基于 Unity3D 的探险游戏的设计

宋亚静 王春洁

四川大学锦城学院 计算机与软件学院 四川 成都 611731

【摘 要】本文详细介绍了 Unity3D 小鸭探险记的实现过程，阐述了游戏界面，游戏角色，虚拟摄像机等的设计。文中引入了手势识别的功能，也是一大亮点，将手势识别应用到玩家与游戏的交互中，能够充分提高游戏的可操作性，玩家不需要依赖于键盘操控，使得玩家与游戏的交互更贴近人类的行为习惯，代入感强，丰富了游戏功能的同时也改善了玩家的体验。

【关键词】Unity3D；游戏开发；界面设计

0 引言

在互联网日渐普及的今天，电子游戏如雨后春笋般进入了大众的视野，因其互动性和多变性逐渐成为人们精神娱乐的重要组成部分，其中探险类游戏由于具有一定的未知性和刺激性，成为了人们心中的游戏首选。本文设计的这款小鸭探险记关卡众多，敌人种类较丰富，灵活性较大，游戏体验较好，主角进入城堡即胜利，极大的吸引了玩家的兴趣，不仅可以使玩家放松心情，还可以锻炼玩家的灵活性，具有一定的应用价值。由于 Unity3D 引擎兼容性强、通用性强，因此本游戏的测试环境选为 Unity 2018.3.9f1，编程环境为 Visual Studio 2019，使用面向对象的开发语言 C#。

1 游戏整体设计

整个游戏界面分为开始界面，运行界面和结束界面三部分。整体逻辑如图 1 所示：

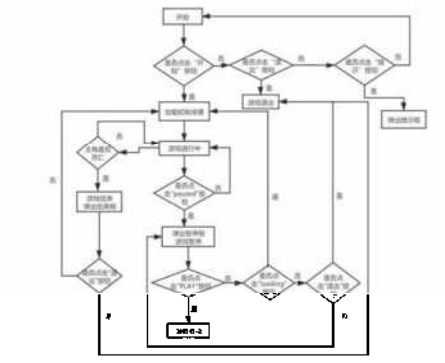

图 1 系统整体流程图

1.1 游戏开始界面

游戏开始界面是一款简单的卡通海报设计，有开始，退出和游戏提示三个按钮，点击开始按钮转换场景进入游戏运行界面，点击退出按钮退出运行状态，点击提示按钮弹出游戏规则框，这三个按钮使用统一的 UI 元素响应用户的单击。

1.2 游戏运行界面

游戏运行界面中有各种各样的怪物，水果，星星等，主角是一只小鸭，在碰到不同道具后释放不同技能，如主角碰到水果后主角生命值会加 1，碰到磁铁后会吸附一定范围内的水果，碰到子弹或者敌人后会死亡等，游戏中设置不同难度，不同种类的关卡用以给主角制造困难，同时主角拾取每一个道具后会将道具隐式地显示在屏幕右上角，按 C 键可以查看已拾取的道具，按 V 键隐藏，目的是使屏幕看起来更整洁，其次在运行界面中有一个按钮 Paused，点击此按钮弹出暂停框，此框内又包含 loading，PLAY 和退出三个按钮。

1.3 游戏结束界面

游戏结束主角死亡结束和探险成功结束两种情况。一旦主角死亡立即进入结束界面，此界面中有一个结束文本，显示几秒后重启，此外还包含一个退出按钮用来结束游戏。主角死亡后若用户不点击按钮，3 秒钟之后会利用 LoadScene 重新加载，若点击退出，直接退出运行。当主角进入到城堡触碰到碰撞体后，城堡的门关上，灯亮起，释放烟花特效即为探险成功，弹出胜利的结束框。

1.4 游戏 UI 设计

游戏中的按钮、弹出框、积分显示等都是用 UI 来实现的。按钮的实现是利用 UI 中的 Button，在 Button 组件下的 OnClik() 下设置按钮的交互事件，本游戏中的按钮包括开始、暂停、继续、退出、重启等。弹出框使用的是 UI 中的 Image 和 Text 组件，将 Image 作为弹出框的背景，在 Text 组件下的 Text 框中输入文本，此文本开始是 Text.SetActive（false），当触发到某一事件如调用 GameOver() 或点击暂停时利用 Text.SetActive(true) 使文本显示出来。积分显示用的是图标加文本的形式，在屏幕上方更新数值。击中敌人的数量是以图标显示的方式，击中多少敌人就显示多少个图标在屏幕上。

```
    enemyChildCount = Enemy.transform.
childCount;
    int enemy = GameManager.instance.EnemyCount;
            for (int i = 0; i < enemyChildCount;
i++){
    Enemy.transform.GetChild(i).gameObject.
SetActive(false);
     }
    for (int i = 0; i < enemy; i++){
    Enemy.transform.GetChild(i).gameObject.
SetActive(true);
            }
```

2 游戏中角色的设计

2.1 主角设计

2.1.1 主角血条

主角是只小鸭，初始生命值 Hearts 为 20，每吃到一个水果加 1，使用 Slider 的形式将生命值附着在小鸭的头部上方，滑动条根据时间的递减和吃到水果的数量增加和减少，血条的颜色是渐变的，通过在代码中改变 Image 组件的 color 属性即 gameobject.GetComponent<Image>().color 改变血条的颜色，在 Hearts 满格时呈现黄色，Hearts 减少到 75% 时呈现蓝色，Hearts 减少到 50% 时呈现灰色，Hearts 减少到 25% 以下呈现红色[1]，当 Hearts 减到 5 时，弹出 UI 文本框提示用户“生命值已严重不足”。

2.1.2 主角运动

用 transform.Translate 的形式给它一定的初速度使其开始运动；用 Input.GetAxis("Horizontal") 方法获取用户输入的横轴信息，此方法返回值若为 -1，则为水平向左，为 1，则为水平向右，为 0 则表示静止，同理 Input.GetAxis("Vertical") 用来获取纵轴信息，返回值为 -1 表示垂直向下，为 1 表示向上，为 0 表示静止，当输入方向向左且此时小鸭朝右时，小鸭转向，反之同理。

2.1.3 主角攻击

当用户按下右键或鼠标向右滑时，激活攻击功能。若此时主角没有死亡且有剩余子弹，就激活扔的动画，在主角的层级视图下建立一个空的 GameObject 用来表示小鸭发射子弹的位置，实例化子弹并用 AddForce 方法给子弹一个向前方向的力，主角所剩的子弹数减 1。

2.1.4 主角手势识别

手势识别是此游戏的亮点，引入手势识别可以丰富玩家的体验，方便用户操作的同时也使玩家对主角的控制更加自然流畅。首先建立枚举类型的变量 InputDirection 用来保存上下左右四个方向，然后获取用户的滑动方向，主要思路为将鼠标松开的位置坐标减去鼠标按下时的位置坐标得到滑动方向的向量[2]，然后判断向量的长度，给它一定的值以避免鼠标刚开始滑动就计算它的方向，这样做可以使滑动方向更准确，有了方向向量，直接利用 Mathf.Acos 计算反余弦的方法计算出与 XY 轴方向的夹角 var angleY = Mathf.Acos(Vector3.Dot(vec.normalized, Vector2.up)) * Mathf.Rad2Deg; 判断夹角的大小，若它与 Y 轴正方向角度小于 45 度，则判断为上划，若与 Y 轴正方向夹角大于 135 度，则判断为下划，X 轴方向亦同理。

已知滑动方向，就可以根据手势实现不同的动作，本游戏中上划代表主角跳跃，下划代表回到地面，左划代表匍匐前进，右划代表攻击发射子弹。

2.1.5 主角跳跃，匍匐和快走

若按上键或鼠标上滑主角跳跃，若当前主角在地面上，给它一个较小的重力值，利用 rig.AddForce 方法给它一个 Y 轴方向的力，播放跳跃动画使之跳跃，若松开上键或者向下滑，将它的重力值再设成正常值，使其可以快速落到地面上。此功能中关于判断主角在不在地面上，有多个判断方法，本游戏中使用的方法是射线检测 Physics2D.Raycast，在主角身上添加一个 CheckPoint 作为射线生成的起点，若返回值大于 0 则表示在地面上，否则说明不在地面上。

主角遇到障碍物时可以匍匐前进，设置两个碰撞体 Box 和 Circle，两个碰撞体交替执行实现匍匐。若按下下键或鼠标左滑时，将 Box 禁用，启用 Circle，否则若不在匍匐状态下就启用 Box，禁用 Circle。捡到某个道具时，速度突增，一段时间后恢复即为快走功能。使用协程函数保存现在速度，设置快走速度，修改状态为快走，时间递减结束后，恢复之前速度，修改状态。依此原理以为主角添加慢走，双倍积分等多种功能，丰富游戏内容。

2.1.6 主角死亡

当主角与敌人碰撞或被子弹击中时调用主角死亡函数，游戏结束。将布尔类型的参数 Die 设置为 true，防止被重复调用，小鸭死亡时利用 Collider.enabled = false 的形式将它身上的两个碰撞体 Box 和 Circle 禁用，速度设为 0，激活死亡动画使小鸭垂直掉落。

2.2 敌人设计

2.2.1 敌人控制

敌人运动与主角一样，当主角踩到敌人时，敌人死亡，实例化特效，同时利用 localScale 给敌人一个竖直方向的缩放，使敌人死亡看起来更直观，为敌人添加两个长方形碰撞体以限制它在一定范围内活动，利用协程函数使怪物 1 秒钟之后消失，当主角踩死怪物时，速度设为 0，给它一个向上的力使它弹起来。

2.2.2 敌人攻击

本游戏中敌人种类较多，例如口水怪发射子弹击中主角，喷火怪向主角喷火使主角死亡，主角触碰到碰撞体后会触发钢管怪，弹簧怪或大 Boss 时常出现，为游戏增加了难度，也增加了趣味性。口水怪的射击与主角的射击相同，当口水怪没有死亡且它还有剩余子弹时，实例化子弹并给它一个向前的力，与主角射击有所不同的是口水怪的射击是随机的，利用协程随机 1 到 5 秒内发射子弹，随机 1 到 2.5 秒内关闭发射，这样可以使主角的探险更加具有挑战性。

2.2.3 Boss 制作

游戏中还设置了终极 Boss，此怪物自身具有一定保护状态，同时会发射子弹精确命中主角，还会随机释放部分敌人，将它放在主角的右上角，与主角保持一定的距离 transform.position = new Vector2(controller.gameObject.transform.position.x + offsetPlayerX, transform.position.y); 同样这些功能都是在 Boss 没有死亡的状态下进行的。为它设置敌人数组，在等待一定时间后随机实例化数组中的一个，释放敌人，同时随机发射子弹。若此时 Boss 处于保护状态，无论主角怎么发射子弹都影响不了它的生命值，在特效数组中随机选取一个特效实例化，若不在被保护的状态，主角射击到它生命值减 10，与主角一样 Boss 头部上方也有一个血条来显示它

当前的生命值，若生命值小于 70% 时会呈现出一个冒烟的渐变效果，利用的也是在代码中改变粒子特效组件中 startcolor 参数的值。

```
SmokeEffect.GetComponent<ParticleSystem>().startColor = newColor(newColor,newColor,newColor);
```

2.2.4 Boss 的精确命中

此功能运用了一部分的数学功能，首先引出小鸭和 Boss 发射的子弹所在的位置，然后使他们的横坐标相减得出小鸭距 Boss 之间的距离 diatance，子弹的落地点放在 1/3distance 处，所以子弹落地点的横坐标即为小鸭的横坐标加 1/3distance，所以主角到达子弹落地点的时间即为 1/3diatance 除以主角的速度，这个时间同时也是子弹到达的时间，二者时间相同即可实现精确的命中。

```
var P1 = controller.transform.position;
var P2 = DropPoint.transform.position;
var distance = P2.x - P1.x;
var P3 = new Vector2(P1.x + 0.33f * distance, P1.y);
var time = (P3.x - P1.x) / (controller.speed / Time.fixedDeltaTime);
```

2.2.5 敌人死亡

对于生命值只有 1 的敌人来说，当主角踩死或者主角发射的子弹打到它们时，敌人就会死亡，播放死亡动画，但对于生命值有 100 的 Boss 来说，当主角在非保护状态下击中它 10 次即会死亡，释放死亡特效。

3 虚拟摄像机

本游戏引入了 Cinemachine 插件，加入了虚拟摄像机 CM vcam1，通过调整 Cinemachine Virtual Camera 组件的各项参数，实现摄像机平滑跟随的功能，同时可以在 Cinemachine Confiner 组件下的 Bounding Shape 2D 中引入范围，即在一定范围内实现摄像机跟随。

4 游戏测试

测试结果如图 2 所示：

图 2 游戏测试结果图

结论

本文使用 Unity3D 引擎开发了一款场景丰富的探险游戏，实现的功能有主角血条，主角运行，主角攻击，主角手势识别，主角各种技能，敌人的控制，攻击以及各种 Boss 的制作，特效动画的制作，游戏界面的设计等，结果表明游戏逻辑设置合理，玩家体验感较好，内容丰富有趣，是一款较好的休闲探险小游戏，能够满足人们的日常娱乐，锻炼玩家的注意力和灵活性，具有一定的应用价值。

【参考文献】

[1]陈阳 . 基于 Unity3D 的游戏开发 [J]. 电子技术与软件工程 ,2020(09):36-37.

[2] 程弘霖 , 杨键 , 唐娅雯 . 基于 Unity3D 的 VR 求生游戏的研究与实现 [J]. 信息与电脑 (理论版),2019,31(24):88-91.