优化排查思路

# 测试机

中高端机：一加手机７Pro，桌面35℃，相册40℃，超过60℃算发烫严重

中低端机：华为Mate10 Pro，桌面27℃，相册32℃，超过55℃算发烫严重

华为P20 Pro，测试包40℃

# 目标

|  |  |  |
| --- | --- | --- |
| 目标 | A | B |
| 帧率 | 60fps | 30fps |
| cpu耗时 | 10-12ms | 18-20ms |
| gpu耗时 | 6-8ms | 8-10ms |
| 温度 | 55℃ | 50℃ |

# 定位问题

首先找到程序的bound在哪里：CPU、GPU、带宽、（cache miss）。

还有考虑内存、耗电、发热量。

用render scale确定是CPU还是GPU问题。

找个顶点数少的地方确定shader整体耗时，一般温度低于55℃还算正常。

精简渲染流程。

数据的收集和效验。

IOS分析。

定位那些shader渲染面积大，那些耗时高，用arm或者gpa。

定位那些整体耗时gpa抓模拟器。

定位

先用一加手机全部场景跑一遍，记录优化前的温度（30fps）

用Profile看看是CPU还是GPU问题，抓下内存看看贴图和shader大小

首先看DrawCall、Vextex、Triangle有没有问题

开发项目用frame debug，看看用了什么效果

用GPA抓Win包：

看每个效果在PC的耗时，找出不合理的效果

粗略看看overdraw

用GPA抓模拟器：(不是很稳)

用renderdoc抓android：

看看android耗时

用arm抓android：

具体看overdraw，

渲染最多的shader，

渲染最耗时的shader。

用xcode抓：

分析shader的每一个指令。

最终解决，修改代码，显示隐藏功能来定位问题。

1. 隐藏场景 看看场景的整体消耗
2. 压缩图片一半的大小

小浣熊目前问题：

1. overdraw，用预深度，渲染排序
2. shader耗时、替换内置shader、树预深度、不要所有shader PBR
3. 有的shader开了AphaTest
4. 有的shader开了双面

renderDoc ：

每个Draw的耗时，

shader名字、源码、

使用的图片大小格式（一览很方便）、采样器状态、

使用的buff大小

# CPU

分析软件：Unity Profile

指标表：

|  |  |  |
| --- | --- | --- |
| 目标 | A | B |
| DrawCall | 150 | 300 |

# GPU

|  |  |  |
| --- | --- | --- |
| 目标 | A | B |
| Vertex | 30ｗ | 50w |
| Triangle | 20ｗ | 30w |
| OverDraw | 0.5 | 1 |
| Shader Cal |  |  |
| Texture |  |  |
| Buff |  |  |
| Bandwidth |  |  |

# 内存

分析软件：Unity Profile

指标表：

|  |  |
| --- | --- |
| Texture | 400MB |
| Shader | 50MB |
| Mesh | 50MB |