分类号：TP311　　UDC：D10621-408-(2018)2319-0
密　级：公开　　编号：2014053064

成 都 信 息 工 程 大 学
学 位 论 文

“弘扬羌族文化”
羌族神话故事手游 app 开发

论 文 作 者 姓 名：钟以琛
申 请 学 位 专 业：数字媒体技术
申 请 学 位 类 别：工学学士
指导教师姓名（职称）：吴琴（讲师）
论 文 提 交 日 期：2018 年 5 月 10 日

“弘扬羌族文化”
羌族神话故事手游 app 开发

摘　要

羌族传统文化是中华文明的重要组成部分。然而随着工业化与全球化的冲击，以及 5.12 汶川特大地震的重大打击，羌族自然聚居地、羌族原生文化栖息地遭受重创，羌族传统文化面临着消亡的危险。

羌族无统一的文字，对于民族文化的传承仅靠心口相传，其无形性及难以保存性使得羌族在对其文化进行传承与延续方面存在较大困难；与此同时，对羌族非文化遗产的保护至今还缺乏相关法律法规的有力支持，因此目前对于羌族非物质文化遗产的保护仍面临巨大难题。

如何结合时代发展、从自身专业技能出发，有针对性地对民族文化资源的保护手段，提出可操作强的解决措施和具体项目，具有重要意义。本课题旨在结合数字媒体专业特点，利用电子游戏在交互性与传播性具有强大功能和潜质的特点，将游戏软件作为知识载体，以承载涵盖面广、种类丰富的羌族传统文化体系，对其进行数字化的资源保护。

本课题将通过实地调研、桌面研究等方式搜集、整理和筛选欲展现的羌族传统文化中的部分，同时根据文化知识部分，精心打造核心游戏玩法，将游戏、文化与教育相结合，设计一款以弘扬羌族文化为目标的，极具交互性、知识性与趣味性的手机游戏，并基于 Unity3D 引擎，遵循游戏产品开发流程，利用常见的游戏开发技术和性能优化技术，对产品进行开发与实现。本文主要从游戏开发的技术角度，试图在明确原则、呈现现状、分析问题的同时，针对性地提出一系列关于游戏技术的实现方案，并详细阐述核心功能的技术开发流程。

本课题将丰富的羌族文化元素引入计算机软件工程领域，通过设计弘扬羌族特色的手机游戏，使游戏、文化与教育相结合，将为上述多个领域带来良好的机遇的发展空间，促进游戏的教育性、教育的趣味性，以及羌族文化的弘扬与传承。

关键词：羌族文化；数字媒体；游戏开发；文化保护；教育

“Carry forward the culture of the Qiang nationality” Development of the Handicraft App of the Qiang People's Myth Story

Abstract

The Qiang traditional culture is an important part of the Chinese civilization. However, with the impact of industrialization and globalization, as well as a major blow to the 5.12 Wenchuan Earthquake, the Qiang nationality settlements and the native cultural habitats have been hit hard. The traditional culture of the Qiang people is in danger of disappearing. How to combine the development of the times, starting from their own professional skills, targeted protection of ethnic cultural resources, and putting forward actionable solutions and specific projects are of great significance.

One important way to protect intangible cultural heritage is to effectively spread and spread it. Combining with the status quo in the field of digital media, video games have great educational potential due to their characteristics in terms of interactivity, competitiveness, and challenges, and can be used as an efficient knowledge carrier to deliver rich information to users and have significant advantages. The role of popularization and dissemination. The introduction of a wealth of Qiang cultural elements, designing mobile games that promote Qiang characteristics, and integrating games, culture, and education, will provide good opportunities for development in these areas, and promote the educational and educational fun of games. , as well as the promotion and inheritance of the Qiang culture.

This paper is based on the combination of games and culture, education and other fields, and aims to promote the promotion and promotion of the Qiang culture. From the technical point of view of game development, it attempts to formulate a series of targeted statements while clarifying principles, presenting the status quo, and analyzing problems. Besides, it also introduce the implementation of game technology, and elaborated the technological development process of core functions.

Key words: Qiang Culture; Digital media technology; Game development; Culture saving; Education

目　录

论文总页数：32页

1 引言.....1
1.1 课题背景.....1
1.2 国内外研究现状.....1
1.2.1 文化资源的数字化保护.....1
1.2.2 教育游戏的设计与开发.....2
1.3 本课题研究的意义.....2
1.4 本课题的研究方法.....2
2 需求分析及项目方案.....3
2.1 需求分析.....3
2.1.1 功能需求.....3
2.1.2 系统约束条件.....3
2.2 项目方案.....3
2.2.1 项目特色.....3
2.2.2 游戏规则介绍.....4
2.2.3 Unity 开发技术简介.....4
2.2.4 ShaderLab 技术简介.....5
2.2.5 协同程序技术简介.....6
2.2.6 Unity 光照技术简介.....6
2.2.7 帧动画技术简介.....7
2.2.8 图集技术简介.....8
2.2.9 Git 技术简介.....8
2.2.10 Adb 技术简介.....9
2.2.11 开发工具和运行环境.....9
3 项目设计及实现.....10
3.1 功能模块的设计与实现.....10
3.1.1 系统整体框图.....10
3.1.2 动画模块的设计与实现.....10
3.1.3 UI 模块的设计与实现.....11
3.1.4 打地鼠游戏.....13
3.1.5 羌笛音乐游戏.....15
3.1.6 拼图游戏.....17
3.1.7 版本控制.....19

3.1.8 打包与安装.....19
4 特殊问题及解决方案.....20
4.1 光影效果实现问题.....20
4.1.1 问题描述.....20
4.1.2 解决方案.....20
4.1.3 结果.....22
4.2 DRAW CALL 优化问题.....22
4.2.1 问题描述.....22
4.2.2 解决方案.....22
4.2.3 结果.....23
4.3 UI 屏幕适配问题.....24
4.3.1 问题描述.....24
4.3.2 解决方案.....24
4.3.3 结果.....25
5 结果测试及性能分析.....25
5.1 测试概要.....25
5.1.1 测试环境.....25
5.1.2 测试范围.....25
5.2 性能分析.....27
5.2.1 CPU 使用率分析.....27
5.2.2 渲染分析.....27
结 束 语.....29
参考文献.....30
致 谢.....31
声 明.....32

1 引言

1.1 课题背景

在现代化与全球化的冲击下，羌族传统生活方式逐渐被新时代所颠覆，加之5.12 汶川特大地震，使羌族文化原生栖息地再遭重创，许多羌族传统非物质文化陷入濒临消亡的地步。然而羌族没有统一文字，对于非物质文化元素而言，缺乏重要的传承载体，传统文化世代仅靠人们心口相传，其无形性及难以保存性，为羌族人民对其传统文化的传承与延续，制造了重大困难。与此同时，对羌族非物质文化遗产的保护，至今仍缺乏相关法律法规的有力支持，法制的建设与普及速度尚未跟上对非物质文化遗产保护的迫切需要，因此对于羌族非物质文化遗产的保护，在当前还面临巨大难题。

关于如何对非物质文化遗产进行安全、有效得保护，如今已有部分国家与地区，开展了对非物质文化遗产进行数字化保护的相关项目的规划、研究与开发工作，并已取得部分突破性进展。结合数字媒体技术，对历史文化资源的保护举措展开新的探索与应用，已成为迫切需要深入挖掘且具有重要意义的新课题。

本课题以弘扬和传播羌族文化为主题，将文化知识和趣味游戏相结合，在对文化资源进行数字化保护的同时，也利用游戏为信息载体，对用户进行知识普及。在产品类型上，将文化保护、知识教育有机结合的游戏产品，在目前国内游戏市场中较为少见，在同领域内将具有一定的开创性；在载体形式上，利用计算机软件作为载体，既能充分提升历史文化资源保护水平，又能在完全不破损历史文化资源的条件下对其进行数字化保护和开发；在表现内容上，基于民族传统神话故事、传统服饰、风俗习惯等非物质文化遗产内容进行游戏开发，其潜在市场不仅局限于使用游戏产品的用户和玩家，还顺应了潜在用户对民俗景区、观光景点等旅游产品的娱乐需求，可与旅游业、文化产业等市场相结合，存在巨大商机。

1.2 国内外研究现状

1.2.1 文化资源的数字化保护

近年来，随着大数据的兴起，以及数字化在技术与市场的迅猛发展，人们逐渐开始越来越多地将历史文化资源的存储和保护与数字化联系在了一起。将无形的非物质文化遗产数字化，作为数据存储起来，对于其保护、传承与传播，都将有大幅改善和极高的价值。现如今，不少国家和地区陆续开始开展相关项目的研究与开发，尝试在该领域取得一些探索性的突破；国内也纷纷开展了对非物质文化遗产纳入发展规划中的措施，促进与支持历史文化资源数字化的发展进程。

国内外不断进行着数字化技术的历史文化资源保护与开发的探索与应用，其重要作用和前景已日趋显著和宽广。

1.2.2 教育游戏的设计与开发

西方国家在教育游戏，乃至整个游戏开发领域，起步较早，教育类游戏的设计与开发研究，对于知识教育类游戏的设计与开发，除了传统电子游戏的核心玩法设计之外，更重要的是如何将学习型与趣味性有机且平衡地结合起来，学习内容和游戏融合的方法，也一直是教育游戏设计开发领域的瓶颈难题所在。对于此类问题，只有为数不多的学者对整合教育知识类游戏的设计和开发模式做出过尝试，提出了部分方法模型。而对于游戏中的规则制定、剧情及关卡设计等的研究问题，却移植缺乏具有可操作性强、效果佳的方法论。

国内对于教育游戏的研究刚刚起步，乃至于对于游戏开发领域也落后于欧美国家几十年之久。最初人们主要是以处理和解决电子游戏，尤其是网络游戏所带来的社会问题为目的，试图将游戏与教育结合，以扬长避短。直至 2004 年开始，国内才出现关于教育游戏的设计模式与开发工作流的研究。之后尽管也不乏有学者提出相关的教育游戏开发模式，但大多数方法研究仍停留在较零散的理念和高度上，没有形成系统化、可操作性强、具有普遍指导意义的设计模式。

1.3 本课题研究的意义

“弘扬羌族文化——羌寨神话故事手游”项目以弘扬和传播羌族文化为主题，通过游戏将文化知识和趣味游戏相结合，在对文化资源进行数字化保护的同时，也利用游戏作为载体，对用户进行知识普及。

在产品类型上，将文化保护、知识教育有机结合的游戏产品，在目前国内游戏领域中较为少见，此类游戏在同领域内将具有一定开创性；

在载体形式上，利用计算机软件作为载体，既能充分提升历史文化资源保护水平，又能在完全不破损历史文化资源的条件下对其进行数字化保护和开发；

在表现内容上，基于民族传统神话故事、传统服饰、风俗习惯等非物质文化遗产的内容开发游戏产品，其潜在市场不仅局限于游戏产品的用户与玩家，还顺应了民俗景区、观光景点等旅游产品的需求，可与旅游业、文化产业等市场紧密结合，存在巨大商机。

1.4 本课题的研究方法

本课题将以 Unity3D 引擎作为开发平台，以 Android 为发布平台，以 C#作为功能脚本编程语言，以 ShaderLab 作为着色器编写语言，利用 Git 作为版本控制工具，使用 AndroidStudio 作为真机调试工具，同时利用 adb 作为软件包安装工具；本项目将把工作重点放在对游戏核心玩法的逻辑编写和对运行性能的优化方面，以上述工具、平台的使用作为技术路线，进行游戏软件的开发和实现。

2 需求分析及项目方案

2.1 需求分析

2.1.1 功能需求

本项目以羌族传统神话故事为主线进行展开，用户在体验游戏的过程中，将与大量的羌族民族元素进行交互。可满足用户对羌文化基本知识的了解，以及对羌族文化深入探索的需求；各类益智解谜类小游戏将被穿插在故事主线中，满足用户对游戏性、趣味性的需求。与此同时，本课题借助游戏作为表现形式，对文化类产品不易于传播的痛点提出了针对性强、可行性高的解决途径，将多方面课题融合在一起，从多个维度和民族文化的高度对现实问题进行思考与实践。

2.1.2 系统约束条件

社交分享功能：缺少后端开发人员，暂无社交分享类功能；

跨移动平台：缺少 ios 开发及测试设备，暂未导出至 ios 平台。

2.2 项目方案

2.2.1 项目特色

良好的平台移植性：Unity 使用了 CIL 指令集，可在任何支持.Net 框架下的环境运行，无需对不同平台进行分别部署。虽然目前受到各方资源限制，本游戏仅的 Release 版本仅支持在安卓移动平台下运行，但后期若有进行平台移植方面的需求，也无需另外繁琐的操作，在 Unity 下即可对其他各大主流平台进行打包适配。

便捷的版本控制：本项目使用 Git 作为版本管理工具，可方便简捷地对项目在不同时期的各个版本进行有效的控制。即使项目源码由于不可控因素导致丢失或被进行了删改，仍能完整地恢复到任意一个已被提交过的版本节点。

丰富的游戏性：多个益智类小游戏嵌入其中，游戏玩法丰富多变。本项目以羌族神话故事为游戏主线，同时串联起多个小游戏，其中涉及到纯游戏性内容的部分共由 6 个小游戏组成，包含打地鼠、知识问答、音乐游戏、拼图、迷宫及卡牌配对等，各游戏元素多变、玩法各异，使整个游戏项目在游戏性和可玩性方面内容丰富、趣味十足。

深刻的主题与现实意义：本项目以羌族传统文化为主题：将羌族传统文化与游戏结合，展示丰富的民族传统元素与文化知识。同时借助游戏作为表现形式，对文化类产品不易于传播的痛点提出了针对性强、可行性高的解决途径，将多方面课题融合在一起，对民族文化的传承问题进行思考与实践。在表现内容上，基于民族传统神话故事、传统服饰、风俗习惯等非物质文化遗产的内容开发游戏产品，在表现手法上，利用计算机软件作为载体，既能充分提升历史文化资源保护水平，又能在完全不破损历史文化资源的条件下对其进行数字化保护和开发。

2.2.2 游戏规则介绍

关卡	地点	神话故事	策划游戏	游戏规则	通关条件
1	萝卜寨	萝卜寨的故事	打坏人游戏	一分钟内，坏人和好人同时出没，打击到好人扣 10 分，打击到坏人加 10 分。达到一定分数获胜	规定时间内，分数超过 80 分获胜
			答题游戏	有选项按钮，当点击按钮正确时，会出现正确点击的声音，若不是，则会消失	全部选择正确获胜
2	黑虎寨	羌笛的传说	拼图游戏	点击 2 张不同位置的拼图会换位置，直到换到正确的位置为止	2 张拼图正确
			羌笛吹奏游戏	当玩家吹笛时开始计时，计时时间为一分钟；顺利击到吹笛的音符则加十分，未击到音符扣十分	吹笛分数不低于 150 分
3	石椅羌寨	羊角花的传说	迷宫寻找羊角花游戏	玩家通过灯光来照射到山洞的影子，30 秒内找到羊角花视为通过游戏	通关拿到羊角花
			翻牌配对羊角花游戏	两张牌为同色系的，为一对，点击到同色系的牌，则不转回去，若不配对，则会恢复到原来的模式	配对全部成功

注：本图引自参照文献[17]中表 4-1

图 2-1 游戏规则介绍

2.2.3 Unity 开发技术简介

Unity 是由丹麦团队 Unity Technologies 开发的一套跨平台的游戏引擎，提供了从游戏设计、开发和调试、多种插件与 SDK 集成、跨平台发布构建、测试到上线等一套完整的工作流，是一个将整套常见电子游戏的解决方案经过全面整合的综合型游戏开发工具。Unity 为游戏设计和开发人员等提供了以可交互的图形化编辑器为首要操作方式的开发环境，降低了游戏的开发门槛，使得不同职能的用户能够在匹配游戏需求的条件下进行便捷、高效的合作。

Unity 编辑器采用层级式的可视化交互结构，具有实时可视化编辑面板、动态游戏预览面板、详细的属性面板和项目文件结构面板。在编辑器内部，Unity

内置了地形编辑器，支持用户对游戏地图进行地形设计、植被贴图、建筑可视化等，同时也支持对实时三维动画、帧动画等类型的互动内容的综合创作。

Unity 使用了 CIL 指令集，可在任何支持.Net 框架下的环境运行，游戏脚本基于 Mono，可用 C#、Javascript 或 Boo 编写代码文件。

Unity 最受青睐的特点在于其优越的跨平台性，单机游戏方面，可导出至 Windows、MacOS、Linux 平台；手游方面包括 iOS、Android 平台；主机游戏方面，包括 PlayStation、XBox、Wii 等，Unity 还可支持 WebGL 开发网页游戏。

2.2.4 ShaderLab 技术简介

Unity 中的所有着色器文件均使用称为“ShaderLab”的声明性语言编写。它通过嵌入 Cg/HLSL 或 GLSL 程序片段的方式来实现着色器的编写。

在 Unity 中编写的 Shader 包括三类：

第一类，Fixed Function Shader：固定功能 shader。Fixed Function Shader 最简单的着色器类型，只能使用 Unity 提供的固定语法，通过指令来修改相关状态或指定属性。现已被弃用，在发布时将自动转换为顶点-片段着色器；

第二类，Surface-Shader：表面着色器。使用 Cg/HLSL 语言规范进行编写，常用于处理复杂光照效果，但简化了大量重复指令，是对顶点-片段着色器的进行了部分封装后的中间过程的 shader；

第三类，Vertex and Fragment Shader：顶点-片段着色器。自由度最高，选择性最大，因此可实现的效果最为丰富，同时也是最底层的着色器，使用难度最为复杂，需要手动处理相关的大量指令和过程。

无论是利用可视化 shader 制作工具，如 ShaderForge 插件，还是通过使用 ShaderLab 语言编写 shader，每个 shader 都需要为其指定名字，名字一般以目录结构的形式来指定，指定后即可通过外部在 Inspector 面板中显示，并以树状目录显示出来。

其次，ShaderLab 语法中最重要的语法之一是 Properties 块，在 Properties 块可以可选地声明一些当前 shader 中需要用到的属性，主要包括浮点数、整型数、颜色、变量和贴图等，在 Properties 块中声明的属性都可以在 Inspector 面板中进行具体数值的设置和修改。

ShaderLab 语法中另一个重要的关键点是 SubShader，每个 shader 中都至少需要一个 SubShader。在有多个 SubShader 的情况下，Unity 将根据当前 shader 运行环境的硬件性能，选择第一个性能条件允许执行的 SubShader，并执行渲染。

如果遍历所有的 SubShader 后，未能找到当前硬件性能允许执行的选项，则使用 Fallback 块中指定的默认 shader，Fallback 又称作回滚，其中包含的 shader 可称为备用 shader。因此，在编写 shader 时，应根据实际效果的炫丽程度，准确

地说是根据对设备硬件要求的苛刻程度，由高至低，排列 SubShader 的顺序，以达到在充分利用显卡性能的条件下，尽可能将 shader 效果完全展现的理想状态。

2.2.5 协同程序技术简介

协程是一个分部执行，遇到条件（yield return 语句）会挂起，直到条件满足才会被唤醒继续执行后面的代码。简单来说：程序内部可中断，然后转而执行别的子程序，在适当的时候再返回来接着执行。

协程不是线程，也非异步执行。协程和 Unity 生命周期中的 Update 函数都是在主线程中执行的，是一种对多线程的模拟机制。协程常用于延时一段时间后执行代码，或等其余某个操作执行完后再执行指定的代码。

在 Unity 的生命周期工作流中，Coroutine 在每一帧都会被处理，具体来说，Coroutine 中 yield 语句，即条件语句之前的代码，将在 LateUpdate()方法之前执行，而条件满足后的将被唤醒继续执行的代码，将在 LateUpdate()方法之后执行。对 Coroutine 的应用，除了作为延迟调用的实现机制，主要还可利用协程自身分部执行的特性，分部分来处理一些比较耗时的工作，如加载游戏资源，加载游戏资源的过程可能将持续较长的时间，但未必要等待所有资源都全部加载完毕后方才进入游戏操作，可利用协程，设置多重挂起条件，每满足一个条件就加载一部分资源，直至所有资源分部分、都以较小的耗时加载完毕，而期间的所有游戏循环并不会出现卡顿的现象。

2.2.6 Unity 光照技术简介

Unity 中的光照可以粗略地分为实时和烘培，并且，两种光照可以组合使用，以创建沉浸式的场景光照。Unity 光照系统可分为两大类：实时光照与烘培光照：

实时光照，Unity 中的灯光——平行光、聚光灯和点光源，是实时的。这意味着，它们为场景提供直接光照，并且每帧更新。如果灯光和游戏对象在场景中移动，光照将立即更新。在场景视图和游戏视图中都可以观察这种变化。

烘培光照，Unity 可以计算复杂的静态光照效果（使用称为全局光照或 GI 的技术），并将它们存储在一张称为光照贴图的纹理文件中。这个计算过程成为烘培。

Point Lights(点光源)：点光源位于空间的一个点上，并且均匀地向各个方向发出光线。 光线撞击表面的方向是从接触点回到光线物体中心的线。 强度随着距光线的距离而减小，在指定范围内达到零。 光强度与距光源距离的平方成反比。 这被称为“反平方律”，与光在现实世界中的表现类似。点光源可用于模拟场景中的灯和其他本地光源。常用于作为可照亮周围环境的荧光或爆炸光效。

Spot Lights(聚光灯)：像点光一样，聚光灯具有特定的位置和范围，光在该位置和范围内脱落。 然而，聚光灯被限制在一个角度，导致锥形的照明区域。 锥

体的中心指向灯光对象的前（Z）方向。 灯光的锥体边缘光线也会减弱。 加宽角度会增加锥体的宽度，并增加该渐变的大小，即“半影”。聚光灯通常用于人造光源，如手电筒，汽车大灯和探照灯。

Directional Lights(平行光/方向光)：类型太阳光，平行光可以被认为是无限远处存在的远距离光源。 它不具有任何可识别的光源位置，因此灯光对象可以放置在场景中的任何位置。 场景中的所有物体都被照亮，就好像光始终来自相同的方向一样。 光线与目标物体的距离没有规定，所以光线不会减弱。平行光可以用来模拟太阳或月亮。 在虚拟世界中，平行光可用来在不精确指定光线来自哪里的情况下为对象添加逼真的阴影。

Area Lights(区域光)：区域光由空间中的矩形定义。 光在所有方向上均匀地在它们的表面区域上发射，但仅从矩形的一侧发射。 区域光的范围没有手动控制，但强度在距离光源远处的反平方时会减小。 由于照明计算的处理器密集程度相当高，因此区域光在运行时不可用，并且只能映射到光照贴图中。区域光可用于创造一个逼真的路灯或靠近玩家的一盏灯。 小面积的光可以模拟较小的光源（例如室内照明），但具有比点光更真实的效果。

Unity 内置了 LightMapping，即光线映射技术，通过对光照图的预先计算生成基于三维引擎的光强数据，可加速游戏的渲染效率，也可避免游戏地图中的地形、建筑等物体在光照下发生变形。

2.2.7 帧动画技术简介

通过 Unity 自带动画编辑器，利用图片序列制作帧动画，并为动画状态的切换、过渡等提供方便的预览。

帧动画技术，就是通过在关键帧设置该帧对应的精灵图片，将多帧连接起来，则形成由图片序列组成的 2D 帧动画。一般使用方法如下：导入所需动画的精灵图片序列，通过 Unity 自带的动画编辑器，选取关键帧，并在该帧从导入的动画图片序列中选取并设置欲表现的单张精灵图片，完成对该帧的编辑。对每一帧做相同步骤的操作，再播放此动画剪辑，则形成由精灵图片序列组成的 2D 帧动画。

在制作完需要的动画剪辑后，不同的动画剪辑还需要进行切换，Unity 中通过 AnimationCotroller(动画控制器)来对同一物体上的不同动画的切换进行状态的管理和触发条件的设置，以规则的方式实现不同动画的切换和过渡效果。

在创建好动画剪辑及其对应的动画控制器后，还需要为物体添加 Animator(动画组件)，并将创建好的动画控制器属性添加至动画组件中。若不同的物体具有相同的动画切换逻辑，但是其实际动画表现又各不相同，则可创建一个 Animator Override Controller(动画重写控制器)，类似于一个动画控制器的派生类，通过继承具有目标切换逻辑的动画控制器，可对其中的每个动画进行重写，

实现用相同的切换逻辑控制具有不同表现的动画状态。

2.2.8 图集技术简介

图集（Atlas）也称作精灵图集或纹理图集，由若干张尺寸更小的纹理图片所形成的集合所组成。

图集主要用于以下几个方面：首先，对于每张子图来说，在打包成图集后，其周围的空白无用区域将被去除，在空间上可以大大减少游戏包体和内存占用；其次，多个子图被打包成一个图集后，所有子图图像都从同一张图集纹理中来获取，即同一图集中的所有子图的渲染请求，都可在同一次批处理中完成，这将大大减少 CPU 的运算时间，提高运行效率。

Unity2D 内置了一款官方插件 SpritePacker，用于将单张的 2D 图片分类、分组式地打包成图集。在设置栏的 Window 选项中找到 Sprite Packer，打开 Sprite Packer 面板。对图片打包成图集的具体操作而言，首先需要将导入的图片素材的 TextureType 设置为 Sprite(2D and UI)，由于是对单张图片进行合并处理，所以还需要将 Sprite Mode 属性设置为 Single，在以上两类属性设置完毕后，需单击 Apply 应用上述操作。接下来，为了良好的组织同类图片的结构，将需要合并至同一图集的各图片选中，并统一设置其 Packing Tag，Packing Tag 可称作打包标签，也类似于将要打包成的图集的名称，在 Sprite Packer 中可通过该属性找到指定的图集。在设置完打包标签后，即可在 Sprite Packer 面板中点击 Pack 按钮进行图集打包，打包后，所有源图片将被统一整合至一个新的图集中，所有源图互相紧密贴合，去除了原有的边缘间隙和空白、有效节省了占用空间。特别地，所有对图片的引用将保持原状，不被丢失，在编辑模式下，源图在打包过程前后并无任何差别，而在运行模式下，所有对已打包成图集的源图的读取均从图集中获得，故对同一图集的所有图片的渲染，都仅需要一次批处理即可。

2.2.9 Git 技术简介

Git 是一个分布式版本控制软件，可以对项目在不同时期，或同一时期来自不同开发者的不同版本进行管理与控制：如添加和提交、推送改动、分支的创建和处理、更新与合并、标签、版本回退、版本重置、查看版本差异等。利用 Git，即使项目源码由于不可控因素导致丢失或被进行了删改，仍能完整地恢复到任意一个已被提交过的版本节点。同时，Git 由于支持分支结构，为项目的多人合作开发提供了极大的便捷。开发者可随时创建一个新的分支，分支内可能仅包含一个小型功能模块的内容，但随时都可以通过切换分支，回到原来的工作进度并将新分支的内容合并进来。这种基于特征的工作流，能够为每个正在开发、测试或处理的新功能创建新的对应分支，并能够在其间来回切换，最后在希望合并功能的位置将其合并到主线上，再删除掉其余不再使用的分支。即使某个分支只是为

了做一些实验性的尝试，也大可不必担心它会对主线产生任何不必要的干扰，只需在不需要的时候将其删除即可。

2.2.10 Adb 技术简介

ADB 全称 Android Debug Bridge，是 Android SDK 中的一个开发工具。常用于对 Android 模拟机或 Android 真机设备进行应用的安装与调试等。

ADB 支持两种连接方式：局域网无线连接和 USB 连接。若要通过局域网连接移动设备，首先需进行网络设置，确保 pc 与移动设备同处一个子网。连接设备，用 usb 线连接 pc 与移动设备，执行相关命令并成功后，再拔掉 usb 线，即成功进行了局域网连接设置，再次连接 usb 线并执行相关命令，则可转回 usb 模式。

2.2.11 开发工具和运行环境

系统开发工具和开发环境如 2-1 所示。

开发工具名称	软件用途
Android 6.0.1	系统环境
Unity 2017.3.1f1 (64-bit)	开发平台
Visual Studio 2017	IDE
Android Debug Bridge	安装工具
Android Studio	调试工具
XMind	系统框图绘制工具
PhotoshopCS6 ImageMagick	美术素材处理工具
MonoBehaviour	IDE

表 2-1 开发工具表

本项目以 Unity3D 引擎作为开发平台，以 Android 为发布平台，以 C#作为功能脚本编程语言，主要用到 Visual Studio2017 和 MonoBehaviour 两款 IDE 进行 C#脚本的编写；以 ShaderLab 作为着色器编写语言，利用 Git 作为版本控制工具，使用 AndroidStudio 作为真机调试工具，同时利用 adb 作为软件包安装工具；对于美术素材的处理，用到 Photoshop CS6 及 ImageMagick 两款软件。以上述工具、平台的使用作为技术路线，进行游戏软件的开发和实现。

3 项目设计及实现

3.1 功能模块的设计与实现

3.1.1 系统整体框图

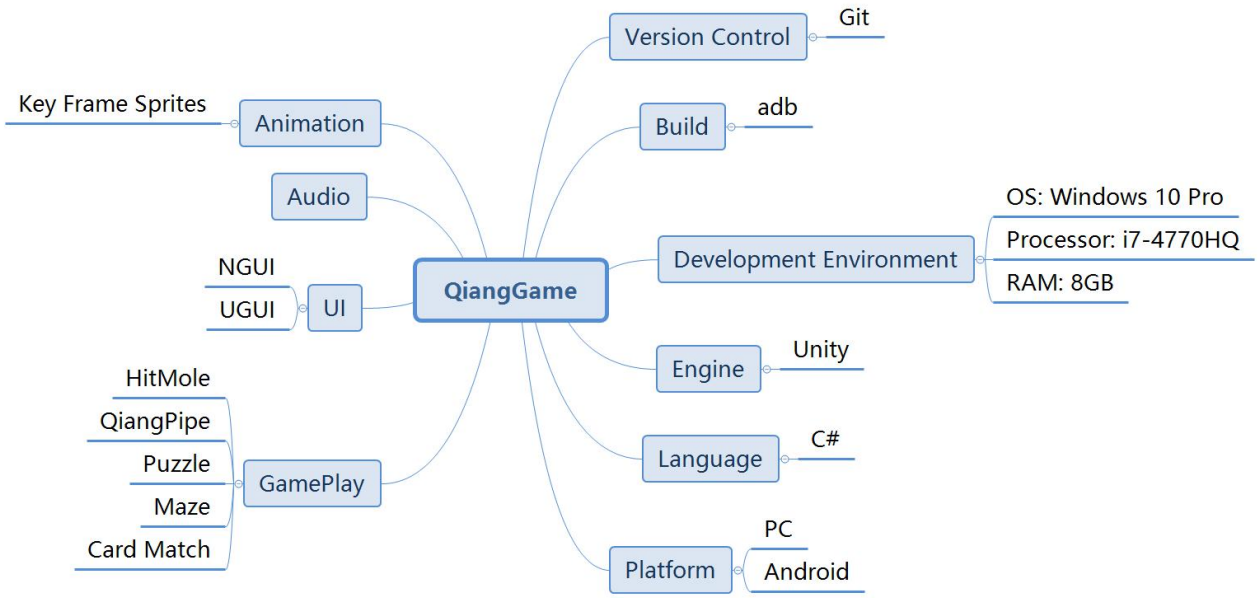

图 3-1-系统整体框图

3.1.2 动画模块的设计与实现

通过 Unity 自带动画编辑器，利用美术人员提供的帧动画图片序列，编辑和制作帧动画，并用脚本对动画状态进行控制。

动画模块的实现过程：第一，创建动画剪辑(Animation Clip)：在 Unity 中，为了实现动画功能，需要对物体添加 Animator（动画控制器），在 Animator 中设置一个或多个 Animation Clip，分别存储每个不同的动画状态及属性信息，再通过设置动画状态机，即可指定不同 Animation Clip 之间的切换条件；在 Animator 组件添加完成后，点击动画视图(Animation)选项，选择 Create New Clip，即可为当前所选的 GameObject 创建一个新的 AnimationClip。对 AnimationClip 指定存储为之后，若该 GameObject 已经附加了 Animator，则新的 AnimationClip 就将作为一个状态，被添加到现有的动画控制器中；其次，使游戏对象动起来：添加关键帧：点击动画记录按钮，进入动画记录模式，此时，对该游戏对象的修改会被记录到动画剪辑。组成图片序列：将添加的关键帧组成图片序列，点击播放按钮，则可预览当前动画剪辑的动画效果。

在制作完需要的动画剪辑后，不同的动画剪辑还需要进行切换，Unity 中通过 AnimationCotroller(动画控制器)来对同一物体上的不同动画的切换进行状态的管理和触发条件的设置，以规则的方式实现不同动画的切换和过渡效果。

关于动画间的触发条件，可在 Animator 面板中进行定义和设置。在 Animator 面板中，设置不同动画间的切换条件，其实也就是根据实际情况，通过声明不同类型的参数，并设置对应切换条件下的具体参数值来实现的。其中支持的参数类型共有四种：Float 浮点数、Int 整型数、Bool 布尔值及 Trigger 触发值。例如需要设置玩家从静止到跳跃的动画状态切换条件，不妨称静止动画为 Idle，跳跃动画为 Jump，则可设置一个 Bool 值 isGrounded 参数作为两种动画的触发条件，isGrounded 表示玩家是否正处于地面，当 isGrounded 为 true 时，静止动画向跳跃动画过渡，当 isGrounded 为 false 时，跳跃动画向静止动画过渡。

在创建好动画剪辑及其对应的动画控制器后，还需要为物体添加 Animator(动画组件)，并将创建好的动画控制器属性添加至动画组件中。若不同的物体具有相同的动画切换逻辑，但是其实际动画表现又各不相同，则可创建一个 Animator Override Controller(动画重写控制器)，类似于一个动画控制器的派生类，通过继承具有目标切换逻辑的动画控制器，可对其中的每个动画进行重写，实现用相同的切换逻辑控制具有不同表现的动画状态。

3.1.3 UI 模块的设计与实现

根据 UI 美术素材及 UI 流程图、原型图等，以 UGUI 为主，对界面进行布局、编辑与屏幕自适应；NGUI 为辅，实现 DoTween、iTween 等常用缓动效果与简单动画。

首先，最基础的 UI 元素包含以下视觉组件：文本组件、图像组建、原始图像组件和遮罩组件；又包含以下交互组件：按钮、开关；下面依次介绍其工作流：

文本组件（Text）：文本 Text 组件（也称为标签 Label）具有一个文本区域，用于输入将要显示的文本。可以设置字体、字体样式、字体大小，以及是否具有富文本功能；

图像组件（Image）：图像 Image 具有一个矩形变换组件 Rect Transform 和一个图像 Image 组件。图片精灵可以被应用于图像组件的 Target Graphic 域，颜色可以在 Color 域设置。材质也可以应用于图像组件。Image Type 域定义了图像精灵的显示方式。

原始图像组件(Raw Image)：图像组件 Image 接受一张图像精灵，而 原始图像 Raw Image 接受一张纹理（无边框等）。原始图像只应该在必要时使用，大多数情况下，图像更适合。

遮罩(Mask)：遮罩 Mask 不是可见的 UI 控件，而是一种修改控件子元素外观的方法。遮罩限制（则遮盖）子元素为父元素形状。因此，如果子元素大于父元素，那么只有位于父元素中的部分将是可见的。

按钮(Button)：按钮具有 OnClick 事件，用于定义点击后执行的行为。

开关(Toggle)：开关 Toggle 含有一个复选框，用于确定 Toggle 当前是打开还是关闭的。当用户点击 Toggle，它的值被反转，并且相应的打开或关闭视觉标记。它还具有一个 OnValueCHanged 事件，用于定义值被改变时执行的行为。

在 UGUI 中，凡是创建一个 UI 元素，若此时 Hirerarchy 面板中尚未创建 Canvas，则会自动生成 Canvas 及 Event System 两个物体，Canvas 是 UGUI 中所有 UI 元素的根节点物体，用于从全局定义、控制和管理 UI 布局，包括 UI 元素的锚点设置、间距大小、布局类型包括屏幕适配等等，都需要根据 Canvas 来进行具体管理，并受到 Canvas 组件的影响。Event System 物体用于响应用户与 UI 元素的交互事件，如对 Button 的点击、对 Image 的拖拽等，若无 Event System，则用户无法与 UI 进行任何有效的交互。

Unity 出色的跨平台性，使得同一款游戏能够在不同平台上完美运行，并还原其预期的视觉效果。而不同平台下的设备屏幕的物理尺寸、分辨率大小等不尽相同，要达到完美还原预期视觉效果的目的，就必须考虑到 UI 界面在不同屏幕分辨率下的自适应性。

游戏中的分辨率自适应主要包括在不同屏幕下整体画面的比例缩放，以及 UI 控件在屏幕宽高比不同的条件下的布局情况。针对画面的整体比例缩放，在 UGUI 中，可通过向 Canvas 物体添加 Canvas Scaler 组件，并设置该组件上的 UI Scale Mode 为 Scale With Screen Size，即根据屏幕实际尺寸进行整体缩放；而对于 UI 控件的布局模式，则是在 Canvas Scaler 组件上的 Screen Match Mode，即屏幕适配模式下，选择 Shrink，使 UI 控件布局根据运行设备的屏幕分辨率进行收缩。

3.1.4 打地鼠游戏

图 3-2 打地鼠游戏主界面

3.1.4.1 设计

2d 向 3d 视角的转换：通过对“地鼠”与“鼠洞”的图片切割，并更改其 sortingLayer 属性，表现出“地鼠”从“鼠洞”穿出的透视效果。sortingLayer 是 SpriteRenderer 组件中的一个重要属性，SpriteRenderer 只能添加至 2D 物体上，用于渲染该物体指定的精灵图片，通过给不同的物体设置相同的 sortingLayer，并在同层 sortingLayer 下改变 order in Layer 属性的值的大小，可在同一层级上改变不同贴图的渲染顺序。而不同的 sortingLayer，也遵循同样的规则，设置中靠前的 sortingLayer 渲染层级低，靠后的层级高。具体来说，本项目中对打地鼠游戏中 2d 向 3d 视角转换的实现方法，是将一张地鼠洞图片，切割为上下两半，上半部分称作图 A，下半部分称作图 B，将角色图片称作图 C，对于图 A、B、C 来说，其 sortingLayer 可设置为同层，不妨称作 Mole，在设置好同层 sortingLayer 后，将图 A 的 order in Layer 值设置为 0，图 C 对应值为 1，图 B 对应值为 2，三图的渲染顺序则为：B>C>A，即图 B 在最顶层，图 C 次之，图 A 在最底层，即实现了地鼠穿透鼠洞的 2d 向 3d 视角转换的效果。

生成规则："地鼠"每隔一段时间，随机从所有鼠洞中的某一个鼠洞位置出现。

触发条件：击打检测：通过用户输入（鼠标点击/手指触碰）确定击打位置，并通过击打位置判断此次击打是否有效；击打规则：若击打有效，则根据锤子击打到不同类型的"地鼠"，分别给出不同响应：打到好人，扣除相应分数；打到坏

人增加相应分数；若击打无效，不作反馈处理。

胜负判断：在一定时间范围内，分数是否到达胜利条件（一定分数值），若是，则游戏胜利，出现下一关入口；否则游戏失败，出现重玩入口。

3.1.4.2 实现

2d 向 3d 视角的转换：本项目中对打地鼠游戏中 2d 向 3d 视角转换的实现方法，是将一张地鼠洞图片，切割为上下两半，上半部分称作图 A，下半部分称作图 B，将角色图片称作图 C，对于图 A、B、C 来说，其 sortingLayer 可设置为同层，不妨称作 Mole，在设置好同层 sortingLayer 后，将图 A 的 order in Layer 值设置为 0，图 C 对应值为 1，图 B 对应值为 2，三图的渲染顺序则为：B>C>A，即图 B 在最顶层，图 C 次之，图 A 在最底层，即实现了地鼠穿透鼠洞的 2d 向 3d 视角转换的效果。

生成规则：对每只“地鼠”，先根据一定概率随机出其“地鼠”类型；

再随机出其出现位置——“鼠洞”序号；向上运动——钻出“鼠洞”；随后停留一段时间供玩家进行操作响应；向下运动——钻回“鼠洞”；回到步骤 1，开始下一轮行为；

触发条件：射线检测：获取用户点击屏幕的屏幕坐标，转换成世界坐标，再以该世界坐标点为起点，向 z 轴（正前）方向发出一条射线，并返回与该射线相交的游戏物体；击打判断：若返回的游戏物体是“地鼠”物体，则通过获取该被击打“地鼠”的 id 判断击打到哪一只“地鼠”。

胜负判断：定义有效时间、胜利标准分值，在有效时间内分数达到标准分值则获得游戏胜利，否则游戏失败。

核心代码：

```
    if (Input.GetButtonDown("Fire1") && (moveState == MoveState.MovingUp || moveState == MoveState.Stay)) {
        Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);
        RaycastHit2D hitInfo = Physics2D.Raycast(ray.origin, gameObject.transform.position);
        if (hitInfo.transform.gameObject == gameObject) {
            moveState = MoveState.MovingDown;
            if (ifGoodSprite == false) {
                spriteRen.sprite = hitMole[0]; scoreManager.score += 1;
            } else {
                spriteRen.sprite = hitMole[1]; scoreManager.score -= 1;
            }
        }
    }
```

3.1.5 羌笛音乐游戏

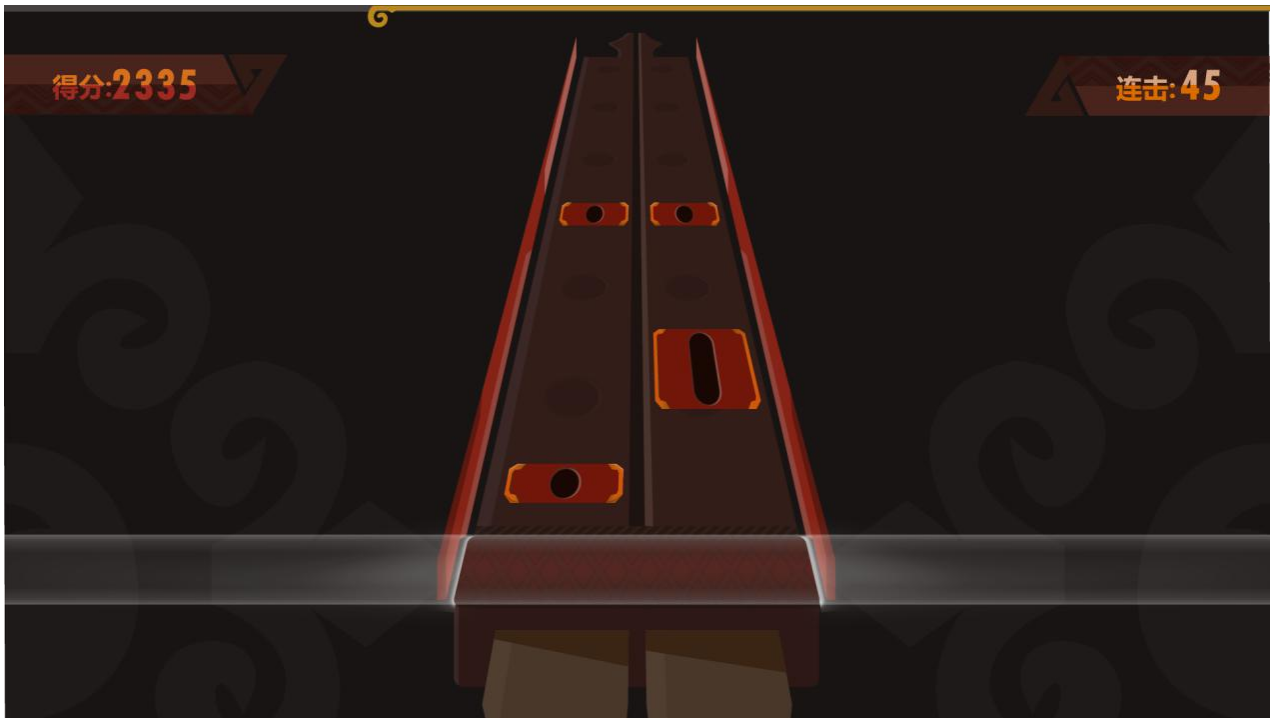

图 3-3 羌笛音乐游戏主界面

3.1.5.1 设计

2d 向 3d 视角的转换：由于游戏中音符由远至近运动，根据“近大远小”的透视原理，通过改变音符在距离不同状态下的尺寸与运动速度，实现透视效果。

生成规则：以各自对应的概率分别生成三种类型的音符：长单音符、短单音符、双音符；生成的音符自动持续向下运动。

触发条件：在音符位置进入击打判定区域后，检测用户输入，若在有效区内完成输入，则根据音符类型增加对应的分值。

胜负判断：在指定时间内得到一定分值，判定游戏胜利；在游戏胜利条件下，再根据分值进行等级划分，对玩家进行评价反馈。否则游戏失败。

3.1.5.2 实现

2d 向 3d 视角的转换：

```
//    scaling up every fixed interval
if (Time.time > nextGrow && !ifStop)
{
      nextGrow = Time.time + growTimeRate;
      ScaleGrow();
}

//    speeding up every fixed interval
```

```
        if(Time.time > nextSpeedUp)
        {
            nextSpeedUp = Time.time + speedTimeGrowRate;
            GetComponent<Rigidbody2D>().velocity = new Vector2(0f, -1 * moveSpeed *
Time.deltaTime);
            moveSpeed += speedValueGrowRate;
        }
```

如上图代码示例，每隔一定时间间隔(growTimeRate, speedTimeGrowRate)，分别对尺寸和速度(沿-y 轴方向)进行一定比例的增大，由于音符物体始终在朝-y 方向运动，随着时间的推移，位置由远至近，与此同时，尺寸由小至大、速度由慢至快，实现近大远小、近快远慢的效果。

生成规则：先计算概率，根据概率值，确定生成音符的位置 a. 左边 b. 中间 c. 右边；如果是 1.a 或 1.c，再确定生成音符的类型 a. 短音 b. 长音，如果是 1.b，则只有一种情况，即 c.双音确定了从某位置生成某类型的音符后，通过该类型音符预制体的引用将其实例化。

触发条件：碰撞检测：若在获取到用户按下指定按钮的瞬间，音符物体的碰撞体与得分判定区的碰撞体相交，则触发成功，根据当前音符物体的 tag 判定其音符类型，再加上对应的分数；若碰撞体不相交，则不进入触发成功的相应判断。

胜负判断：

```
    public void GameOver() {
        GetComponent<CreateNoteWave>().enabled = false;
        //  判断胜负与等级
        if (scoreValue > scoreValue_Low)      //  > 最低分
        {
            ifWin = true;    //  获胜
            if (scoreValue > scoreValue_Middle)       //  中等分段
            {
                gameLevel = GameLevel.scoreHigh;
                resultImage.sprite = scoreHigh_Image;
            }
            else   //      高分段
            {
                gameLevel = GameLevel.scoreMiddle;
                resultImage.sprite = scoreMiddle_Image;
            }
        }
        else   //     <= 最低分，为低分段，游戏失败
        {
            ifWin = false;
```

```
            gameLevel = GameLevel.scoreLow;
            resultImage.sprite = scoreLow_Image;
            nextLevelBtn.SetActive(false);
        }
        TweenPosition.Begin(resultPanel, resultPanelShowDuration, resultPanelMoveTo);
        resultScoreText.text = scoreValue.ToString();
    }
```

如上述代码，根据最终游戏得分，进行胜负与等级判断：若游戏胜利，即达到符合胜利条件的最低分，则再次进入等级的判断——胜利分段、中等分段、高分段；若未达到符合胜利条件的分数，则为低分段，游戏失败。

核心代码：

```
    void FixedUpdate()
    {
        if (ifStop)
        {
            GetComponent<Rigidbody2D>().velocity = new Vector2(0f, 0f);
        }
        //  scaling up every fixed interval
        if (Time.time > nextGrow && !ifStop) {
            nextGrow = Time.time + growTimeRate;
            ScaleGrow();
        }
        //  speeding up every fixed interval
        if(Time.time > nextSpeedUp) {
            nextSpeedUp = Time.time + speedTimeGrowRate;
            GetComponent<Rigidbody2D>().velocity = new Vector2(0f, -1 * moveSpeed *
Time.deltaTime);
            moveSpeed += speedValueGrowRate;
        }
    }
```

3.1.6 拼图游戏

素材处理：将整张拼图分割成 3*3 共 9 张地图碎片，以便对其进行分别控制。确定拼图分辨率，在 ps 中分别将拼图的长、宽进行 3 等分切割，共切割成 3*3 共 9 张尺寸相同的拼图碎片；

生成规则：玩家共需要解谜两块拼图，分别均由 9 块碎片组成；9 块拼图碎片在游戏开始时随机散落在一块 3*3 的区域内。相关数据结构包括：碎片列表(字段名：puzzleUnits<GameObject>)：共 9 个元素，每个元素表示一个地图碎片；位置列表(字段名：unitPosList<Vector2>): 存放一个 3 * 3 矩阵，共 9 个元素，

每个元素表示一个二维坐标值。生成碎片列表: puzzleUnits<GameObject>；生成位置列表: unitPosList<Vector2>；对 puzzleUnits 进行遍历，每次遍历中，从 unitPosList 中随机选取一个元素，将当前遍历的碎片在生成的随机位置上实例化，再将该位置元素从列表中移除，以免遍历不同碎片时取到同一位置元素。当遍历结束时，全部碎片均已取到其唯一对应的位置，并在该位置完成实例化。至此，拼图初始化完毕。

触发条件：将玩家前后点击的两张地图碎片交换位置。相关数据结构： 碎片-位置字典(字段名：unitDic<int, Vector2>)：该字典 key: (int) id, 表示该碎片的 id 号；value: (Vector2) position，表示对应碎片的当前位置坐标；当用户连续选择了两块不同碎片时，对已选的两块碎片进行位置交换；位置交换后，两块碎片在 unitDic 中所对应的记录即被更新。

胜负判断：玩家成功将两块拼图按正确顺序拼出，则游戏胜利。否则游戏无法向前推进。相关数据结构：unitDic<int, Vector2> 、unitPosList<Vector2>。胜负条件：当每块碎片的当前位置与其对应的正确位置相等时，即判断胜利；否则未成功(检测到任意一块碎片不满足该条件即表示尚未成功)。获取碎片的当前位置：通过该碎片的 id 号从字典中获取: unitDic<id>获取碎片的正确位置：id 号为 x 的碎片的正确位置即 unitPosList 下标为 x 的元素。如果 unitDic<id> == unitPosList<id>，则该碎片正处于正确位置，可进行下一轮遍历，否则游戏尚未胜利。

核心代码：

```
public void UnitSelected(GameObject unit) {
    if (isHandling) return; //  存在两个碎片正在交换位置，此时不能进行对其他碎片的操作
    selectedCount++;  //  若无碎片正在交换位置，则可操作当前选中的碎片
    if(selectedCount == 1)  //  为 1 时，表示当前只选中了一个碎片
    {
        selectedUnit_1 = unit;
    } else if (selectedCount == 2)      //  为 2 时，当前已选中了第二个碎片，可进行位置的交换与胜利的判定
    {
        selectedUnit_2 = unit;
        selectedCount = 0;          //  立刻置为 0，保证下一轮交换顺利进行
        isHandling = true;          //  开始对当前选中的两碎片进行位置交换，此时不能再操作其它碎片
         UnitPosExchange();         //  位置交换
     }
}
```

3.1.7 版本控制

Git 是一个分布式版本控制软件，可以在不同的时间或同一时期管理和控制来自不同开发人员的不同版本的项目：如添加和提交，推送更改，创建和处理分支，更新和合并以及标记、版本回滚，版本重置，检查版本差异等。使用 Git，即使项目源代码由于不可控制的因素而丢失，或者内容被修改，它仍然可以完全恢复到已提交的任何版本节点。

- 下载、安装 Git for Wind
- 本地 Git 与 Github 的连接
- 注册 Github 帐号
- 本地配置用户名和邮箱，命令如下：

```
git config --global user.name “ALEX-WHISPER<userName>”
git config --global user.email “15196608305@163.com<email>”
```

- 生成 ssh key：

```
ssh-keygen -t rsa -C “15196608305@163.com<email>”
```

- 将生成的 ssh key 复制到剪贴板，执行 clip < ~/.ssh/id_rsa.pub （或者进入资源管理器的指定路径对文件进行复制）；
- 将 ssh key 添加至 Github 服务器：
- 打开 Github，进入 Settings，点击左边的 SSH and GPG keys ，将 ssh key 粘贴到右边的 Key 里面，点击 Add SSH key。
- 创建远程仓库

```
git init
```

- 关联本地仓库：

```
git remote add origin <url>
```

- 将项目文件添加至本地仓库文件夹：

```
git add .
```

- 提交仓库更新内容：

```
git commit -m “update<commit content>”
```

- 将本地仓库上传至 Github 的仓库：

```
git push -u origin master
```

每次对项目内容进行修改后，将记录添加并提交，即可对项目版本进行便捷的管理。

3.1.8 打包与安装

在 Unity 内即可完成对 apk 文件的构建；利用 adb 工具进行 apk 在移动设备上的安装。添加所有需构建的场景，选择导出平台为”Android”，再选择”Build”

开始构建 apk；

安装：网络设置，确保 pc 与移动设备同处一个子网；连接设备，用 usb 线连接 pc 与移动设备，并开启开发者模式；打开 cmd，设置 adb：

```
adb tcpip 5555
```

通过 WLAN 模式连接设备：

```
adb connect 192.168.5.174<ip address>
```

若要转回 usb 模式：

```
adb usb
```

若在 WLAN 模式下连接成功后，可拔掉 usb 连接线：

安装 apk 至移动设备：

```
adb install .../.../QiangGame.apk<apk path>
```

4 特殊问题及解决方案

4.1 光影效果实现问题

4.1.1 问题描述

山洞内实时荧光效果的实现，类似实现一小块具有羽化效果的光晕并使其照亮周围一小部分区域。

4.1.2 解决方案

光源类型的选择：根据需求与参考图示，选取合适的光源作为光照类。

Point Lights(点光源)：点光源位于空间的一个点上，并且均匀地向各个方向发出光线。 光线撞击表面的方向是从接触点回到光线物体中心的线。 强度随着距光线的距离而减小，在指定范围内达到零。 光强度与距光源距离的平方成反比。 这被称为“反平方律”，与光在现实世界中的表现类似。点光源可用于模拟场景中的灯和其他本地光源。常用于作为可照亮周围环境的荧光或爆炸光效。

Spot Lights(聚光灯)：像点光一样，聚光灯具有特定的位置和范围，光在该位置和范围内脱落。然而，聚光灯被限制在一个角度，导致锥形的照明区域。锥体的中心指向灯光对象的前（Z）方向。 灯光的锥体边缘光线也会减弱。 加宽角度会增加锥体的宽度，并增加该渐变的大小，即“半影”。聚光灯通常用于人造光源，如手电筒，汽车大灯和探照灯。

Directional Lights(平行光/方向光)：类型太阳光，平行光可以被认为是无限远处存在的远距离光源。 它不具有任何可识别的光源位置，因此灯光对象可以放置在场景中的任何位置。 场景中的所有物体都被照亮，就好像光始终来自相同的方向一样。 光线与目标物体的距离没有规定，所以光线不会减弱。平行光

可以用来模拟太阳或月亮。 在虚拟世界中，平行光可用来在不精确指定光线来自哪里的情况下为对象添加逼真的阴影。

Area Lights(区域光)：区域光由空间中的矩形定义。 光在所有方向上均匀地在它们的表面区域上发射，但仅从矩形的一侧发射。 区域光的范围没有手动控制，但强度在距离光源远处的反平方时会减小。 由于照明计算的处理器密集程度相当高，因此区域光在运行时不可用，并且只能映射到光照贴图中。区域光可用于创造一个逼真的路灯或靠近玩家的一盏灯。 小面积的光可以模拟较小的光源（例如室内照明），但具有比点光更真实的效果。

综上所述，根据项目实际需求，应选用 Point Light(点光源)，创建一片可照亮周围一小部分环境的荧光效果。

着色器：编写 Surface-Shader(表面着色器)使游戏物体能够与光进行交互，从而产生光影效果。具体效果应从漫反射的属性作为切入点，参考 Unity 内置 shader：Sprite/Diffuse 进行编写。

漫反射着色器的编写：

```
        CGPROGRAM
        #pragma surface surf Lambert vertex:vert nofog nolightmap nodynlightmap keepalpha noinstancing
        #pragma multi_compile _ PIXELSNAP_ON
        #pragma multi_compile _ ETC1_EXTERNAL_ALPHA
        #include "UnitySprites.cginc"
        struct Input {
            float2 uv_MainTex;
            fixed4 color;
        };
        void vert (inout appdata_full v, out Input o) {
            v.vertex.xy *= _Flip.xy;
            #if defined(PIXELSNAP_ON)
            v.vertex = UnityPixelSnap (v.vertex);
            #endif
            UNITY_INITIALIZE_OUTPUT(Input, o);
            o.color = v.color * _Color * _RendererColor;
        }
        void surf (Input IN, inout SurfaceOutput o) {
            fixed4 c = SampleSpriteTexture (IN.uv_MainTex) * IN.color;
            o.Albedo = c.rgb * c.a;
            o.Alpha = c.a;
        }
        ENDCG
```

注意事项：完成着色器的编写后，创建材质并添加该着色器，再将新建的材质附着到目标物体上，添加点光源，即完成物体与光源的交互。但该着色器仍存在一些缺陷：光源作用方向始终面向正前方，因此光影效果只能在相机与物体之间的范围内才能得到表现。若发现物体漆黑一片，有可能是由于光源沿 z 轴的坐标小于物体的 z 坐标，使得物体处于光源光照范围之外。

4.1.3 结果

实现预期光影效果，如下图：

图 4-1 荧光光影效果图

4.2 Draw Call 优化问题

4.2.1 问题描述

美术素材均为单张精灵图片，对每张精灵图片都分别作单独处理将产生不必要的空间浪费与性能损耗。

4.2.2 解决方案

减少空白区域：在进行 2D 贴图时，比较直接和简单的方法是分别对每个需要贴图的游戏物体都赋予一张单独的精灵图片。但对于单张精灵图片，其内部常存在一些无用的空隙，这些空隙将造成运行时空间浪费。为了获得更佳性能，最好能够将多个精灵图片紧紧包装在一起，形成一个 Atlas(图集)，可有效减少无用的空白区域。

减少 Draw Call：在形成图集之后，对图集中的任意一张精灵图片，都只需要统一从该图集中获取。若打包前的精灵图片共有 n 张，打包后生成一张 Atlas

图集，则打包前后可降低 n-1 次 Draw Call。

综上所述，可采用 Sprite Packer 插件对需要处理的多个精灵图片打包封装为一个 Atlas(图集)。

在设置栏的 Window 选项中找到 Sprite Packer，打开 Sprite Packer 面板。对图片打包成图集的具体操作而言，首先需要将导入的图片素材的 TextureType 设置为 Sprite(2D and UI)，由于是对单张图片进行合并处理，所以还需要将 Sprite Mode 属性设置为 Single，在以上两类属性设置完毕后，需单击 Apply 应用上述操作。接下来，为了良好的组织同类图片的结构，将需要合并至同一图集的各图片选中，并统一设置其 Packing Tag，Packing Tag 可称作打包标签，也类似于将要打包成的图集的名称，在 Sprite Packer 中可通过该属性找到指定的图集。在设置完打包标签后，即可在 Sprite Packer 面板中点击 Pack 按钮进行图集打包，打包后，所有源图片将被统一整合至一个新的图集中，所有源图互相紧密贴合，去除了原有的边缘间隙和空白、有效节省了占用空间。特别地，所有对图片的引用将保持原状，不被丢失，在编辑模式下，源图在打包过程前后并无任何差别，而在运行模式下，所有对已打包成图集的源图的读取均从图集中获得，故对同一图集的所有图片的渲染，都仅需要一次批处理即可。

4.2.3 结果

打包前

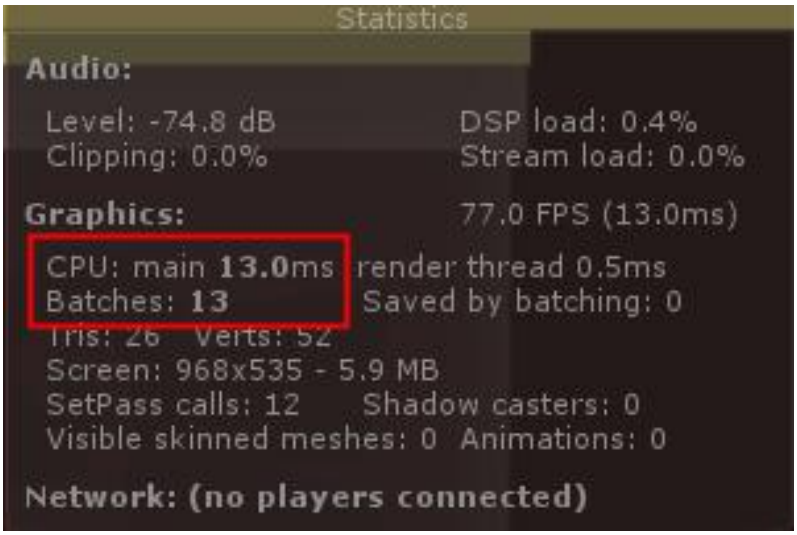

图 4-2 打包前-Batches: 13

打包后

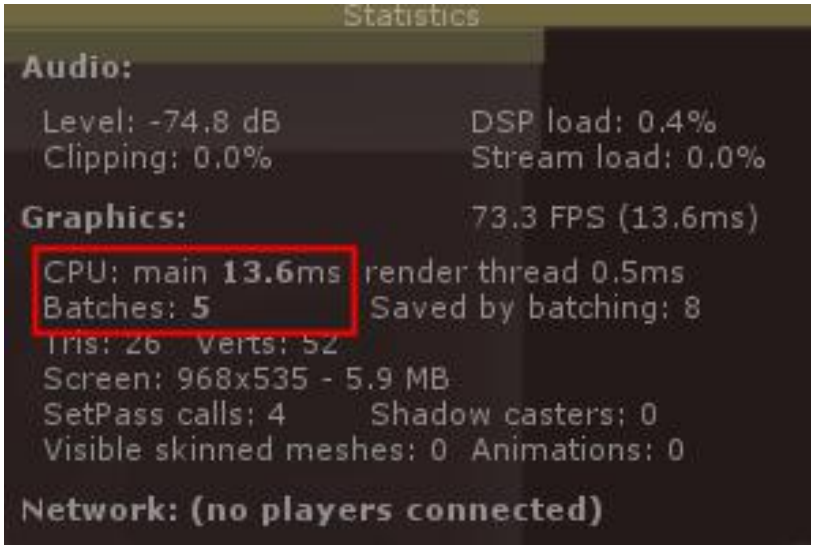

图 4-3 打包后-Batches: 5

对比结论：由上图对比分析可得出结论：将多张单独的精灵图片经过 Sprite Packer 打包后，Batches 数量，也即 Draw Call 的次数明显减少，有效地降低了性能损耗。但由于本项目规模较小，Draw Call 数量上限较低，即便数量明显降低了，对 CPU 耗时的影响也十分微弱。因此上述两图中，CPU 耗时差异不大。不过性能优化是必要需求，即使在性能优化部分，本项目并非典型示例，也应着重指出、深入理解。

4.3 UI 屏幕适配问题

4.3.1 问题描述

UI 布局在不同分辨率的终端上显示不一致，若将游戏分辨率切换至另一不同于初始设置的分辨率，则 UI 布局杂乱或不可见。

4.3.2 解决方案

方案 1：确认相机引用：调整 Canvas 的 Render Mode 属性为 Screen Space - Camera，并将游戏主相机拖入 Render Camera 属性中；

屏幕自适应：在 Canvas 物体中，查看是否有 Canvas Scaler 组件，否则需添加该组件；在 Canvas Scaler 属性中，需要将 UI Scale Mode 属性设置为 Scale With Screen Size 模式，表示 Canvas 中的 UI 布局将会根据屏幕比例自动进行缩放以适应当前分辨率；

适配：Screen Match Mode 属性中，选则 Match Width Or Height，表示采用宽度或高度（默认适配方式）进行屏幕适配。接着，将 Match 调整为 0（完全宽度适配）或 1（完全高度适配），其他值表示介于两者之间采用对应的比例进行宽高适配。

方案 2：调整 Canvas 的 Render Mode 属性为 World Space；将 Event Camera 引用至游戏主相机 MainCamera；调整 Rect Transform 组件中的 Width 和 Height 为设计尺寸的宽和高，同时将 Scale 属性的 X 和 Y 都调整为 0.01（对应 unity2d 默认情况下像素 Pixels 与引擎单位 Unit 对应比例 100）。

4.3.3 结果

UI 元素可根据当前设备分辨率进行自适应布局。经打包测试，打包在 Android6.0.1 版本、主屏分辨率为 1920×1080 的移动设备上，UI 布局与 Unity 编辑器下的预览图基本无异；同时，打包在 Windows 平台上的版本，同样能够根据 PC 屏幕实际分辨率按比例进行合适的 UI 布局。

5 结果测试及性能分析

5.1 测试概要

5.1.1 测试环境

操作系统：Windows 10 Pro

处理器：Intel(R)Core(TM)i7-4770HQ

内存：8G

显卡：核显 Intel(R)Iris(TM) Pro Graphics 5200；独显 NVIDIA GeForce GTX 960M

5.1.2 测试范围

功能测试

“打地鼠”游戏逻辑功能正常：正常获取用户输入并作出及时响应；正常生成随机角色并进入指定运动模式；游戏触发条件正常执行，主要包括：分数变化、交互响应、游戏胜负判断等；

“羌笛音游”逻辑功能正常：正常获取用户输入并作出及时响应：当音符运动至击打判定区域时方才进行有效的输入判断，当且仅当用户进行有效输入时，才作出音符被击打的检测判断；正常生成音符物体并能根据透视原理进行对 3D 视角的模拟运动；符合指定逻辑的事件触发，主要包括：分数变化、胜负判定、等级判定等；

“拼图”游戏逻辑功能正常：正常获取用户输入并作出及时响应：获取前后选中的拼图碎片并交换其位置；正常生成两组拼图碎片并分别组成完整的拼图面板；符合指定逻辑的胜负判断：当且仅当所有拼图碎片都被排列到其唯一指定位置时，才判断游戏胜利；

“迷宫”游戏逻辑功能正常；正常获取用户输入并作出及时响应：用户通过移动(方向键/触屏滑动)输入时，人物基于用户输入进行准确移动；正常的光照效

果：光源与人物、山洞地图及道具等进行自然的光线交互；触发检测正常进行，当人物抵达终点时，触发游戏胜利判断；

“卡牌配对”游戏逻辑功能正常：正常获取用户输入并作出及时响应：前后点击的卡牌在匹配与不匹配的条件下，均能正常触发各自指定的交互效果；符合逻辑的胜负判断：当且仅当所有卡牌都两两配对后，才判定游戏胜利；

特性测试

输入方式：根据平台差异性正常区分。在 PC 下测试，通过键盘正常获取用户输入；在 Android 下测试，通过用户对屏幕的操作正常获取用户输入；具体来收，在 PC 环境下（本项目 PC 环境为 Windows Standalone），调用的是 Input.GetKeyDown()或 Input.GetAxisRaw()等获取键盘输入的 API, 在移动设备下，脚本中通过调用 Input.Touch()获得触屏手势输入数据。

UI 布局：游戏中的分辨率自适应主要包括在不同屏幕下整体画面的比例缩放，以及 UI 控件在屏幕宽高比不同的条件下的布局情况。针对画面的整体比例缩放，在 UGUI 中，可通过向 Canvas 物体添加 Canvas Scaler 组件，并设置该组件上的 UI Scale Mode 为 Scale With Screen Size，即根据屏幕实际尺寸进行整体缩放；而对于 UI 控件的布局模式，则是在 Canvas Scaler 组件上的 Screen Match Mode，即屏幕适配模式下，选择 Shrink，使 UI 控件布局根据运行设备的屏幕分辨率进行收缩。在 PC 与 Android 不同环境下进行测试，UI 元素可根据实际屏幕分辨率进行自适应布局。在不同分辨率下，UI 元素可根据实际情况保持相对位置和宽高比例不变。

5.2 性能分析

5.2.1 CPU 使用率分析

WaitForTargetFPS：当前帧的 CPU 等待时间。耗时比例：72.0%，总耗时：10.80ms；

EditorOverhead：编辑器开销，表示编辑器本身所需的资源对性能的影响。耗时比例：16.4%，总耗时：2.46ms；

Physics.Simulate：当前帧物理模拟的 CPU 占用时间。耗时比例：0.6%，总耗时：0.09ms；

Camera.Render：相机渲染准备工作的 CPU 占用量。耗时比例：3.1%，总耗时：0.47ms

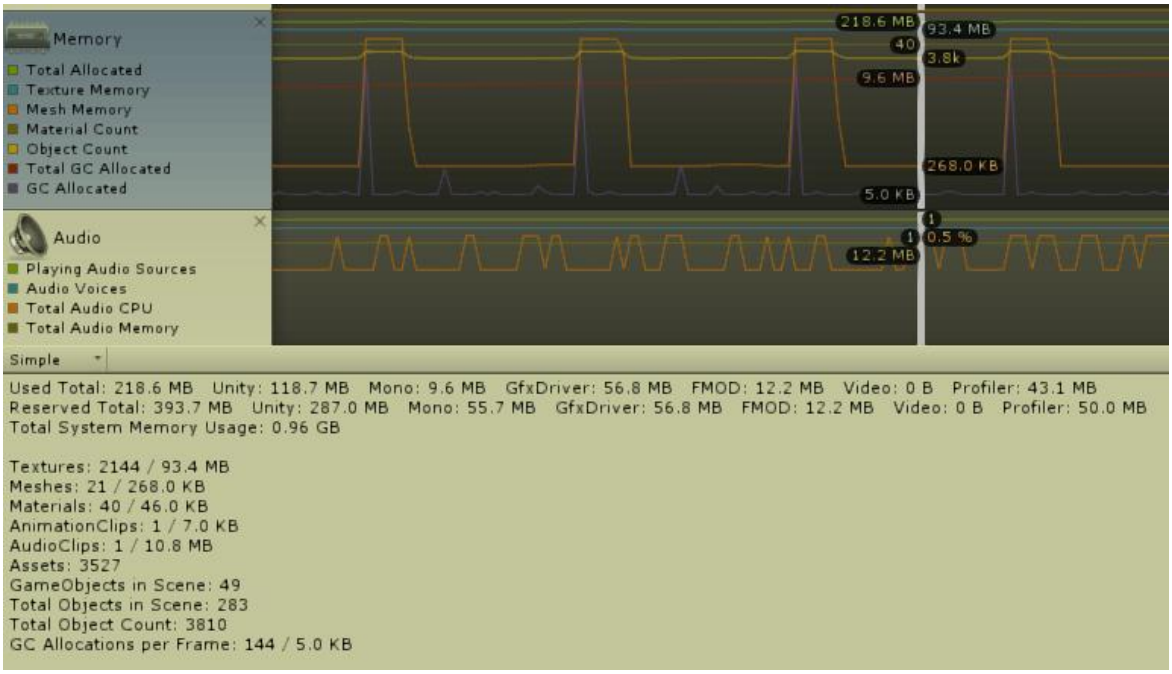

图 5-1 CPU 使用率分析

5.2.2 渲染分析

Draw Calls：应用批处理后网格绘制的总数。次数：13；

Batches：一次 Draw Call 即对应一个 Batch。次数：13；

Used Textures：绘制该帧使用的纹理数及其使用的内存。纹理数：14，使用量：33.2MB；

Tris and Verts：绘制三角形和顶点的数目。三角形数目：104，顶点数目：188；

VRAM Usage：解释：当前状态下被消耗的显存(VRAM) 的大约范围。范围：11.7MB~46.4MB；

VBO total：传入显卡的顶点缓冲区的数量。数量：643，使用量：1.5MB。

综合以上数据可知，在测试数据所属的当前帧内的总耗时中，编辑器开销占比 16.4%(Release 版本不占用任何编辑器开销)，耗时 2.46ms；物理引擎开销占比 0.6%，耗时 0.09ms；相机渲染占比 3.1%,耗时 0.47ms。游戏渲染部分，Draw Call 与批处理次数均为 13 次，该帧所渲染的纹理数量为 14，占用内存 33.2MB；绘制的三角形数目为 104，顶点数目为 188；该状态下消耗显存范围为：11.7MB 至 46.4MB；定点缓冲区数量为 643，占用内存 1.5MB。

综上所述，本项目资源重复率低，Draw Call 峰值低、趋势平缓，顶点数量峰值不超过 200，无超标资源数，项目总体体量较小，优化空间不大，性能良好。

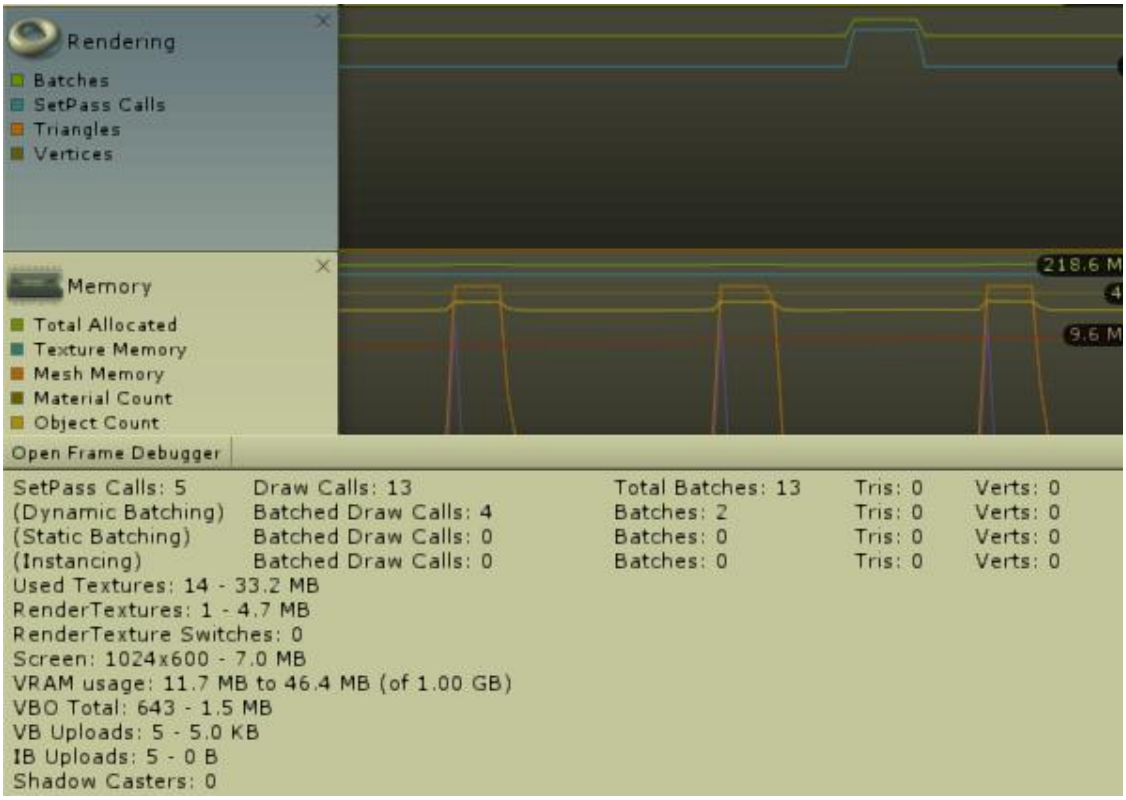

图 5-2 渲染效率分析

结 束 语

本课题通过实地调研、桌面研究等方式搜集、整理和筛选欲展现的羌族传统文化中的部分，同时根据文化知识部分，精心打造核心游戏玩法，将游戏、文化与教育相结合，设计一款以弘扬羌族文化为目标的，极具交互性、知识性与趣味性的手机游戏，并基于 Unity3D 引擎，遵循游戏产品开发流程，利用常见的游戏开发技术和性能优化技术，对产品进行开发与实现。

本课题最终实现的游戏产品，串联起了 3 个羌族传统神话故事作为游戏剧情主线，内含 29 个独立场景、5 个益智类小游戏，在游戏世界中再现了多处羌寨建筑实景与民族传统器具等元素，利用羌族文化中丰富的民族元素作为知识内容，在趣味性、教育性与传播性并存的条件下，将知识与游戏有机结合起来，促进了对羌族文化的弘扬与传播，丰富了文化资源的保护与开发手段，并为游戏与教育的结合提供了较好的范例。

参考文献

[1] 宋敏珠.国内外教育游戏的设计与开发研究综述[J],科技信息,2011,88(19): 179-180.

[2] 蒋彬,张原.羌族传统文化的保护与发展研究[J], 西南民族大学学报,2009, 16(212): 19-24.

[3] 李绍明.汶川大地震后羌区文化重建问题[J],西南民族大学学报,2008,2(5): 9-9.

[4] 冉光荣,李绍明,周锡银.羌族史[M],成都:四川民族出版社,1995,8.

[5] 周毓华.羌族的非物质文化遗产现状研究[J].西藏民族学院学报,2008,29(3): 61-65

[6] 邹循进,叶云青,陈孝威. C++游戏开发中可重用动态有限状态机设计与实现[A].第三届和谐人机环境联合学术会议(HHME2007):论文集[C]. 2007

[7] 姚挺.中国网游“游”向何方?——娱教技术支持下的网络游戏设计[A].教育技术:信息化阶段新发展的研究[C]. 2007

[8] 陈雪梅.基于 Unity3D 的手机游戏开发[J],电子技术与软件工程,2016,51(23): 71-72

[9] 尚俊杰,裴蕾丝. 重塑学习方式：游戏的核心教育价值及应用前景[J],中国电化教育, 2015, 12(5):40-49

[10] 凡妙然. 国内教育游戏研究现状的可视化分析:热点与趋势[J], 现代远距离教育,2018, 4(2):27-34

[11] 赵东. 数字化生存下的历史文化资源保护与开发研究[D]. 济南: 山东大学[博士论文], 2014.

[12] 王龙,李韬伟,杨振发. 游戏引擎研究与分析[J], 软件导刊,2018, 41(2):5-7.

[13] 阎梦真,陈宏利. 3D 游戏开发技术设计与应用[J], 中国新通信,2016, 29(11):85-85

[14] 刘文辉,王艺亭,赵敏,胡贺宁,江丰光. 教育游戏评价指标的设计与开发[J],开放教育研究,2017, 23(2):111-120.

[15] 赵慧臣,王玥,赵琳,严文彬. 少数民族双语教育信息化研究现状与展望[J], 现代教育技术,2015, 39(10):12-18.

[16] 钟令青. 中国游戏设计专业教育现状和发展策略[J], 艺术科技,2016, 11(3):402-402

[17] 朱慧群. “弘扬羌族文化”羌族神话故事手游 app 的策划及 UI 设计[D], 成都: 成都信息工程大学[学士论文], 2018.6.

致　　谢

在论文完成过程中，首先，我要感谢的是我的导师吴琴老师，她对我的研究提出了很多宝贵的意见，使我写作论文有了目标和方向。吴琴老师严谨的治学态度和渊博的学识，朴实无华及平易近人的人格魅力对我影响深远，是我一生都值得学习的榜样！

其次要感谢所有教过我的老师们，他们循循善诱的教导给了我无尽的启迪，非常感谢老师们！

最后我要感谢四年中陪伴在我身边的同学、父母和朋友们，谢谢他们为我提出的所有有益的建议和意见，感谢他们在论文写作过程中帮助我查找资料，使我顺利完成论文写作，感谢他们对我的生活提供的支持和帮助！

最后向在百忙之中评审本文的各位专家、老师表示衷心的感谢！

作者简介：

姓　　名：　钟以琛　　性别：　**男**

出生年月：　1996 年 3 月 12 日　　民族：　汉

E-mail：　15196608305@163.com

声　　明

本论文的工作是2017年 10 月至2018年5月在成都信息工程大学计算机学院完成的。文中除了特别加以标注地方外，不包含他人已经发表或撰写过的研究成果，也不包含为获得成都信息工程大学或其他教学机构的学位或证书而使用过的材料。

关于学位论文使用权和研究成果知识产权的说明：

本人完全了解成都信息工程大学有关保管使用学位论文的规定，其中包括：

（1）学校有权保管并向有关部门递交学位论文的原件与复印件。

（2）学校可以采用影印、缩印或其他复制方式保存学位论文。

（3）学校可以学术交流为目的复制、赠送和交换学位论文。

（4）学校可允许学位论文被查阅或借阅。

（5）学校可以公布学位论文的全部或部分内容（保密学位论文在解密后遵守此规定）。

除非另有科研合同和其他法律文书的制约，本论文的科研成果属于成都信息工程大学。

特此声明！

作者签名：

2018 年 5 月 9 日