

# 基于 Unity3D 的防守游戏设计与实现

李思丞 杨 键

四川大学锦城学院计算机与软件学院 四川 成都 611731

**【摘要】**随着社会的高速发展,人们生活的节奏越来越快,很多人生活在繁忙的工作之中,下班后又找不到合适的放松项目。为了满足现代人在闲暇之余能找到一个放松的需求,旨在开发一款让人们在闲暇时间能够玩耍,并且具有一定考验玩家反应力的三维 RPG 类游戏。本游戏基于 Unity3D 游戏开发技术,在 VisualStudio2019 平台使用 C# 开发语言设计并实现了此款名为小骑士的三维 RPG 游戏。玩家通过不断击败敌人来获得游戏乐趣,并且最终成功保卫国家来增加成就感。

**【关键词】**Unity3D; RPG; 碰撞检测

## 1 绪论

### 1.1 游戏开发的目的和研究的意义

本游戏旨在实现一个 3D 的 RPG 单机的防守游戏。本游戏中的主人公小骑士为保卫国家,不断地挑战前来入侵的敌人,通过不断地升级让自己变得更强大,最终完成保卫国家的使命,本游戏主要考验玩家在极短的时间内做出反击的能力,可以锻炼玩家的反应能力也可以让玩家在繁忙的生活中获得放松。

### 1.2 国内外研究现状

目前国外的很多游戏例如《暗黑破坏神》,《流放之路》等都是这种类型的游戏,并且在国内外都取得了不错的成绩,但是我国国内这方面的市场任然十分匮乏,目前国内这方面游戏大多数都是从国外引进。

### 1.3 游戏的玩法

本游戏中共有两种角色:小骑士和敌人,玩家在进入游戏后可以进行分析,奔跑,放技能等操作,敌人会在三个点定点生成,并且随着时间的增加敌人的生成速度会随之变快,当击败所有敌人后,玩家可以取得胜利,当角色的生命值为 0 时则挑战失败。

## 2 开发工具及技术

### 2.1 Unity3D

Unity3D 为目前主流的游戏开发引擎,集动画,音效,物理于一体,可以任意地搭建场景,编辑地形,有着通过脚本来对角色或者物体进行控制的特点,基于它强大的引擎还能做出真实的场景,所以本游戏基于 Unity3D 来进行对场景搭建和游戏中各种组件的设置。

### 2.2 C#

本游戏中的脚本基于 VisualStudio2019 中的 C# 语言来进行编程。

### 2.3 3Ds Max

基于 3Ds Max 做三维图形好,并且上手容易的特点,本游戏中的部分物体基于 3Ds MAX 来进行建模。

## 3 游戏主要功能及实现

### 3.1 场景搭建

为场景添加一个 Terrain 也就是地面,导入一个环境包,为绘制地面做准备,使用笔刷工具添加一种草地材质,再添加一种土地材质,用笔刷道具把土地绘制在草地上,用山峰道具绘制山脉再用树木道具绘制树木。

### 3.2 主角的控制

为主角添加一个动画组件控制角色的动画,一个 Rigidbody 用来模拟主角的物理特性,一个胶囊碰撞体来模拟主角与物体之间的碰撞,一个声音组件用来控制主角的音效,最后用一个 Player 脚本实现主角的交互。

#### 3.2.1 主角运动

首先用 Input.GetAxis 来获取主角的水平输入量和垂直输入量,用一个 Vector3 变量来存储这两个输入值,当 Vector3 变量不为 0 时,用 transform.rotation = Quaternion.LookRotation(dir) 来控制角色的旋转方向,再用 transform.Translate(Vector3.forward \* speed\*Time.deltaTime) 来改变角色的位置,因为主角一直往前所以用 Vector3.forward 即可。

#### 3.2.2 主角的攻击

在写角色攻击代码的时候先要提前做好准备工作,给主角的剑上添加一个 Box Collider 组件,并且勾选 IsTrigger,平时不用的时候将这个碰撞体组件禁用掉,用 attackTimer += Time.deltaTime 来计算每次攻击的时间,当主角按下 J 键时并且攻击间隔大于 0.5 秒的时候,用 Trigger.enabled=true 调用出这个组件,重置攻击时间,定义一个 AtkEnd 函数来关闭这个组件,用 Invoke("AtkEnd", 0.1f),攻击一秒后关闭这个组件。

#### 3.2.3 主角的动画

在本游戏中,主角的动画分为四种动画,当主角不动时播放主角的站立动画即 Idle 动画,当主角跑动时播放主角的跑步动画即 Run 动画,当主角死亡时播放主角的死亡动画即 Dead 动画,当角色攻击时播放攻击动画即 Attack 动画。最后在 Animator 面板中将各个动画结合起来。

#### 3.2.4 主角的生命值控制

定义一个变量 HP=100,生命值的减少脚本写在主角脚本里,写一个 AddHP 方法,参数为 hp,内容为 HP += hp,每当敌人攻击到主角时就减 10 点生命并在敌人脚本中跨脚本访问这个代码 player.AddHp(-10)。当生命值等于 0 时,敌人播放死亡动画 anim.Play("Death")。

#### 3.2.5 主角血条显示

在绘制主角的生命值控制代码之前,我们需要

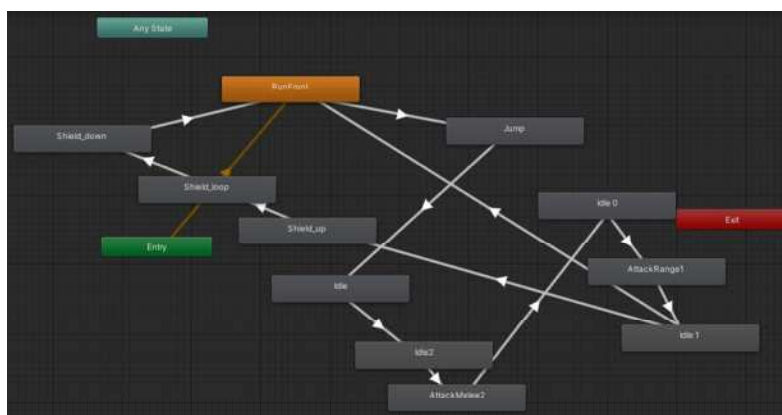


图1 动画状态机展示

先在Unity3D中添加一个Slider，并且将它的背景Background 设置为白色，将它的填充Fill 设置为红色。计算出人物的剩余血量 lifeleft 和总血量 totallife，两者相除再乘以最大显示血量即可， $slider.value = (lifeleft / totallife) * slider.maxValue$ 。

### 3.3 敌人功能设计与实现

敌人身上的组件主要为一个动画组件来控制敌人的动画，一个RigidBody 来模拟敌人的物理特性，一个胶囊碰撞体来模拟敌人与物体之间的碰撞，一个NavMeshAgent 组件来烘焙敌人，使得敌人能自动跟踪主角，一个脚本来实现敌人的交互。

#### 3.3.1 敌人的自动追踪

为了实现敌人的自动追踪我们首先需要烘焙整张地图，在 windows 菜单中的 AI 中找到 Navigation 选项，对地图进行烘焙，敌人也一样需要烘焙。为了调用敌人中的 NavMeshAgent 组件，我们需要在代码的最开头使用 using UnityEngine.AI，并且定义一个 NavMeshAgent 变量，计算敌人与主角之间的距离， $dis = Vector3.Distance(transform.position, player.transform.position)$ ，当距离大于两米时开始追踪设置 agent.isStopped=false，设置敌人追踪的目的地 agent.SetDestination(player.transform.position)。

#### 3.3.2 敌人的孵化

为了保证游戏的挑战性，本游戏设置了三个敌人的出生点，并且游戏的进行时间越长，敌人的生成速度越快，在代码中定义一个敌人的预设变量，计算时间间隔的变量和一个敌人生成位置变量以及一个系统运行时间变量，当运行时间小于 20 秒时 15 秒实例化出一个敌人一个敌人，为提高游戏的难度性，当运行时间大于 20 秒时 5 秒实例化出一个敌人。最后在 Unity3D 中为敌人的生成位置赋值即可。

#### 3.3.3 敌人动画

敌人的动画分为敌人的 Idle 动画，即敌人的默认动画，敌人的跑步动画 Run 动画，当敌人追踪主角时播放此动画，敌人的死亡动画 Dead 动画，当敌人被主角击杀时播放，敌人的攻击动画 Attack 动画，当敌人攻击主角时播放。最后在 Unity3D 中的 Animator 面板中将各个动画联系起来。状态机和主角一致参考图 1。

#### 3.3.4 敌人死亡

每当主角攻击到敌人时，敌人生命值减 1，首先在敌人的 OnTriggerEnter 方法中写判断，当触碰到的碰撞体的 Tag 为 Player 时，if(other.gameObject.CompareTag("Player"))，敌人的生命值减 1，当生命值为 0 时调用死亡方法。

#### 3.4 游戏中的碰撞体检测

一般来说游戏 OnTriggerEnter 和 OnCollisionEnter 两种碰撞，一种是两种都是碰撞检测，OnTriggerEnter 是发生碰撞并且可穿过，OnCollisionEnter 是两者发生碰撞并且不穿过，本游戏中的用的是 OnTriggerEnter 来检测碰撞体检测，当物体与物体之间发生碰撞并且穿过时用 OnTriggerEnter，例如技能与敌人的碰撞，主角的剑与敌人的碰撞。

#### 3.5 游戏中的技能

本游戏中给主角添加了一个技能，技能实在 Unity 中 Asset Store 中下载的一个粒子效果，在脚本中把这个技能定义成一个变量，当玩家点击 K 键并且攻击间隔大于 8 秒时实例化这个技能 Instantiate(skillPre, transform.position+transform.forward \* 1f, transform.rotation)，技能的释放位置为主角前面加上一段距离，方向为主角方向。此外还需要单独定义一个技能脚本，当技能被实例化出 5 秒后销毁此技能 Destroy(gameObject, 5f)，添加一个碰撞检测，当检测到敌人的 Tag 时调用敌人的死亡方法。

#### 3.6 小地图功能

给主角添加一个相机，这个相机在主角的头顶上，将视野选取到一个比较合适的位置，再添加一个 UI 组件 RawImage，将 RawImage 放置在游戏场景中的右上角，添加一个叫做 RenderTexture 的组件，这个组件的作用是将 RawImage 和相机联系起来，将这个组件分别设置给 RawImage 的 Texture 和相机的 TargetTexture 即可实现小地图功能。

#### 3.7 游戏音效

本游戏音效的主要调用是主角和敌人的攻击音效，要调用音效首先在脚本中定义一个 AudioSource 和 AudioClip 组件，在 Start 函数中获得这个组  $s = GetComponent<AudioSource>()$

在调用的时候先将攻击的 AudioClip 赋给这个音效资源, 然后再用 `s.Play()` 播放此音效。

### 3.8 游戏的相机设置

此游戏使用相机脚本使得相机的距离和主角的位置控制在一定的距离之内, 首先计算相机与主角之间的距离 `dir = transform.position - target.position`, 在 Update 函数中实时更新主角与相机之间的距离 `transform.position = dir + target.position`, 最后在 Unity3D 中将主角赋给 target 对象即可。

### 4 游戏脚本

游戏的脚本主要包括主角的脚本, 此脚本主要操控主角的动画, 音效的播放, 移动与敌人的交互等操作;

敌人的脚本, 此脚本主要实现敌人追踪主角的功能, 敌人的动画与音效的播放; 敌人的生成脚本, 实现敌人的自动化生成; 技能脚本控制技能的时间以及触发事件; 相机脚本, 将主角的视角控制在一定范围内。

## 5 游戏测试

### 5.1 测试环境

本游戏基于 Unity3D 的 2019 版本, 电脑为 i7 处理器, 16GB 运行内存, 64 位操作系统, RTX2060 显卡的环境下进行测试。

### 5.2 测试内容

主要对敌人的追踪主角功能是否正常, 敌人与主角的交互是否正常进行测试, 测试部分效果图如下:



图 2 游戏测试

## 6 结论

### 6.1 总结与收获

本游戏基于 Unity3D 开发引擎和 Visual Studio 2019 平台, 使用 C# 语言开发并实现了此游戏, 游戏只有小骑士和敌人的交互, 还不能让玩家沉迷其中, 但是大量的敌人进攻提高了玩家的反应能力, 通关后也能让玩家获得成就感, 游戏的画面还不够精美。

### 6.2 展望

本游戏中的交互性还比较缺乏, 后续的设计中可以为游戏添加任务系统, 主角与 NPC 的交互系统来增加游戏的故事性, 此外还需要丰富游戏场景创建多种敌人,

丰富主角的技能系统使得游戏的趣味性增加。

### 【参考文献】

- [1] 朱喜基, 冯振辉. 基于 Unity3D 虚拟校园漫游碰撞检测的研究 [J]. 江苏科技信息, 2018(11):66-68.
- [2] 袁科, 高启文, 闫永航. 基于 Unity3D 的角色扮演游戏设计与实现 [J]. 软件导刊, 2019, 18(8):75-79.