走迷宫实验报告

- 计62 吴一凡 2016011269

代码结构

整个项目存放在`gvr-android-sdk-1.200\labyrinthvr`路径下。

游戏玩法

移动方式：按住Cardboard Trigger就会持续朝着眼睛看向的地方走过去，松开Cardboard Trigger会停在原地。

视觉交互：任何时候都可以通过摇头晃脑来调整视角。

游戏目标：你处在一个15×15的迷宫中，你要在迷宫中漫游，努力找到一扇木门并控制自己碰到这扇门，随后，你将会被传送到一个新随机生成的迷宫入口，并继续游戏。

配置方法

直接用Android打开`gvr-android-sdk-1.200`文件夹，编译或运行`labyrinthvr`项目。

实现亮点

- 对于给定的N，程序随机生成一个$(2N+1)(2N+1)$的迷宫，例如当$N=3$时，迷宫布局如下：

```
0000000
0 1 1 0
0101010
0 1 1 0
0101010
0 1 1 0
0000000
```

  初始状态下空格表示空腔，0表示不能被消除的墙壁，1表示可以被消除的墙壁，我们要消除一些1墙壁，从而使得所有的空腔连通。随后，游戏的目标变为从左上角走到右下角。

  在视线过程中，我们给所有1墙壁一个随机权值，做一个最小生成树，在树内的墙壁应被全部消除。而为了防止迷宫过于简单，不在树内的墙壁也有一定概率被消除，使得迷宫更具迷惑性。

- 连续移动的实现

  1. 首先通过实验注意到Cardboard的按钮按下相当于触摸屏幕，但GoogleVRSDK中只提供了OnTouch的接口，却并没有提供OnRelease的接口。解决方法是：通过重载`MainActivity`的`onTouchEvent`方法，使得可以检测Cardboard Trigger的一段连续的按下，并表现为持续移动了。
  2. 为了不走到墙壁里面，我们必须进行碰撞检测。 这里碰撞检测是基于二维线段判交实现的。在整个过程中我们一直保证视点高度(y坐标)不变，因此我们事

实上一直在一个平移后的$x-z$平面上移动。 在`onNewFrame`中，我们可以知道视点现在位置和下一帧将要到达的位置，这可以看作一条线段p_1。同时对于每个墙面，我们将墙向外平移一段距离称空气墙p_2。设定空气墙的目的在于不希望离墙面过近出现穿模现象。如果p_1, p_2相交，说明我们的移动穿过了空气墙，因此并不能移动到预定的位置，而会被卡在空气墙上。 如果已经被卡在空气墙上，则由于移动线段的起点已经在空气墙上，无论如何移动都不能奏效。此时我们进行特判，如果预定移动方向与墙面外法向点积为正(即希望离开空气墙)，则允许移动。

至此，在场景中的连续移动实现。

使用的库

本程序只使用了Android库、Java内置库、GoogleVRSDK，除此之外的代码都是自己实现的。其中与VR有关的库是：

`android.opengl.GLES20`，用来进行`OpenGL`绘制。

`android.opengl.Matrix`，用来进行矩阵运算。

`com.google.vr.sdk.base`，用来获取手机传感器的数据，并探测头部姿态，同时使得只需实现一个函数就能对双眼的画面进行绘制，并对两眼的画面进行合理布局完美适配Cardboard。

这些库的使用方法完全参考SDK中自带的`sdk-hellovr`项目。在这里就不详细列出接口了。

实现流程

1. 回顾OpenGL，阅读GoogleVRSDK样例代码
2. 实现基础场景搭建以及连续移动
3. 实现迷宫随机生成并完成最终场景搭建

本来还想做一些其他功能，但由于时间太紧没有时间去做了。

在文档目录下能找到完整的实验流程记录。

遇到困难

1. 理解OpenGL及样例代码原理花了很长时间
2. 写好碰撞检测后发现被卡在空气墙上，花了一段时间才想出解决方案
3. 一开始发现20×20的迷宫帧数过低，只有$11 \sim 12$帧，后来发现将纹理绑定到OpenGL的操作重复进行了很多，改成对于接下来要画的一批同样纹理的图元只进行一次纹理绑定，将帧数提高了50%。