的，直接操作四元数抽象性较高。在 Unity 中，提供了非常简洁的 API 操作旋转，屏蔽了底层的复杂数学，四元数在 Unity 中的操作全部被封装在 Quaternion 类中，表 4-2 列举了 Quaternion 类中几个常用的 API。

表4-2 Unity 中旋转相关函数

函数	描述
LookRotation（Vector3 forward, Vector3 upwards）	创建一个旋转，沿着 forward（z 轴）并且头部沿着 upwards（y 轴）的约束注视。也就是建立一个旋转，使 z 轴朝向 y 轴朝向 up。

Angle（Quaternion a, Quaternion b）	返回传入的两个旋转之间的角度。
Euler(float x, float y, float z)	返回一个旋转角度，绕 z 轴旋转 z 度，绕 x 轴旋转 x 度，绕 y 轴旋转 y 度
Slerp(Quaternion a, Quaternion b, float t)	球形插值，通过 t 值 from 向 to 之间插值。
FromToRotation(Vector3 fromDirection, Vector3 toDirection)	从 fromDirection 到 toDirection 创建一个旋转。
Quaternion.identity	返回恒等式旋转（只读）。这个四元数对于“无旋转”：这个物体完全对齐于世界或父轴。
Quaternion.operator *	由另一个四元数来旋转一个旋转角度，或由一个旋转角度来旋转一个向量

4.2.3 协程的原理

从程序结构的角度来讲，协程是一个有限状态机，说到协程，首先需要提到另一样东西，那就是子例程，子例程一般可以指函数，函数是没有状态，函数返回之后，函数的所有局部变量会被释放内存，但是在协程中可以在一个函数里多次返回，局部变量被当作状态保存在协程函数中，直到最后一次返回，协程的状态才被清除。简单来说，协程就是：一段顺序的代码，标明哪里需要暂停，然后在下一帧或者一段时间后，系统会继续执行这段代码。本质上，协程和多线程根本不是一个概念，但是可以通过协程的使用，来实现宏观上程序中数据的同步。虽然 C#提供了对多线程的完整支持，但 Unity 引擎是主循环结构，逻辑更新和画面更新的时间点要求有确定性，如果在逻辑更新和画面更新中引入多线程，就需要做同步而这加大了开发难度，因此在默认的对象组件脚本类中想要达到多线程的效果可以通过协程这样“伪同步”的方式来实现[24]。

在 Unity 当中，可以使用 StartCoroutine()来开启一个协程，传入的参数直接为函数名，如果想要实现延迟操作，就必须使用 IEnumerator 类型的返回。C#中可以使用 yield return 来推迟程序的运行。如 yield return new WaitForSeconds(1f)，并非直接间隔 1 秒钟之后程序才向下执行，它依然是异步的。只是到了下一个一秒钟程序直接返回迭代器而已。因此有了协程，可以大大简化程序，不需要将所有的刷新等待操作都放到 Update()当中进行，每一帧进行的多余操作无疑都是性能上的浪费。

4.2.4 游戏对象间通信

Unity 开发过程中两个对象见的通信其实往往是挂载在两个对象身上的脚本进行通信、信息传递。一般可以主要概括为以下的几种方法：通过事件的监听处理；通过

GameObject 的 SendMessage 方法向另外一个脚本传递消息；通过对象脚本公开的静态方法或属性，向其他对象提供需要的数据信息，信息的关注者在感兴趣的时间监听对象的属性或调用公开的静态方法。在具有一定规模的项目中，会涉及到大量的对象通信的情况。一般通过事件的发布和接收能够有效、有逻辑的管理多组对象间的通信，也能够处理一对多、多对多的对象通信[25]，但是为了配合第三章游戏美术设计的小节将对象通信和 UI 设计结合起来，此小节就暂不对事件作详细阐述，主要阐述通过 SendMessage 方法和共有静态成员的方法。

1. SendMessage 方法进行对象通信

Unity 中通过 SendMessage 开进行对象间的通信是最简单也是最基本的方法。基本的流程是消息的发送对象脚本中找到消息的接收方对象，调用接收对象的 SendMessage 成员方法，向其传入接收消息的方法名和参数数据。在消息接收对象绑定的脚本中添加同名的带参方法，获取到的参数值即为发送方传入的参数，可以做出相应的响应工作。

2. 访问公有静态属性的方法

访问公有静态成员属于较为直接的一种对象通信方法，即不同对象身上的脚本通过查看其他对象脚本中的静态成员数据来直接进行消息传递。这种方式较为简单直接，可以应用于逻辑处理简单、数据相对不复杂的地方。但是缺点在于使用公有静态成员很难保证数据的安全，如可修改的数据可能会被多处修改造成逻辑混乱，因为无法使用动态成员（使用会新实例化出一个对象脚本出来），频繁的外部修改可能会造成数据不一致的情况。

4.3 游戏美术及动画的主要制作方法

游戏美术前前后后需要涉及到多人参与和角色分工。从 3D 建模，2D 原画，美术的 UI，到美术的动画制作及画面的特效表现，在实际项目当中往往需要多人专项负责。因此我也无法做全面专业的讨论。在此主要讨论游戏美术制作的基本流程以及我在游戏当中担当的角色部分：美术的 UI、动画、特效控制等上层的美术制作以及在 Unity 一些制作的主要方法。

4.3.1 游戏美术的基本制作流程

游戏美术在制作的过程中，主要包含的内容有 3D 建模、2D 原画、动作的制作、美术 UI 的制作等几个部分，并按照策划的需求定义有组织、有步骤的完成资源的准备，最终会将资源交付到游戏引擎中进一步进行程序开发。实际开发中还会根据美术在游戏引擎中呈现的效果反馈进行资源的休整。游戏美术的基本制作流程，如图 4-5。

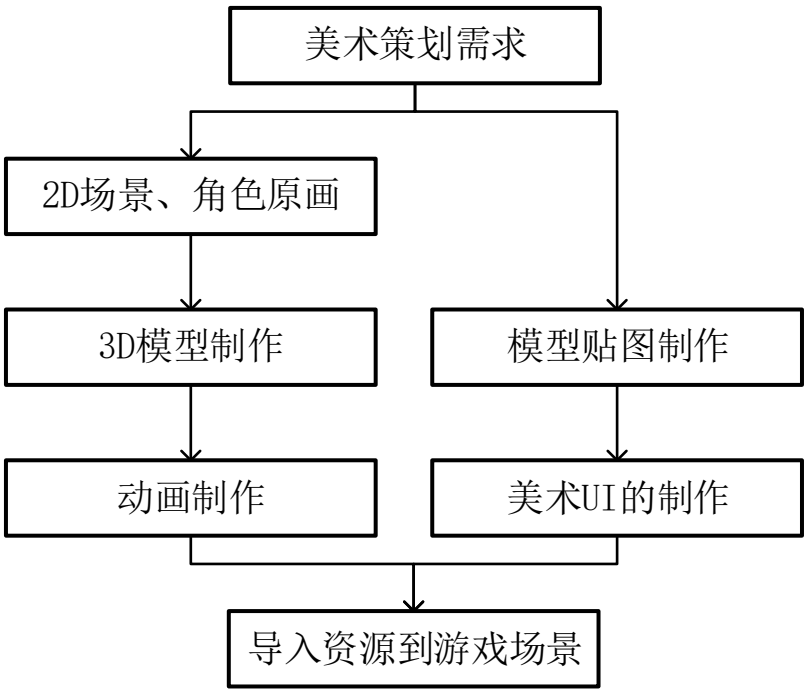


图 4-5 游戏美术的基本制作流程

4.3.2 游戏动画的制作方法

在 Unity 当中，动画的制作方式主要分为两种：序列帧动画，骨骼动画[26]，此外还有在插件当中出现的动画，如 ITween 缓动动画。其实一般主要讨论前面两种方式，因为缓动动画常常只是作为方便制作使用小动画组件出现。

（1） 序列帧动画 序列帧动画往往由一系列的图片构成，为 2D 的图片动画。制作方式较为传统， 动画制作的表现效果主要取决于动画的帧数和图片资源的质量，与原模型资源无关。一般适用于 2D 游戏，作为 Animation 组件的一部分被使用。但是在 Unity 4.2 版本之后，就不能够将新制作的动画 AnimationClip（动画片段）添加到 Animation 组件中被控制播放了。需要添加到 Animator 组件当中作为一个动画状态出现，而之前版本中制作好的动画依然可以作为 Animation 组件的成员出现，单独被控制。通过 Animation 控制动画的播放，好处在于可以直接进行动画片段的方便使用。

但是另一方面，采用序列帧动画，虽然可以补充低模（面数较低的模型）来带的质量不足，但是容易占用较大的内存，动画播放时资源被一次性载入，容易造成运行时帧速率的降低。此外，制作复杂和平滑度要求高的动画片段也需要投入更多的精力和成本。图 4-6 为一个简单特效的序列帧动画片段（Clip）。

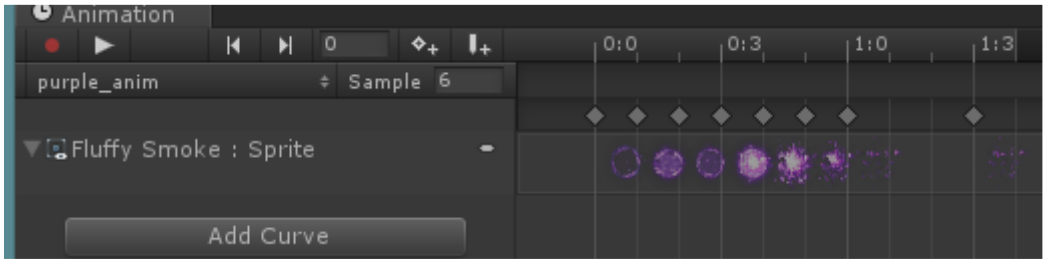


图 4-6 序列帧动画片段（Clip）

（2） 骨骼动画

骨骼动画是相对区别于序列帧动画而存在的。较为特殊的是动画片段在 Animator 组件中是作为一个个独立的状态而存在的。单独的动画状态制作的时候与 Animation 的 Clip 并没有特别之处。但是在制作片段的时候，如果对象是已经绑定了骨骼的模型，那么在制作动画的时候会默认呈现所有的骨骼节点，便与制作动画[27]。

但是骨骼动画相对于序列帧动画较为繁琐的一点是：动画片作为动画控制器的元素出现。想要实现动画状态的切换必须要通过动画状态机完成。状态间的切换通过传入的参数作为切换的过度条件，这样做的好处在于能够方便的管理动画状态，动画片段的逻辑关系较为清晰。但复杂之处在于当要切换的动画状态较多的时候，动画状态及的制作需要更多考究。

4.4 游戏策划的职责与使命

游戏策划在一款游戏当中作为游戏内容的设定者，游戏剧情的编剧，数值的策划者，往往扮演着一款游戏的灵魂人物的角色，好的游戏策划一般也被称为游戏制作人，将游戏真正想要展现的表达出来。

进一步而言，一款游戏能否做的足够吸引人，在系统框架搭建的时候就能够体现出游戏的核心系统和核心玩法，在各个子系统的策划案中就已经能够看出端倪了。策划在设定游戏内容的时候就已经决定了游戏是否较为有趣、有较高的可玩性。因此一份好的策划案也往往决定了一个游戏是否有希望，甚至游戏的生命线能有多长。

4.5 体感游戏的 SDK 接入准备

在体感游戏的接入准备工作方面，一般涉及到以下一些需要准备的方面：

1. Kinect 开发环境的搭建和配置，以及了解基本的 Kinect 开发知识：了解 Kinect 开发的基本流程以及能够通过 Kinect 获取数据等，了解设备和程序异常的应对处理；

2. Kinect 结合 Unity 开发的时候，由于 Unity 无法直接获取 Kinect 传入的数据，需要学习和了解中间件的使用，必要的时候还可能要开发自己的中间件来从 Unity 获取 Kinect 的传感器数据；

3. 在进行开发之前预留处理 Kinect 数据的借口和做输入控制的设计，便于逻辑处理和后期程序维护。

4.6 本章小结

本章主要介绍了通过 Unity3D 开发体感游戏所需要的技术基础，主要从游戏开发的组织分工、游戏开发的技术要点、游戏美术及动画的主要制作方法以及游戏策划在游戏当中担当的指责使命四个方面进行了介绍和概述。目前的游戏制作当中，由于涉及到的技术点较多，本章只作了重点介绍，除了考验编程技巧的脚本编写具体的技术、牵扯较为复杂的运动控制以及大量的浮点向量运算，以及种类较多的动画制作方法，更多的将在第三重游戏的系统方案设计当中体现。尽管 Unity 提供了多种控制运动和动画制作的方法，但是具体采取哪种方式还要依现有的资源情况以及适用性而定。

第 5 章 裸眼 3D 体感游戏系统设计

本系统主要是将 Kinect 接入到一个用 Unity 制作的局域网联机飞行射击游戏中，并将使用裸眼 3D 算法将游戏画面用特殊的裸眼屏幕展示。因为游戏地图的容量限制，同时考虑到可玩性，最多可四个人通过局域网联机游戏，玩家通过特定的肢体动作操作飞船 360 度飞行，通过手势控制飞船发射激光炮击杀敌人玩法采取个人竞技模式，其他人都是你的敌人。当某一玩家击杀超过 3 人后，游戏结束。由于肢体动作的反应速度和对鼠标键盘的操作速度相比较慢，所以游戏总体来讲比较考验用户的应变反应能力和操作能力。为了让用户充分体验游戏过程中的健身性和娱乐性，不设定撞击障碍物死亡的情况，而是放大击杀成就，从另一方面激励用户。

5.1 总体框架设计

本系统的总体设计结构主要分为三层：应用层，网络层以及 Kinect 的硬件接口层。其中应用层为用户交互层，用户通过预定义的肢体动作、手势和系统进行交互，总体框架设计如图 5-1。

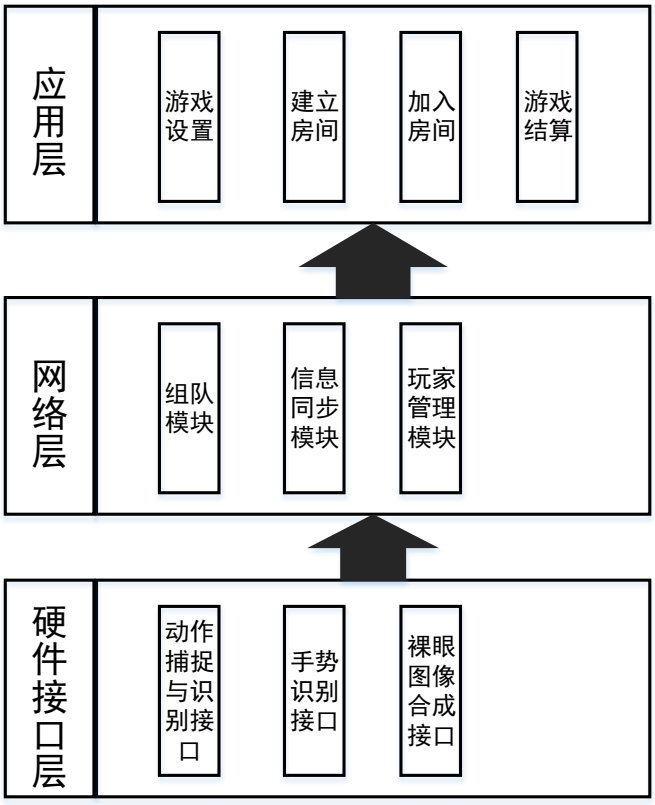


图 5-1 游戏总体框架设计图

其中，用户与系统的所有交互操作出现在应用层。系统的总体实现流程如下：

（1） 启动游戏，初始化 Kinect 模块，根据是否初始化成功选择游戏操作方式，根据游戏设置选择是否打开裸眼模式，进入游戏菜单的主界面；

（2） 用户选择创建房间或者加入房间，进入房间后，选择进入准备状态，当房间内所有人进入准备状态后，进入游戏场景；

（3） 进入游戏场景后，飞船开始自动飞行，用户通过偏转身体方式实现对飞船的方向控制，通过握拳攻击其他飞船。当被其他玩家的激光炮击中后会减少血量，血量小于等于零则判定为被击杀；

（4） 在游戏运行过程中，依然可以进行菜单操作，可以选择关闭背景音乐或者退出游戏，但是游戏并不会暂停；

（5） 直到有人击杀数达到 3 人，游戏结束，展示结算列表。

该系统涉及到的几个主要功能：飞船的运动控制，游戏设置，建立以及加入房间等,但是在开发的时候为了更系统性的管理控制模块,主要的设计分为以下几个部分：场景设计及搭建，游戏美术的设计，飞船控制系统，Kinect 接入，裸眼模块，网络通信模块等几个模块。

场景设计及搭建部分，主要包含了室内的密闭场景和四种飞船模型。导入 Unity 的 3D 模型格式一般为 fbx，其他格式一般需要通过 3DMax、Maya、Blender 等建模软件导出为 fbx 格式比较好。贴图方面，Unity 常用的贴图格式一般为 dds、tga、png、jpg 等等，若遇到不支持的贴图格式则需要转换之后才可，否则在编辑环境中则会呈现模型贴图丢失。

游戏美术方面，该系统主要设计的美术制作内容为游戏中出现的动画、特效、美术 UI 设计以及用户交互的响应部分。

飞船控制系统是该系统中的核心模块，直接管理着与游戏玩法相关的角色控制部分，主要讨论其中包含的主要控制方法。

Kinect SDK 的接入部分为游戏最后的收尾工作，接入和调试直接决定着最终的用户体验是否良好，系统的稳定性是否足够高。

裸眼模块是将游戏画面合成裸眼屏幕能展示的图像，其中涉及到着色器（Shader）的编写，是整个系统的技术难点，其直接影响着最后在裸眼屏幕上的显示效果。

网络通信模块是系统中最核心的模块，是整个系统的基础，直接决定了游戏的稳定性，同时，由于 Unity 的网络组件在 Unity5.1 中才正式推出，并没有完善，也没有完整的文档和社区支持，是整个系统中开发难度最大的部分。

以下小节将从上述的几个模块出发，分别阐述系统的设计过程，图 5-2 为该系统的总体流程图。

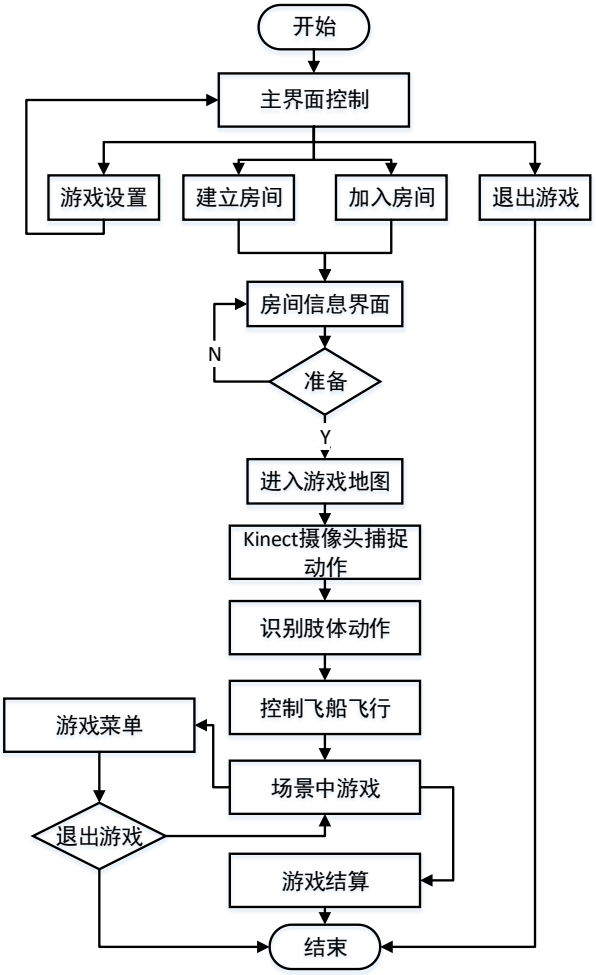


图 5-2 系统总体流程

5.2 游戏美术设计

游戏美术在整个游戏系统中起着至关重要的作用。可以说，一款游戏不管有多么完美的逻辑和操作，但是没有一个颇具风格、特色的好美术的话，依然是不完整的，在用户体验上也是受限的。因此一直以来游戏中的美术都被作为一个不断追求和突破的技术点参与到游戏开发当中，甚至很多时候，美工和程序同样可以实现的效果，会尽量选择二者更优更高效的实现方案。

该系统中主要包含的美术制作方面分别为美术资源准备、游戏的场景设计、特效控制部分、UI 的设计以、场景灯光的渲染及优化四个部分。本小节主要围绕这几个角度进行阐述。

5.2.1 美术资源准备

目前 3D 建模主要采用的两种软件工具为 3DMax 和 Maya。一般 Unity3D 需要的 fbx 模型格式主要是由 3DMax 导出的。在导出的时候，可以一并导出模型中包含的贴图资源。不过需要注意的是，贴图资源尽可能用英文命名，避免中文命名造成的乱码麻烦。倒入的模型资源虽然可以当作对象拖动使用，还是尽量将需要用的模型做成预设便于使用和管理。

5.2.1 游戏场景设计

为了支撑游戏玩法，避免因为游戏场景太大，无法找到玩家的位置，游戏场景设计为一个封闭的室内，如图 5-3。为了优化性能，将整个静止的场景都标记为静态（static），在 Unity 中静态物体在脚本加载的时候就开始渲染，不随着 update 每帧更新，并且在渲染时，Unity 会通过静态批处理（Static Batching），将标记为静态的物体组成 Batch，在一次调用 GPU 渲染中批量处理多个物体，提高渲染效率。

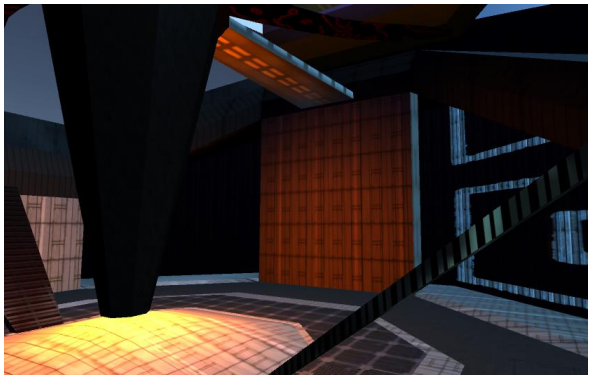


图 5-3 游戏主场景

5.2.2 粒子效果实现

粒子系统（ParticleSystem）是通过一个粒子发射器、粒子动画片段、粒子渲染等组件来实现粒子的播放控制、从而实现有动画特效的美术效果。粒子系统在场景中一般用来实现瀑布、火焰、烟雾自然效果以及游戏中的特效等。在本系统中，包含大量具有粒子效果的游戏元素。图 5-4 中为飞船引擎喷出的火焰。

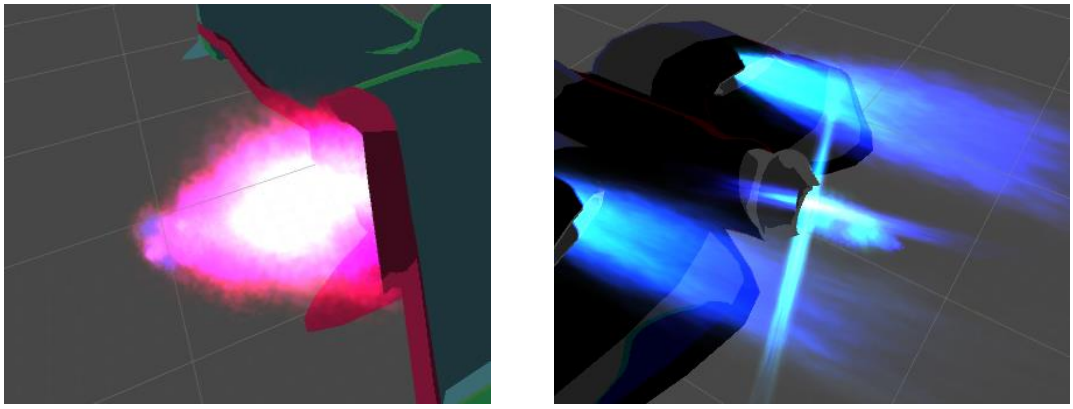


图 5-4 飞船引擎喷出火焰效果

5.2.3 UI 设计及用户交互

本系统中的 UI 设计全部采用 Unity 自带的 UGUI 制作。系统中的 UI 主要包含进入游戏的开始菜单、游戏设置界面、房间信息界面、运行主界面、玩家列表界面和游戏结算界面 6 个界面。首先设计的是 UI 跳转的逻辑设计部分，主要定义了游戏中包含的主要界面。以及每个界面呈现上的核心信息，图 5-5 为 UI 设计逻辑图。

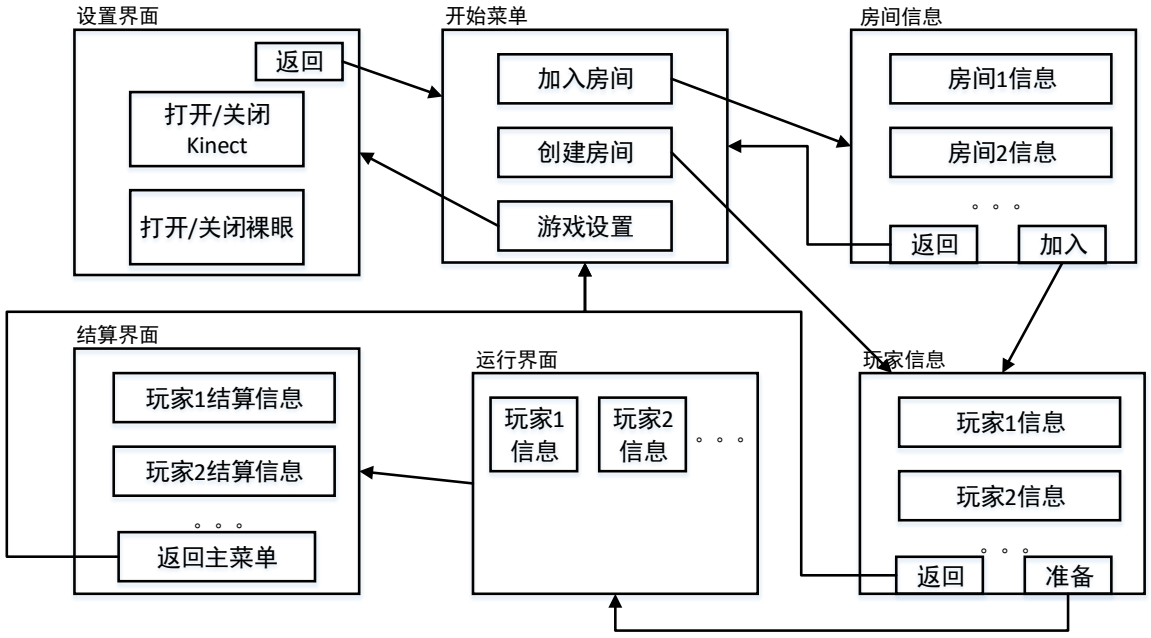


图 5-5 UI 设计逻辑图

界面刷新处理上，主要包含 UI 界面的用户响应、游戏数据刷新等两个方面，以下主要阐述 UI 用户响应及数据刷新的主要方法。

（1）委托及事件响应

UI 操作上的用户响应一般通过事件来完成。事件其实是指对象发出一定的消息，以通知操作的发生。引发事件的对象称为事件的发送方，而捕获事件并作出相应处理的成为事件的接收者。但是在实际中发出事件的对象往往不知道谁是接收事件的对象，因此在发送者和接收方之间需要一个中介，即为委托，委托也类似 C++中的函数指针。

用户交互过程中涉及到的事件种类较多，鼠标的点击、停留等一般事件常用的界面库已经做了较好的封装，需要实现的主要是自定义的事件。

本系统的自定义事件处理主要体现在界面信息的刷新上，以下为更新当前障碍物数目和玩家分数的自定义事件的实现。

通过 public delegate void EventHandler(int value)定义一个名为 EventHandler 的委托类型，而通过定义一个委托类型的事件可以使对象向外发布消息，一个事件的产生往往需要事件的发布方，以及一个或多个事件的接收方，以下为事件发送方的处理流程：

1. 定义更新消息的函数；

2. 定义消息的发送方函数，在需要发布事件的部分，调用消息函数；

相应的，在事件的接收方，也要添加相应的逻辑来处理事件的相应。例如在游戏运行主界面 UIPlayerInfoPanel，需要关心玩家的血量，通过添加事件的监听函数 OnHealthChange(int value)来实现事件的响应。如下为事件的接收方的逻辑流程：

1. 为发送方定义好的事件增加一个委托函数，用于处理事件的响应；

2. 实现事件的响应函数。

（2） 协程实现 UI 刷新

界面上时常会有一些需要频繁刷新的操作，例如计时器的显示或者一些频繁切换的效果。本系统中游戏运行主界面的倒计时器的显示用到了协程函数。在 Unity 中直接通过调用函数 StartCoroutine()来实现一个协程，传入的参数要为迭代返回的 IEnumerator 函数才能够实现灵活调用。协程函数中通过 yield return null 在下一帧会从返回语句的下一条语句执行。如图 5-6 为用于 UI 刷新的协程函数逻辑实现。

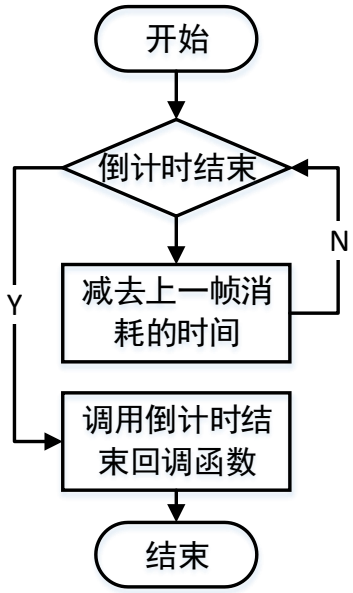
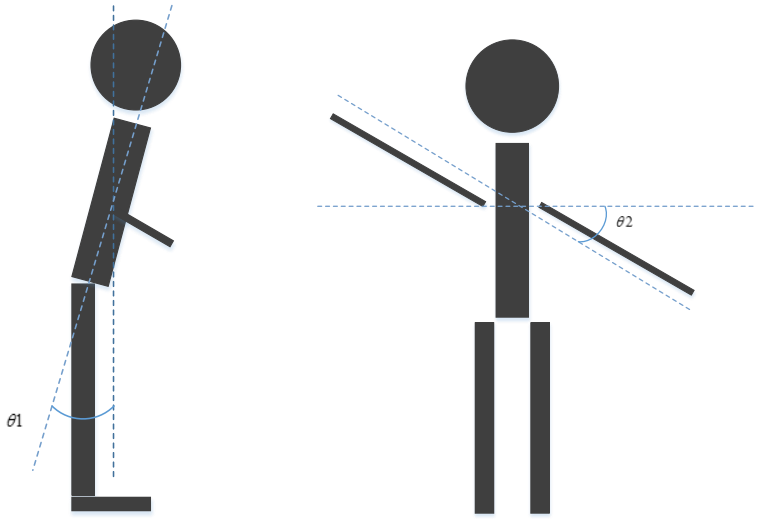


图 5-6 UI 刷新的协程函数

5.3 飞船控制系统

角色的动作控制主要通过接收用户的输入来控制模型。如图 5-7（a）玩家通过身体前倾或者后仰控制飞船的俯仰角。如图 5-7（b）玩家通过展开双手与水平面保持一定的角度来控制飞船的左右旋转。左右任意一只手握拳控制飞船射击。



（a） （b）

图 5-7 控制飞船方向姿势

5.4 裸眼模块实现

此前在第二章已经对裸眼 3D 的原理进行了简单的原理描述，本节将实际实现裸眼 3D 合图算法。本系统使用的硬件为长虹的 48 英寸 4K 裸眼屏幕，具体参数见表 5-1。

表5-1 长虹4K 裸眼屏幕参数

屏幕尺寸	48英寸	裸眼3D 技术	柱镜光栅
视点个数	8视点	屏幕比例	16:9
亮度	350cd/m²	对比度	1200:1
分辨率	1920*1080	3D 最佳可视角度	120°
3D 最佳观看距离	3-6m	规格	1124mm*663mm*73mm

5.4.1 八视点图像获取

系统使用的屏幕只支持 8 视点，所以游戏的画面获取采用了由 8 个摄像机的摄像机组，8 个子摄像机分别获取 8 个视点的所看到的图像。要达到更好的裸眼效果，需要让摄像机尽可能的模拟人两个眼睛间的关系，即 8 个子摄像机的相对位置和旋转，关键在于三点：子摄像机排列方式，子摄像机间隔，摄像机组焦距。子摄像机排列排列方式采用弧形排列，如图 5-8，使每个相机到焦点的距离相同，这样能有效避免两个相机间的画面断层问题。

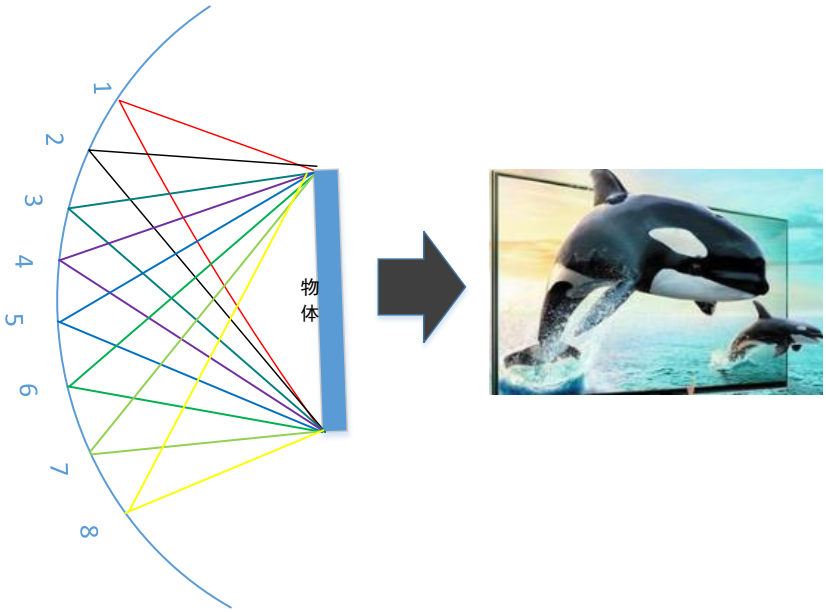


图5-8 相机排列示意图

摄像机的聚焦很好解决，Unity 的 Transform 组件提供了 LookAt 方法，可改变物体的旋转使之朝向传入的点的位置。为了方便子摄像机的排列，将子摄像机全部放入摄像机组物体下，如此，只用考虑其在父物体坐标系下的位置即可，同时在操作摄像机组做普通相机的跟随效果时，也不用考虑其 8 个子摄像机的位置。子摄像机间隔排列通过给定的焦距以及相机间隔确定。如图 5-9，为子摄像机排列方式的俯视图，1-8 为 8 个摄像机排列的位置，横轴为 Z 轴，纵轴为 X 轴，由于子摄像机都处于同一水平面上，所以可将所有子摄像机的 Y 值都设为 0，使其简化为二维。如公式 5.1，H 为焦距，delta 为子摄像机间弧长。可先求得两个摄像机间的夹角 θ，然后代入摄像机的序号 i，则可求得摄像机和 Z 轴的夹角 θ_i，将夹角 θ_i 代入式 5.1 中的最后两个子公式，即求出子摄像机的 X 坐标 X_i 和 Z 坐标 Z_i。如此就得到了 8 个子相机在父物体坐标系下的坐标，摄像机的排列问题就解决了。

$$\begin{cases} \theta = delta / H \\ \theta_i = 4.5 \times \theta - i \times \theta \\ x_i = \sin\theta_i \times H \\ z_i = \cos\theta_i \times H \end{cases} \tag{5.1}$$

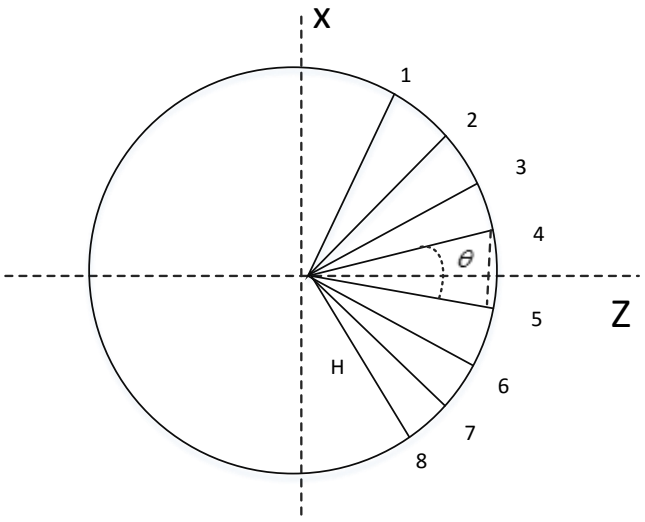


图5-9 相机排列俯视图

5.4.2 裸眼图像合成

通过上一节的相机组，系统获得了 8 张依次排列的画面，而裸眼屏幕和普通屏幕一样，只接受一张图显示，所以需要根据屏幕的特性，开发特定的算法，由于涉及到屏幕的详细硬件设计，属于商业机密，因此对原算法的优化难度相当大，其算法裸眼

呈现效果理想，所以系统只是优化了算法的一些效率问题，并没有改变核心算法。长虹提供的算法涉及到了屏幕上硬件的适配，算法细节不详细讨论，只介绍一下算法的简单流程图。如图 5-10，算法以 GPU 编程 Shader 实现，通过 Unity 传入的 UV 坐标获取到像素的实际坐标，由于裸眼屏幕像素的排列为倾斜，所以需要将 X 减去 cot*Y 的偏移，然后将坐标余 8 求得当前像素应该从哪张图上取。如图 5-11 a 为普通相机看到的方块，5-11b 为通过 8 视点合成算法合成的图片。

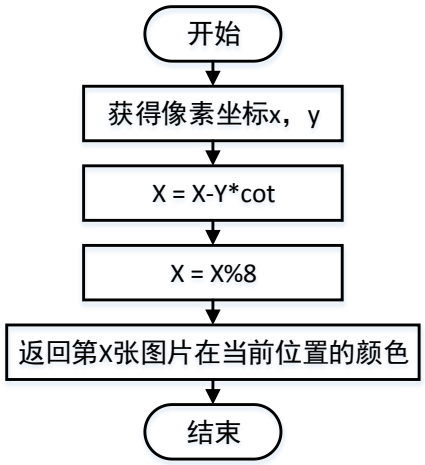
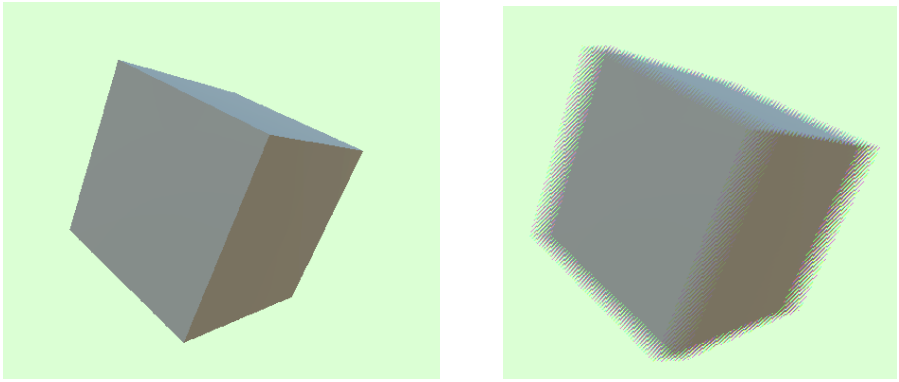


图5-10 裸眼图像合成算法简要流程图



（a） （b）

图5-11 裸眼图像合成算法效果图

5.5 网络通信系统

Unity 在 5.1 中推出了新的网络引擎模块 UNET，通过 UNET 不用单独开发服务器，可以很方便的创建，UNET 提供了一套网络开发 HLAPI（高级 API），通过 HLAPI 开发网络游戏无需关心底层网络细节。HLAPI 主要提供了以下功能：

（1） 消息处理程序

（2） 通用高性能序列化

（3） 分布式状态管理

（4） 状态同步

（5） 网络组件

5.5.1 UNET 网络模块

使用 UNET 开发的网络游戏有一个服务器和多个客户端。当没有专用的服务器时，客户端之一扮演服务器的角色，称之为此客户端 host（本地主机）。在 host 上，服务器和客户端在同一进程中，在 host 上的客户端 LocalClient（本地客户端），而其他客户端被称为 RemoteClients（远程客户端），其结构如图 5-12。LocalClient 与服务器通信通过直接的函数调用和消息队列，因为它是同一进程中。它实际上与服务器共享一个场景。RemoteClients 通过定期的网络连接与服务器进行通信。

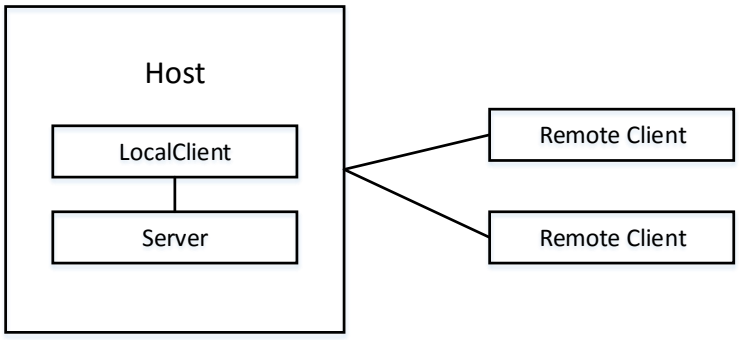


图5-12 UNET 服务端客户端结构

在单机游戏中，创建新的游戏物体直接在本地使用 GameObject.Instantiate 函数即可，但开发联机游戏，只能在服务器上创建一个游戏物体，然后游戏将被分布式状态管理模块同步到客户端。在网络系统中，玩家对象都是特殊的，每个人的 player 对象命令都会发送到该对象。一个人不能对另一个人的 player 对象调用命令。所以有“my player”对象的概念。当为一个客户端添加一个 player 时，这个 player 对象就成为该玩家客户端上的“local player” 对象。如图 5-13 显示两个客户端和他们 local players。属于本地客户端的 player 对象 被设置了 isLocalPlayer 标志。这标志可以用于控制输入的处理以及相机的跟随。除了 isLocalPlayer 标志，player 对象还有“local authority”标志。 改标志为 True 表示拥有客户端上的 player 对象的管理权，最常用到的是玩家

输入控制运动。对于非玩家对象如敌人，没有相关联的客户端，其控制权就在服务器上。NetworkBehaviour 上的属性 "isAuthority" 表示本地客户端是否有对象的控制权。

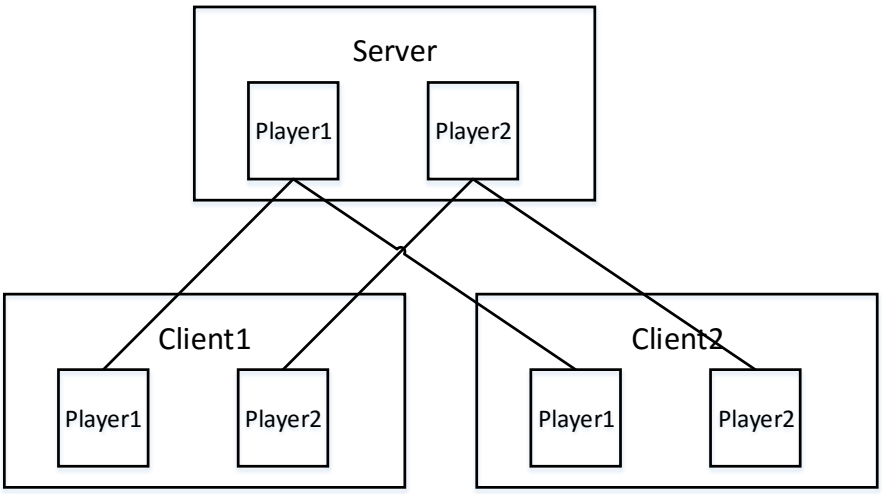


图5-13 UNET 物体同步

Unity 提供了两种方式使客户端和服务器直接交互，RPC（Remote Procedure Call 远程过程调用）方式，PRC 在服务器调用，在客户端执行。Commands（命令调用）方式在客户端调用，在服务器执行。联机游戏的总体流程如图 5-14 所示，首先客户端连入到服务器，服务器生成物体，服务器将物体同步到客户端，服务器和客户端通过 CMD 和 RPC 交互，客户端断开连接，服务器销毁游戏物体。

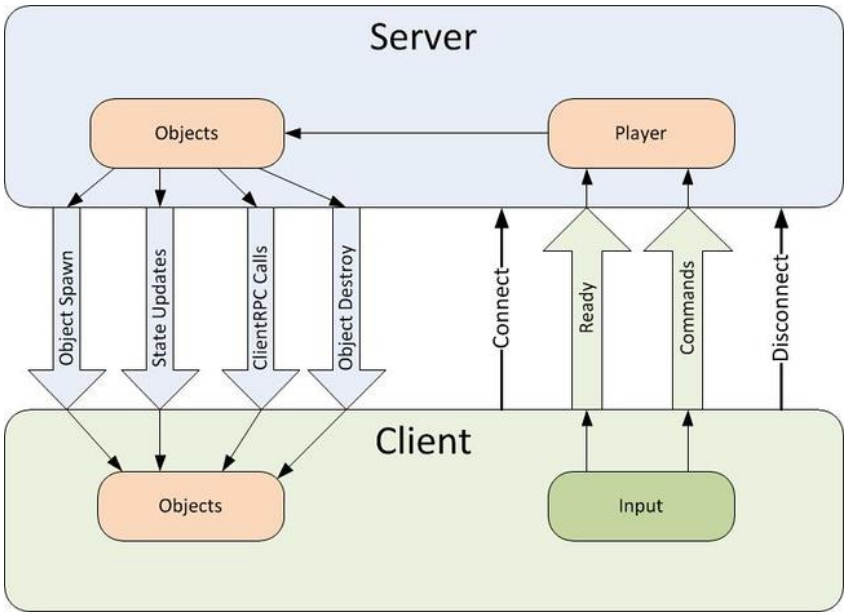


图5-14 客户端服务器交互

5.5.2 局域网广播

有了 UNET，可以很便捷的处理物体的状态同步，UNET 还有完善的消息机制和序列化机制。客户端和服务器的通讯也封装的很简洁。UNET 提供的局域网连接方式是客户端通过传入服务器的 IP 地址和端口，和服务器建立 TCP 连接，但是在局域网内，玩家可能并不知道服务器的 IP 和端口，针对这种问题，系统实现了基于 UDP 的局域网内广播系统，当玩家建立服务器后，在开始游戏前都会向整个局域网内广播服务器信息，开始游戏后停止发送信息。而当客户端打开服务器列表界面后，将会开启一个单独的线程监听广播端口，每收到一个信息就尝试解析服务器信息，解析成功则将信息加入到服务器链表中，当关闭服务器列表界面后，关闭线程并清空服务器链表。由于在 Unity 中并不支持在子线程中操作游戏物体，所以服务器列表界面管理类需要每帧不断遍历服务器链表并刷新到服务器列表界面上。

5.6 Kinect SDK 接入

Kinect 支持的.net 环境最低为 4.0 版本，而当前最高版本的 Unity5.3 仅支持到.net3.5，无法通过 Unity 直接获取 Kinect 传感器数据，采用的开发包为 KinectV2WithMS-SDK。本小节主要针对 Kinect 的接入流程、动作识别两个部分进行阐述。

5.6.1 Kinect 接入流程

以下简述 Kinect SDK 接入的主要流程以及 Kinect 动作识别部分的主要实现思想。首先是 Kinect 识别工具接入的主要流程：

在 Kinect 传感器识别动作的硬件方面，涉及到的 Kinect 的传感器核心就是骨骼跟踪技术，即通过红外投影机发射红外线，红外摄像头接收红外光反射，再计算出视场范围内每个像素的深度值[28]，从深度数据总体取出物体主题和形状以及每个像素点的用户索引信息，从而根据形状信息来匹配认人体的各个部分[29]。在动作识别的前期一般涉及到几个比较重要的处理步骤：

（1） 空间坐标的转换：即通过矩阵转换将 Kinect 的空间坐标转换为更有意义的用户空间坐标，以用户作为坐标原点进行处理；

（2） 特征数据的表示：由于骨骼在某个时间的状态为静态姿势，骨骼关节在空间的运动具有一系列特征参数，通过预先定义这些特征参数，来实现对特定的骨骼动

作的识别[30]。其中的特征数据包含着关节点的三维坐标信息，双手空间距离，左右手距离身体的水平距离，上身的长度等等，通过每个关节定义的向量属性，能够实现其运动方向和速度的计算。

5.6.1 姿势识别

由于本系统的控制动作设计并不复杂，在 4.3 节飞船控制系统中已经描述过了，主要包含五个动作：双手展开左/右倾斜、身体前/后倾斜、任意一只手握拳。其中握拳在 Kinect 开发包中已经提供，不用重新开发握拳算法，下面将介绍前 4 个动作的识别方式。

（1）双手展开左/右倾斜

首先需要判断双手是否展开，由于每两个人臂展都不相同，不能简单通过判断双手的空间距离是否大于一个固定的阈值判断，考虑到每个人的臂展和上半身的长度是有一定的比例，因此系统通过计算玩家上半身的长度再乘以一个比例作为阈值，双手空间距离改阈值则判断为双手展开。在双手展开状态下通过双手坐标即能计算出双手与水平面的夹角。如图 5-16 为倾斜角度的识别流程图，MIN 为最小识别角度，MAX 为最大识别角度，θ 为双手展开同水平面的夹角，V 为最终得出的左右倾斜度。为避免误操作，当 θ 小于最小识别角度 MIN 时判断为无效输入，当角度大于最大识别角度 MAX 时，将角度设置为最大角度，最后通过最后一步中的公式将角度归一到 0 到 1 之间，最后得到的 V 就是一个 0 到 1 之间的数。

（2）身体前/后倾

同双手的倾斜算法相似，将双肩到臀部的向量同 Y 轴所成的夹角代入到图 5-15 的描述的算法中即可同样得到一个 0-1 之间的数。需要注意的是判断身体前/后倾时识别范围需要调整。

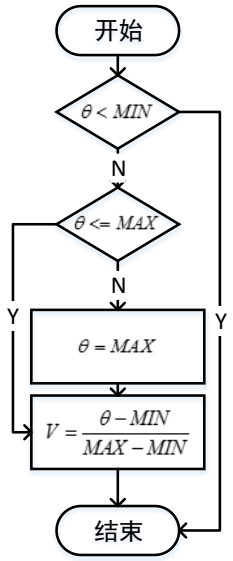


图5-16 倾斜角度计算流程图

5.7 本章小结

本章主要阐述了裸眼 3D 体感游戏系统方案设计，对整个游戏系统的设计以及各个模块的设计过程进行了着重描述，分别从游戏的总体框架设计、游戏美术设计、角色控制系统、裸眼模块、网络通信模块、Kinect SDK 接入几个角度来进行方案设计，其中重点讨论了每个模块的技术难点以及本系统采取的解决方案。

第 6 章 裸眼 3D 体感游戏系统实现与结果分析

本系统的实验平台主要参数如表 6-1。

表6-1 实验平台参数

序号	属性名称	参数
1	处理器	Inter(R) Core(TM) i7-4800M
2	CPU 主频	2.7GHz
3	内存	8GB
4	操作系统	Windows8专业版(64位)
5	显示器分辨率	1920*1080
6	集成开发环境	Visual Studio 2013
7	调试工具	Visual Studio 2013
8	游戏引擎	Unity5.3
9	界面制作工具	UGUI
10	图片处理工具	Photoshop CS6
11	模型处理工具	3DMax 2014
12	开发语言	C#
13	体感摄像头	Kinect for Windows2.0
14	游戏帧数率	30FPS

本系统按照功能性的划分主要分为基本的裸眼 3D 显示、游戏功能两个主要的功能模块，是本系统的主要技术点。本章节将分别对两个功能进行实现验证，并给出游戏的运行效果。

6.1 核心游戏功能实现

本系统中，核心的游戏功能就是指用户通过开始菜单界面进入游戏之后的一系列游戏活动。通过预定义的肢体动作：双手左右倾斜，身体前后倾斜，任意一只手握拳等动作来控制飞船飞行，操纵游戏中的行为活动。实现了游戏能够根据用户给出的动作指令给出相应的动作。

本系统在 PC 以打包后的 exe 文件执行。启动可执行文件后，游戏默认全屏运行，并且该游戏的裸眼模式只能在长虹的 4k 裸眼屏幕上使用。游戏启动之后，首先进入的游戏开始菜单界面，图 6-1 为游戏的开始界面。

图6-1 游戏开始界面

在进入游戏之后，主界面菜单上呈现的几个选项：CREATE（创建服务器），JOIN（加入服务器），SETTING（游戏设置），QUIT（退出游戏）可供选择，点击按钮的方式为当系统检测到 Kinect 时会自动启用 Kinect 控制，否则使用鼠标控制。点击设置按钮跳转到如图 6-2 的游戏设置界面，可以选择是否打开 Kinect 控制和是否打开裸眼显示模式。注意在此处如果没有插入 Kinect 设备，无法选择打开 Kinect 控制。

图6-2 游戏设置界面

点击创建服务器，会进入玩家列表界面，如图 6-2 界面显示连入改服务器的玩家列表。在主界面菜单点击加入服务器按钮，页面会跳转到服务器列表界面，不断接受局域网中的服务器广播消息如图 6-3 为局域网中的服务器列表，选中某一服务器点击 JOIN 跳转到图 6-2 的玩家列表，当服务器中的所有玩家都选则准备后，进入游戏场景。

图6-2 玩家列表

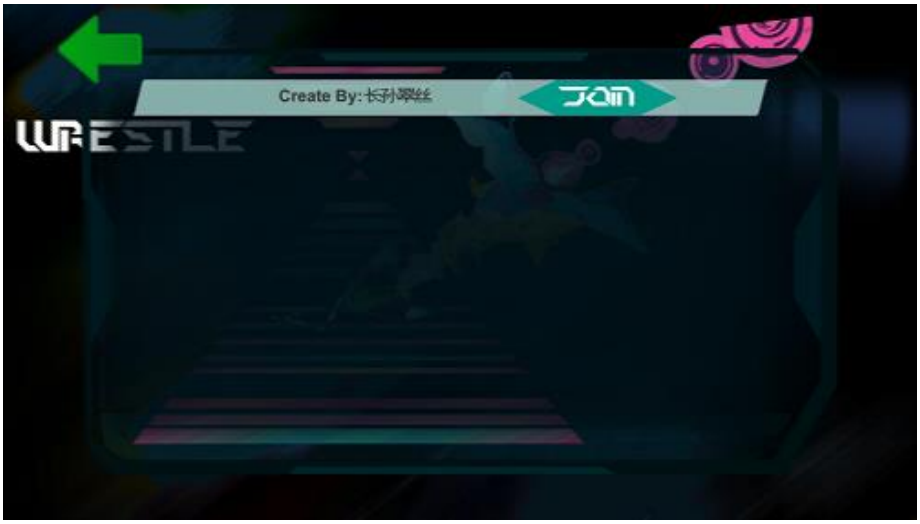


图6-3 服务器列表

在游戏主场景中，UI 界面主要显示玩家信息，包括玩家名、血量、击杀数和飞船外观，为了避免玩家间分不清敌我，如图 6-4 系统提供了四种飞船外形，玩家通过手势或者鼠标操作飞船发出激光炮，飞船每被激光炮打中一次减少一滴血，当血量为 0 时，玩家飞船损毁。如图 6-5 每当玩家击落一艘飞船，将会在玩家信息下加一个击杀图标。当玩家被击杀后，会显示如图 6-6 的倒计时画面，倒计时结束后玩家重生。当某一玩家击落 3 艘飞船后游戏结束，显示如图 6-7 的结算界面。结算信息包括击杀数，死亡数，自己的信息将蓝色高亮显示，点击退出图标游戏退出到主界面。在游戏过程通过键盘或者手势可以打开系统菜单，有关闭/打开声音选项和退回到主菜单选项。

6.2 结果分析与总结

前面已经详尽的展示了本系统中的主要功能界面，阐述了实现的主要功能内容。以下部分要分析总结该系统的稳定性及运行的资源消耗情况，同时也总结该系统存在的不足和值得改进完善的地方。

（a） （b） (c) (d)

图6-4 四种飞船外形

图6-5 游戏进行中画面

图6-6 重生倒计时画面

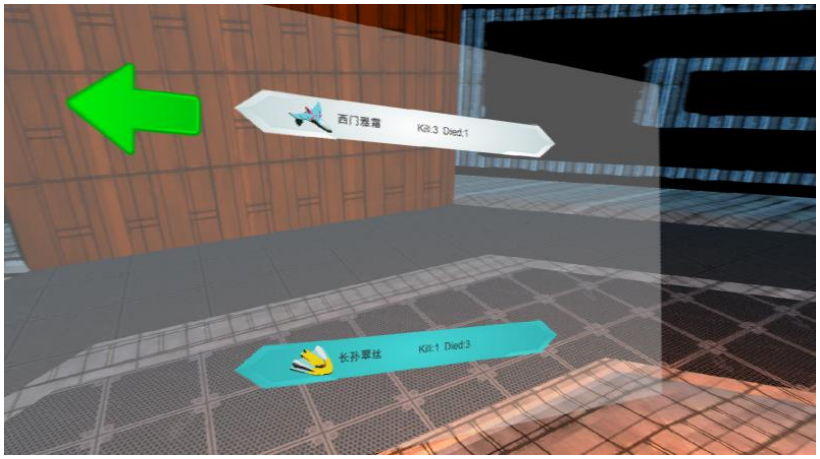


图6-7 结算界面

首先是游戏性能的衡量，根据 20 次游戏运行的 CPU 和内存占用采样统计，表 6-2 为系统运行时性能的数据样本。

表 6-2 系统性能数据

序号	CPU 占用(%)	内存占用(Mb)	序号	CPU 占用(%)	内存占用(Mb)
1	17	126	11	16	187.1
2	13	126	12	11	193.1
3	28	138.5	13	18	125.2
4	30	158.3	14	24	137.2
5	27	157.9	15	25	130.6
6	26	156.8	16	30	130.2
7	23	167.3	17	20	158.6
8	22	173.4	18	14	143.2
9	19	77.3	19	16	183.6
10	13	185.7	20	23	178.5

实验表明，系统在刚启动阶段最耗 CPU 和内存，但是在运行中会逐渐趋于稳定，在多次的测试过程中没有系统崩溃或异常的情况发生，表明系统还是具有较高的稳定性。

虽然系统具有较高的稳定性，并且能够在预定的平台上顺利的运行，但是该系统仍然存在很多的问题：

1. 尽管内存和 CPU 占用经过了优化，但是本系统还是比同样大小安装包的游戏更耗性能，游戏的帧速率在低配置平台不够稳定；

2. 没有实现单人玩法，只有对战玩法，玩法上有所欠缺；

3. 裸眼效果不理想，缺乏硬件知识，无法改进裸眼算法；

4. 游戏开发经验不足，C#还欠深入学习，代码构建上不够合理，很多类的方法耦合度较高，使得很多方法不容易扩展和维护；

5. 对于设计模式的使用不够熟练，类的设计结构还不够合理。

6.3 本章小结

本章主要针对该裸眼 3D 体感游戏进行了较为全面、详尽的展示，主要针对游戏中的核心玩法、游戏中涉及到的主要障碍物和不同的动作操作进行了展示，并且展示了部分特效表现的截图。同时也展示了几个主要功能界面：游戏设置界面、玩家列表界面、服务器列表界面以及游戏结束的结算界面。除了界面上的展示之外，还分析统计了游戏在多次运行情况下的内存占用情况以及系统的稳定性，并通过数据和结果证明了该系统能够在预设的实验平台上流畅的运行，并且具有较好的稳定性。

结论

随着计算机软硬件的不断发展，虚拟现实与人机交互行业不断的变革。裸眼 3D 技术不断发展，裸眼 3D 技术成为了影像行业最前沿的高新技术之一，它改变了传统平面图像带给人们的视觉疲惫感，以其生动的表现力，优美高雅的环境感染力以及强大的视觉冲击力感染了大众。

同时近些年体感技术也不断蓬勃发展，为人们的娱乐方式提供了一种全新的理念，使得作为游戏参与者的玩家不需要禁锢在传统的鼠标、键盘操作计算机的方式中，能够使得身心都从虚拟的游戏空间解脱出来，让自己身临其境的体感游戏带来的真实感和代入感，并且在放松的游戏体验当中身体得到充分的锻炼，同时能够有效的缓解传统的电子游戏对身体健康的影响。

裸眼 3D 带来了更真实的沉浸式视觉体验，体感交互带来了一种全新的交互方式，为人们的娱乐方式提供了一种全新的理念。本系统将两者有机的结合起来，探索了裸眼 3D 游戏的可行性，实现了一款飞行射击裸眼 3D 体感游戏，玩家通过自然的人体姿势控制飞船运动和涉及，裸眼 3D 立体效果明显，为游戏玩家带来了沉浸式的游戏体验，体感丰富的肢体动作又使得用户能够充分体验游戏的互动性，并且体感的输入控制让玩家正好可以在裸眼 3D 最佳观看区域同游戏交互，弥补了裸眼 3D 效果必须在 1M 外观看的缺点，两者结合起来相得益彰。

在游戏的开发过程中，查找了大量国内外裸眼 3D 的研究资料以及体感游戏开发的情况，游戏的设计中也体现了作者对于游戏的一些看法。本文首先研究了裸眼 3D 和体感输入两个技术要点，然后根据游戏涉及到的游戏开发知识进行了较为深入的讨论和分析，对于游戏设计中主要包含的场景设计、游戏美术设计、角色控制部分、Kinect 接入 Unity 的方法、裸眼图片合成的方法进行了较为细致的讨论，并举例阐述了当前游戏美术中重要的设计制作方法。网络游戏并不是在单击游戏上简单的修改即可完成，需要处理很多在单击中根本不会遇到的问题，在网络模块的设计中，在阅读完 Unity 提供的网络模块官方文档后，由于网络模块推出时间不久，文档并不全面，根本不知道如何下手。在开发过程中遇到很多问题，只能去阅读源代码。在磕磕碰碰中，终究功夫不负有心人，完成了整个游戏的设计。

总体而言，裸眼 3D 体感游戏有较好的游戏体验。虽然当前裸眼 3D 在国内的发展还不够成熟，国内裸眼 3D 应用开发商也不多，但是从裸眼 3D 的沉浸式体验来看，还是有巨大的市场潜力和广阔的发展空间。

致谢

时光荏苒之间，回首来看，四年匆匆而过，就要告别母校西科大了，四年来学到了很多，大一泡图书馆学会了如何自己学习，管理个人时间，大二开始泡实验室，不断寻找自己的方向，寻找过程中不断补充基础知识。在大二下学期，开始了游戏开发学习，专业技能不断提升，更重要的是通过不断的遇到问题解决问题，学习到了解决问题的方法，感谢西科大教会了我这么多。

感谢一路上给予指引的老师们。其中首先感谢是指导老师吴亚东老师和张晓蓉老师。吴老师和张老师在专业上严谨认真、敬业负责的态度一度让我感动，让自己不断向他们看齐，从他们身上学到了太多太多。

感谢将作者领进虚拟现实实验室的陈永辉老师，使得我从大二开始就有机会接触 Unity 游戏开发，还要感谢虚拟现实实验室，给了这么好一个平台，让作者学以致用。感谢实验室的师兄师姐们，和大家在一起学习，奋斗真的很开心。

最后，最为感激的还是父母。四年的外地读书已经让他们为我操心不少，即将离开校园，为了更好的机遇去了离家更远的北京也让父母更加牵挂。这几年每次回家停留的日子都不多 ，但是幸好有一个幸福完美的家庭，有身体尚且健康的父母。感激他们一直以来的支持和鼓励。暂时走向远方是为了风雨兼程的努力之后、明天一个更强大更独立的人回报父母对我的养育之爱，为了我爱和爱我的你们，作者会努力让自己越来越强大，让你们越来越骄傲。

参考文献

[1] LAWTON.3D displays without glasses: coming to a screen near you[J]. computer,2011,44(1):17-19

[2] 夏勇峰. Kinect 与人机交互的未来[J]. 商业价值, 2011(2):60-63.

[3] XunBo Yu. Autostereoscopic three-dimensional display with high dense views and the narrow structure pitch [J]. Chinese optics letters ,2014(6):30-33.

[4] 雷雯. 基于体感交互中间件的人体互动框架的设计与实现[D].北京：北京交通大学，2012.1.

[5] 胡颖群. 基于 Kinect 体感识别技术的研究与实现[J]. 甘肃科技纵横,2013(7):18-20.

[6] 王树斌. 浅析 Unity3d 开发游戏流程及常用技术[J]. 电脑知识与技术,2012(2):351-352.

[7] 吴志达. 一个基于 Unity3d 游戏引擎的体感游戏研究与实现[D]. 广州：中山大学，2012.

[8] 郭冠军. 裸眼3D 视频转换技术研究[D]. 电子科技大学, 2013.

[9] Dodgson N. Autostereoscopic 3D Displays[J]. Computer, 2005, 38(8):31-36.

[10] Jianli Luo. GPU-Based Multi-view Rendering for Spatial-Multiplex Autostereoscopic displays[C]. IEEE International Conference on Computer Science and Information Technology.2010:28-32

[11] 刘东泽. 基于柱镜光栅的仿真立体图像生成方法[J]. 印刷杂志,2012(11)：23-25.

[12] 侯春萍,许国,沈丽丽.基于狭缝光栅的自由立体显示器视区模型与计算仿真[J]. 天津大学学报,2012,45(8):677-681.

[13] 姚剑敏,辛琦,郭太良. 自由立体显示器中锯齿状交错狭缝光栅的设计[J]. 光子学报,2012,41(10):1176-1179.

[14] LIU R. Study on Binocular Stereo Imaging Based on Computer Stereo Vision[D]. Dissertation of Chongqing University,2007

[15] WANG S X. Research on the Realization Method of Nakedness-Eye Stereoscopic Display Based on Single Chip DMD[J]. Chinese Journal of Electron Devices, 2008(1):16-18.

[16] GUO H,QING K,MAO M,etl. Real-Time Tiled Multi-projector Autostereoscopic Display Algorithm for Dual-view 3D Video Files [J]. Journal of Computer-Aided Design & Computer Graphics,2015,27(9): 1734-1742.

[17] ZHOU L,TAO Y,WANG Q,etl. Design of Lenticular Lens in Autostereoscopic Display[J]. Acta Photonica Sinica,2009,38(1):30-33

[18] SHI D,KANG X,LI S,etl. Multi-Views 3D Display Technology[J]. Electronic Science & Technology,2014,01(03):276-282

[19] Huang J, Wang Y. 30-view projection 3D display[J]. Proceedings of SPIE - The International Society for Optical Engineering, 2015, 9385:93850B-93850B-7.

[20] 王宏安, 戴国忠. 自然人机交互技术[J]. 中国图象图形学报, 15(7):980-983.

[21] 黄石. 论游戏设计的基本原则[J].装饰, 2007, (6):34-36.

[22] 王树军.三维游戏引擎中物理引擎关键技术的研究[D].天津：天津大学，2007.

[23] 张帆. Unity3D 游戏开发基础[M]. 杭州:浙江工商大学出版社，2013：34-35.

[24] 宣雨松. Unity3D 游戏开发[M]. 成都:四川电子音像出版中心，2012：15-18.

[25] 刘炜伟，张引，叶修梓. 3D 游戏引擎渲染内核架构及其技术[J]. 计算机应用研究，2006，（8）：45-48.

[26] 徐宇峰，刘秀珍，王革. 3D游戏引擎构架及游戏动画渲染技术[J]. 多媒体技术及其应用，2008，（7）：1324-1328.

[27] 王超.3D 游戏引擎场景渲染技术的研究与实现[D].武汉：武汉理工大学，2006.

[28] George Jacob. Surgrons use hand gestures and/or voice commands without interrupting the natural flow of a procedure[J]. Communications of ACM,2013,vol.56:pages 68-75.

[29] Samuele Gasparrini *. & Enea Cippitelli. & Susanna Spinsante and Ennio Gambi. A Depth-Based Fall Detection System Using a Kinect®Sensor[J]. Sensors,2014,vol.14:pages 2756-2775.