基于 Unity 的道路 3D 跑酷游戏的设计与开发

董文静 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】nity3D 作为游戏开发引擎，开源、免费、灵活性的特性使它在游戏开发中多被选择使用。跑酷游戏是一种非常受欢迎的游戏，只要主角生命值大于零，游戏就可以一直进行下去，且在跑酷过程中，会有各种障碍物的随机出现和技能的拾取和展示，能激起玩家对游戏的兴趣。该文就是基于 Unity3D 开发引擎，设计并实现的一个跑酷类游戏，以便人们了解跑酷游戏等开发的常用方案。

【关键词】Unity3D; 跑酷；游戏设计

0 引言

国内游戏产业的飞速发展和手机的普遍性，使得大多数人都会在自己的休闲时间去选择玩会儿游戏放松一下，这也算是娱乐消遣的一种方式。本文就是基于 Unity3D 引擎工具设计和实现的一个跑酷游戏。在该设计中利用了人机交互的方式给用户带来更好的体验感。在该游戏的菜单栏中添加功能键使搭建道具场景更加方便，这也是该设计的一个亮点[1]。

1 游戏设计思路及流程图

设计思路就是当进入游戏运行状态时，点击 Play 按钮去开始游戏，在游戏过程中可以暂停游戏、恢复游戏，当主角碰到障碍物结束生命时，则会显示重新开始和退出按钮，玩家来决定后续操作。流程图如图 1 所示。

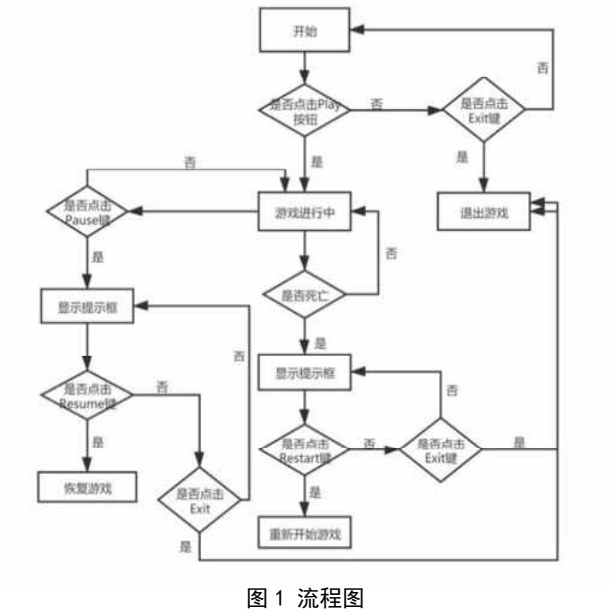

图 1 流程图

2 开发环境与软件

开发环境：Windows10 操作系统

制作软件：Unity 2018.3.9f1 Visual Studio 2019

3 游戏的设计和实现

3.1 游戏场景的制作

首先跑酷游戏的道路是没有尽头的，只要主角生命值大于 0，场景就会一直生成直至主角死亡结束游戏。

我们需制作出四段道路 start1 和 start2，road1 和 road2。start1 和 start2 道路是道具的拾取道路，此道路不设置障碍物；road1 和 road2 上设置一定的障碍物和技能道具，当主角跑到 road2 时，此时的 road1 会拼接到 road2 前面，走到 road2 时 road1 会拼接到 road2 的前面，以此来达到主角无尽跑酷的效果。

3.2 主角的移动及摄像机的跟随

3.2.1 主角的移动

在游戏过程中，主角是需要一直保持运动状态。为主角添加一个角色控制器组件，此项目中会用到此组件的 IsGround 和 Move 属性，我们就是通过函数中调用此组件的 Move 属性来使主角获取 z 轴的速度，使其向前运动起来。关键代码如下：

```
CharacterController characterController;
Vector3 xDirection;
Vector3 moveDirection;
moveDirection.z = speed;
characterController.Move((xDirection * 5 + moveDirection) * Time.deltaTime);
```

3.2.2 摄像机的跟随

为了使游戏运行起来更加有真实感，使玩家的体验感更强，要为主角添加一个摄像机跟随功能，会使游戏运行起来给用户更好的体验感。

为摄像机添加脚本，定义一个 GameObject 变量 target 为跟随的目标，定义距离主角的距离和高度来确定跟随主角的画面显示程度，定义一个 Vector3 pos 空间变量用来引用摄像机的位置，若当前游戏状态时 play 状态并且不在暂停状态时，摄像机跟随起作用，代码如下：

```
pos.x = target.transform.position.x;
pos.y = Mathf.Lerp(pos.y, target.transform.position.y+height, Time.deltaTime*6);
pos.z = target.transform.position.z - distance;
transform.position = pos;
```

3.3 手势识别及播放动画

3.3.1 手势识别

此处功能就运用了人机交互方式，用户通过对鼠标的上下左右滑动操作可以使主角实现向上跳跃、向下滚动、向左移动和向右移动的功能并播放其动画效果。

定义枚举类型的变量上、下、左、右和空，布尔类型变量 activeInput 表示是否按下左键，在脚本中写一个获取滑动方向的函数实现此功能，如果按下鼠标左键，activeInput 为真，记录当前的鼠标点击位置 mousePos，如果在当前按下鼠标左键且点击后随机滑动一定的方向松开时，此时用当前鼠标坐标减去鼠标按下时的坐标即可获得鼠标滑动方向的向量 vec，判断如果当前向量长度大于 25 时，计算当前向量与 Y 轴和 X 轴的角度。如果 angleY>45 及视为向上滑动、angleY>135 视为向下滑动、angleX>45 为向右滑动、angleX 为向左滑动。计算与 X、Y 轴角度代码如下：

```
var angleY=Mathf.Acos(Vector3.Dot(vec.normalized, Vector2.up))*Mathf.Rad2Deg;
var angleX=Mathf.Acos(Vector3.Dot(vec.normalized, Vector2.right))*Mathf.Rad2Deg;
```

3.3.2 识别后播放动画

为手势识别方向后的主角添加动画效果。所有的动画都是在动画控制器中设置逻辑触发，然后在脚本 AnimationManager 中去调用实现的，其他地方若使用动画函数则直接通过全局实例去跨脚本访问调用函数实现 [2]。

在脚本中定义一个声明代理，代理名称自拟，可以是无参的函数，再声明代理类型变量，使用时可以将一个函数名称赋给这个变量。

```
public delegate void AnimationHandler();
public AnimationHandler;
```

识别到哪个方向，就通过代理委托去调用实现此方向的函数。实现向左移动的函数代码如下：

```
AnimationManager.instance.animationHandler = AnimationManager.instance.PlayTurnLeft;
```

3.4 道具的制作

3.4.1 障碍物的制作

在主角跑酷的过程中，要在道路上设置一些障碍物使主角在游戏过程更有挑战性。一类是静止的障碍物，比如挡路的石头、阻碍的跨栏、静止的车子、木梯等；一类是动态的障碍物，比如向主角迎面驶来的汽车，我们只需给静止的车子添加代码，给车子赋予一定的速度即可使其运动起来。

将这些物体拖入场景中，分别为其添加碰撞体并勾选 is Trigger，并将其制作成预制体便于后面道具的摆放。为这些障碍物添加脚本，当主角触碰这些物体的触发器时主角死亡即结束游戏。

3.4.2 技能道具的制作和实现

为了丰富道路场景和游戏体验，在游戏中设置了多种技能道具。整个场景中的技能道具有主角可以吃金币增加积分、拾取磁铁道具吸附一定范围内的金币、拾取跑鞋可以二连跳跨越障碍物、拾取二倍积分道具则金币积分双倍和拾取星星技能道具可实现快跑的技能，这些道具功能的实现都有时间范围的约束。

将这些技能道具拖入场景中，分别为其添加碰撞体并勾选触发器，因为我们这些道具的拾取原理都是一致的，都是主角触碰到这些技能的碰撞体时，播放拾取音效，在主角位置处实例化拾取道具后的效果，所以我们将这些共有的实现功能放到一个 Item 脚本中，我们分别为这些技能道具添加同名的脚本，在这个脚本中，我们只需继承 Item 即可，然后我们要想实现具体功能，只需在技能脚本中复写函数即可。

以拾取到星星道具为例，我们实现 Item 脚本中播放音效，实例化效果的功能，具体的实现快速移动的我们则在复写函数中去调用 QuickMove() 函数，具体的实现函数我们在 PlayerCintroller 脚本中实现暴走，定义一个协程类型的变量 quickMoveCor；QuickMove() 函数的实现，如果此时的 quickMoveCor 不为空，即已有协程正在进行，则停止当前协程函数，开启新的协程函数。暴走协程函数的实现是默认拾取道具后，技能剩余时间即为技能的持续时间，如果当前不在暴走状态则用 saveSpeed 保存初始速度 speed 的值 5，否则设置速度为 20，将当的状态设置为暴走，当技能剩余时间大于等于零时，且当前游戏状态不在暂行再运行状态时，剩余时间递减，时间用完后将之前保存的初始速度再赋给 speed，再将暴走状态设置为假。其他技能的实现与此方法同类。

```
if(quickMoveCor != null) {
StopCoroutine(quickMoveCor);
 }
quickMoveCor = QuickMoveCoroutine();
StartCoroutine(quickMoveCor);
}
```

主角金币的拾取也是在 Coin 脚本中复写 HitItem 函数，再在其基础上去实现增加积分的功能；磁铁技能的实现就是为主角添加一个圆形碰撞体，设置一定的大小范围，在主角移动的过程中此范围的金币都会被主角吸附并增加积分。

3.5 场景道具的随机摆放

在主角跑酷的过程中，不仅是 road1 和 road2 上的交替呈现，且地形上的各种道具位置、种类也会变化。在此设计中我们则是提前设置多种不同道路上的道具摆放情况称之为道具库，在游戏过程中会随机的在 road1 和 road2 上生成道具库中的一种，这样确保在主角跑酷的过程中出现不会一直重复的场景道具，使游戏更加有惊喜感。

在脚本中使用一个类 Pattern 专门去装道具，及为道具库，然后在存放道具的类中再定义道具的种类及位置即可。再定义一个地形列表，我们就可根据我们定义的 Pattern 去随机选择一种道路生成。代码如下：

```
public List<Pattern> Patterns = new List<Pattern>();
public class Pattern{
    public List<PatternItem> PatternItems =
```

```
new List<PatternItem>();
    }
    public class PatternItem{
        public GameObject gameObject;
        public Vector3 position;
    }
```

每次都在Inspector面板上拖道具会非常麻烦，所以在菜单栏Windows下添加一个功能键AddPanternToSystem，我们在road1下面的Item中布置好的道具场景，直接点击即可将这些道具添加进我们道具库。

3.6 UI的制作

3.6.1 技能UI的显示

创建UI属性下的图片，然后将技能照片拖入即可显示，将四种技能放到一个空对象下，为空对象添加一个Vertical LayOut Group组件即可使这些图片数列排版，再分别为其添加文本框显示各种道具发挥作用的剩余时间。

3.6.2 功能键的制作

在整个游戏中会有Play、退出、暂停、重新开始和恢复等功能键，在unity中点击运行时，点击Play键则开始游戏，在游戏运行过程中点击暂停键则暂停游戏，此时会出现恢复和退出键，点击恢复则恢复游戏场景，点击退出则退出游戏，碰到障碍物主角结束生命时，会出现重新开始和退出键，点击重新开始则重新开始游戏，点击退出则退出游戏。

点击按键实现功能则要为对象添加Event Trigger组件，添加Click事件，将挂载此脚本的对象拖进去，然后选择实现此功能对应的函数。当点击Play键时，此时隐藏play键显示暂停键，进行游戏、播放音效。隐藏和显示按键则是通过iTween中的MoveTo函数来实现的，将隐藏在屏幕外的按键移动显示到屏幕中来。代码如下：

```
iTween.MoveTo(PlayUI,canvas.transform.position + new Vector3(-Screen.width / 2 - 500,0,0), 1.0f);
```

3.7 音频的制作

在一个游戏的设计中会用到大量的拟真音效来达到游戏与现实的真实效果，同时也会烘托整个游戏氛围使用户有身临其境的感受。

用一个AudioManager脚本专门实现音频的播放，定义一个播放音效的函数PlayAudio(Audio Clip)，通过调用时传进的片段名称即可在拾取道具的位置处播放相应音效。代码如下：

```
AudioSource.PlayClipAtPoint(clip, PlayerController.instance.transform.position);
```

4 测试及修改

点击unity中的运行，然后点击Play按键进行游戏测试，把游戏当中拾取道具的功能都过一遍，看是否出现错误提示，若出现红色预警则对代码进行进一步的修改。游戏测试界面如图2所示。

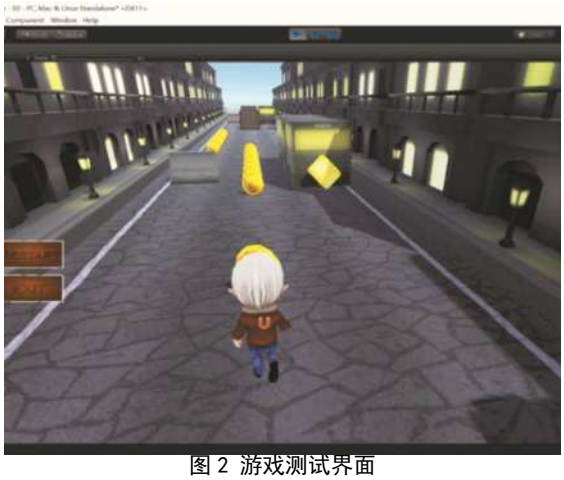

图2 游戏测试界面

结束语

在游戏设计过程中通过绘制流程图可以使整个游戏框架更加清晰，在制作过程中会更加直观的显示出来。由于Unity引擎开源、免费、灵活性的特点，所以在游戏开发领域也深受开发者的喜爱，该文就是通过Unity3D和C#脚本来设计和完成的一个跑酷类游戏，给用户带来无尽的冒险体验感。

【参考文献】

[1] 任建邦 . 基于 Unity3D 的手机游戏客户端的设计与实现 [D]. 北京交通大学 ,2013.

[2] 窦杨辉 , 王巍 , 何鑫 , 等 . 基于 Unity 的校园 3D 跑酷游戏的研究与开发 [J]. 计算机时代 ,2017(3):52-54.