微信小游戏纹理压缩实践

Ocean*2019-11-04*2486浏览

一、概述

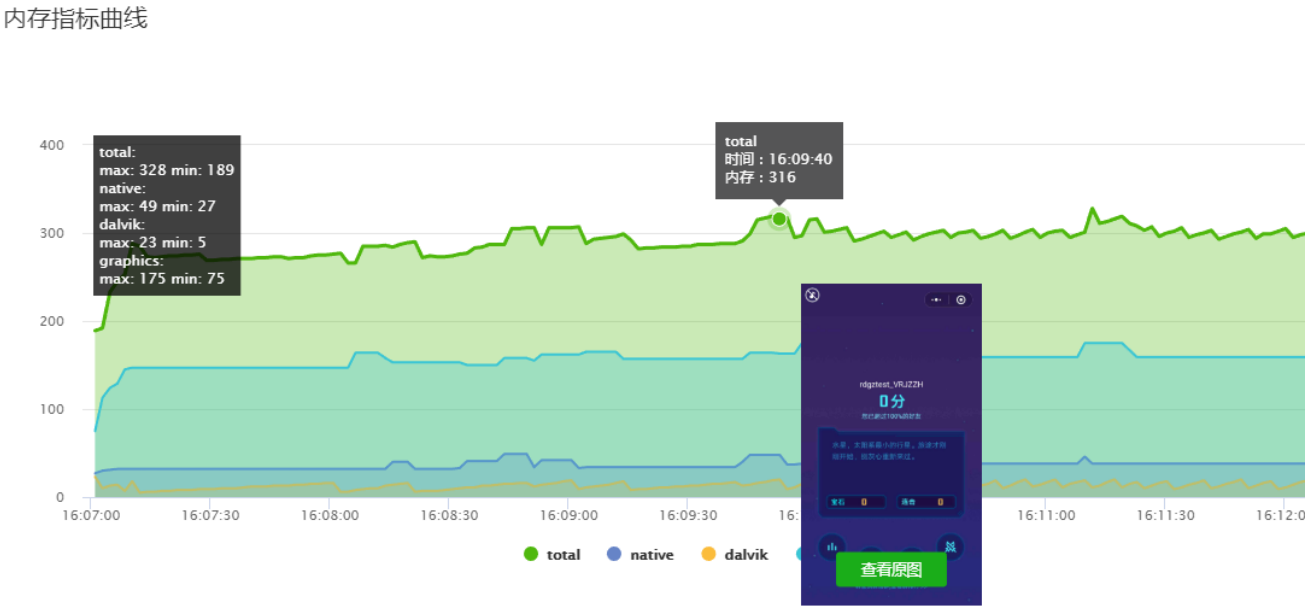
1.1 图形资源内存

小游戏运行过程中，内存过大是最为困扰开发者的一个问题。因为小游戏的内存使用过大时，在操作系统内存管理机制下，有可能会导致闪退或无法运行从而使得用户流失。据统计有**超过千款**小游戏内存使用不佳（超过优化建议值500MB）。

小游戏内存相关介绍可以参考https://developers.weixin.qq.com/minigame/dev/guide/best-practice/memory.html

现网运行时内存数据查看可使用**《小游戏数据助手》-性能-运行性能**

在内存的使用中， 图形渲染资源往往是游戏内存消耗大户，比如《星途》的内存使用大概如图所示（图片来自**《小游戏云测试服务》**，开发者请可通https://developers.weixin.qq.com/community/minigame/doc/000846255205f8c92d5912fca5e401参与内测）。



可以看到图形内存（graphic）占比是很大的， 休闲游戏犹如是，对于重度游戏我们发现图形渲染所使用的内存占比更大。那么优化小游戏的图形内存对于整体内存使用率的降低是非常有必要的。

1.2 纹理压缩

对于图形渲染资源的内存使用是否有比较好的办法去降低呢？ 纹理压缩是一种专为在计算机图形渲染系统中存储纹理而使用的图像压缩技术。与普通图像压缩算法的不同之处在于，纹理压缩算法为纹素的随机存取做了优化。

二、 压缩算法与性能

2.1 压缩算法

Android设备中一般使用ETC1压缩，一种有损的图像压缩方式。 ETC1是OpenGL2.0支持的标准，压缩之后每个像素占4bit，压缩之后的格式为KTX或者PKM，前者支持存储多纹理，后者只支持单纹理。 ETC1的缺点是不支持Alpha通道，不支持有透明度的图片压缩。 ETC2解决了Alpha通道，但是它是OpenGL3.0标准，考虑到2.0设备的市场占有率，一般使用ETC1。

iOS设备中采用的图像格式一般是PVR，也是一种有损的图像压缩方式。PVR压缩分为两种，PVRTC2和PVRTC4。 除了压缩内存，PVR可以直接被显卡读取，载入速度更快；缺点是PVR需要PowerVR芯片支持，目前iOS设备都能完美支持，Android支持尚少；此外，PVRTC4只支持方形贴图，非方形会被处理成方形，且长宽必须为2的幂。

2.2 文件格式

KTX和PKM都是纹理存储的文件格式，前者支持存储多纹理，后者只支持单纹理，大部分移动设备的 GPU 均支持这两种格式。可以有效降低设备的显存占用，提高运行效率和稳定性。

2.3 压缩纹理的性能优势

测试的手机是一加3T， 实验中使用了30张都是584KB的JPG图片进行渲染。

图1：实验前初始内存

图2：经过纹理压缩的图片渲染，GPU内存消耗总量。

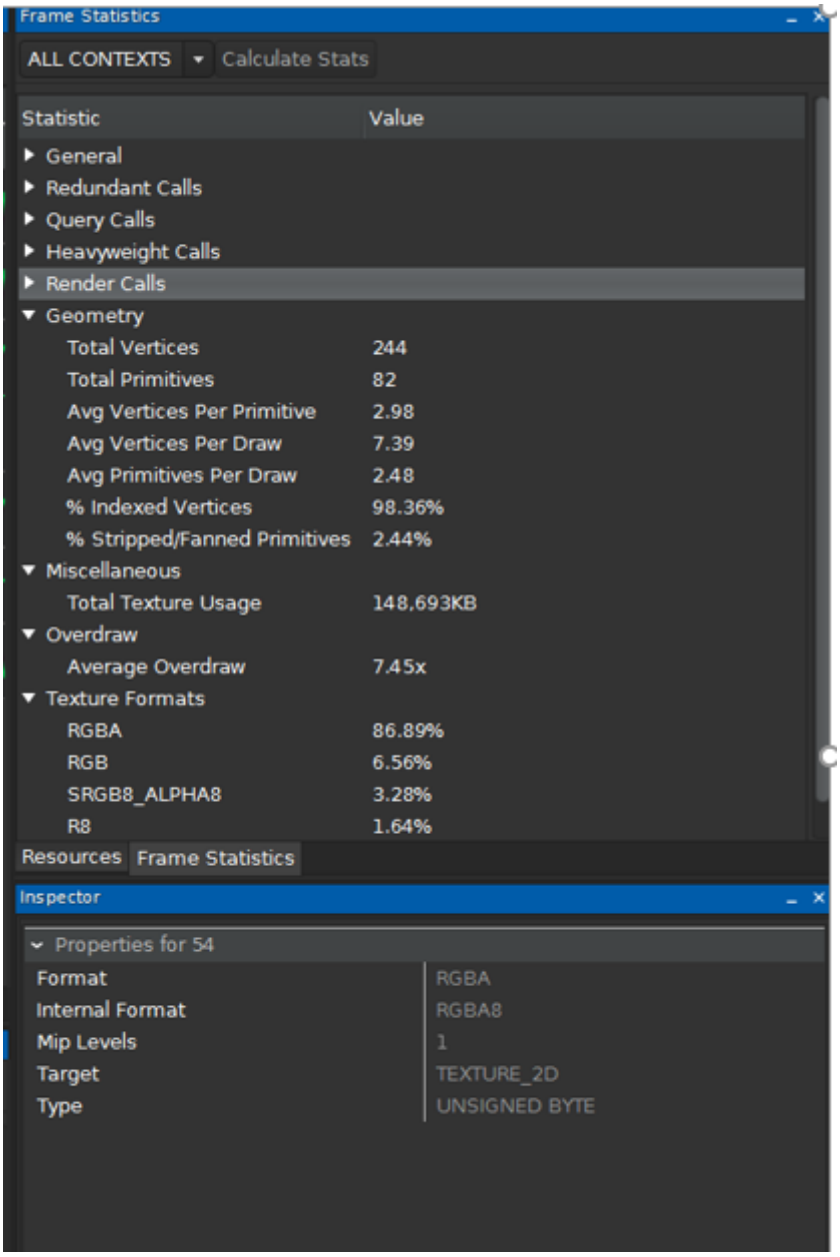
图3：JPG格式的图片渲染，GPU内存消耗总量。

内存比较：使用纹理压缩时内存从78(176MB-98MB)降低到了20MB(118MB-98MB), 带来约为**70%**的图形资源内存消耗。

下面就是使用了Snapdragon Profiler工具进行的一些测试数据 使用了纹理压缩的测试数据：

Total Texture Usage 只有46M

未使用纹理压缩的测试数据：

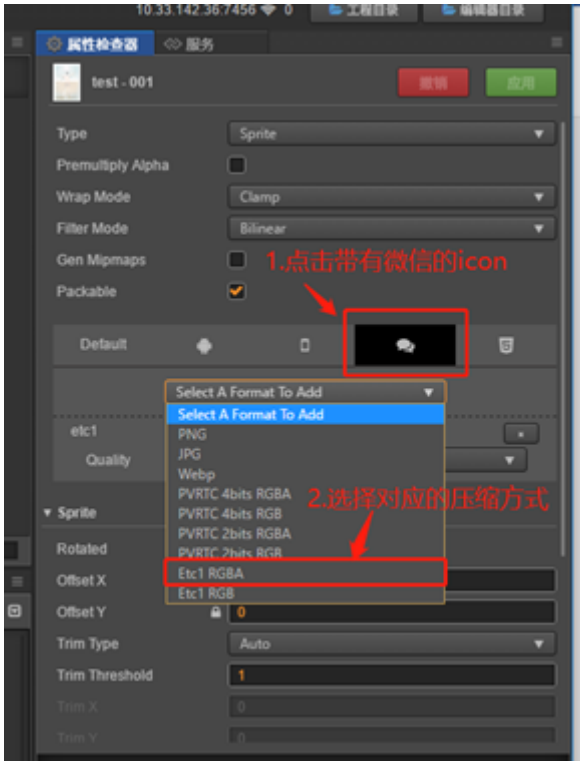


Total Texture Usage为148M 从这里也可以佐证压缩纹理时图形内存占用能得到极大的降低。

三、Cocos Creator微信小游戏纹理压缩实践

引擎文档： https://docs.cocos.com/creator/manual/zh/asset-workflow/compress-texture.html

在cocos creater里面进行简单的配置，就可以在打包小游戏的过程中把图片进行纹理压缩。



这里不仅能选择压缩算法，还能选择压缩算法质量等参数，非常方便。 不过目前引擎自带设置还不能支持批量操作（其实有方法解决不能批量操作的问题，这就要在cocos社区上找）。

四、Egret 微信小游戏纹理压缩实践

引擎文档：http://developer.egret.com/cn/github/egret-docs/Engine2D/bitmapTexture/ktx/index.html

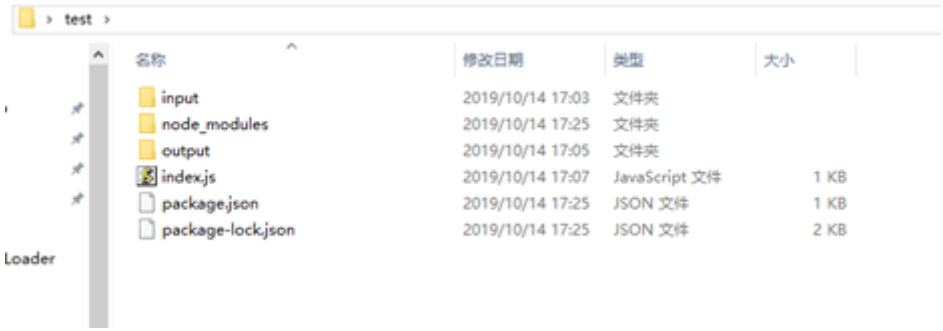
白鹭是没有提供像Cocos Creater那样的配置的，是需要我们手动转。 白鹭官方提供了4种工具让我们进行制作压缩纹理:

> 我们制作了一个 ktx 转换工具：下载地址和说明文档
>
> 也可以使用其他的转换工具:
>
> - PowerVRSDK
> - Mali_Texture_Compression_Tool
> - texture-compressor

我们选择了texture-compressor 这个工具进行转换，该工具是可以批量操作的。

npm install texture-compressor 进行安装

我这里的目录结构看下图



input文件夹是存放纹理压缩前的图片，而output是存放纹理压缩后生成的.KTX后缀的文件。 index.js是执行文件，代码如下。

上面这张图的配置是转换ETC的用于安卓系统。

```
const fs = require('fs');
const { pack } = require('texture-compressor');

fs.readdir('./input',function(err,files){
    if(err) return console.warn(err)

    for(let i = 0,len = files.length; i<len; i++){
        pack({
            type: 'etc',
            input: `input/${files[i]}`,
            output: `output/${files[i].replace('.jpg','.ktx')}`,
            compression: 'ETC2_RGB',
            quality: 'etcfast',
            verbose: true
        }).then(() => console.log('done!'));
    }
})
```

下面这张图是转换PVRTC的用于IOS系统。

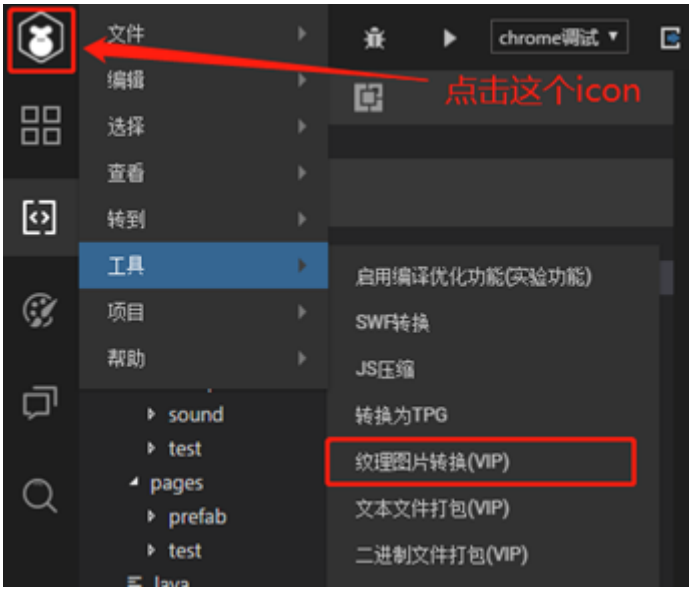
```
fs.readdir('./input',function(err,files){
    if(err) return console.warn(err)

    for(let i = 0,len = files.length; i<len; i++){
        pack({
            type: 'pvrtc',
            input: `input/${files[i]}`,
            output: `output/${files[i].replace('.jpg','.ktx')}`,
            compression: 'PVRTC1_2',
            quality: 'pvrtcnormal',
            verbose: true
        }).then(() => console.log('done!'));
    }
})
```

运行中如何加载使用压缩纹理则可以通过上述引擎文档参考demo。

五、Laya微信小游戏纹理压缩实践

引擎文档：https://ldc2.layabox.com/doc/?nav=zh-ts-4-14-7

目前LayaAir2.x版本提供的纹理压缩功能是只有VIP会员账号才能使用，可以支持批量转换的操作如下。 ‘



选择需要纹理压缩的图片所在文件夹和转换后的保存路径，接着选择使用平台按确定就可以了。使用方法也很简单：

```
//以下代码片段仅作参照
.......
    if (Browser.onAndroid) {
        this.extension = "ktx";
    } else if (Browser.onIOS) {
        this.extension = "pvr";
    }

    var sp:Sprite = new Sprite();
    sp.loadImage("res/1."+this.extension);
.......
//以上代码片段仅作参照
```

代码片段摘自Layabox官方文档提供，代码中根据所在系统判断使用哪个的纹理压缩文件就可以了。

六、压缩纹理需要注意的一些问题

从上述实践过程中，我们发现：

1. 压缩纹理能够很好的节省图形资源内存，从而让我们小游戏的内存使用得到优化。

2. 引擎制作工具也提供了支持， 让我们在制作资源制作打包时能够快速使用压缩纹理。

然而，是否所有情况都适合使用压缩纹理呢？ 是否还存在使用上需要注意的问题

1. 资源大小

纹理压缩的资源体积会比常规压缩算法偏大, 硬件对不同压缩算法的尺寸有要求，这也是最为影响开发者制作流程的一个因素。典型地，在Android中的ETC1纹理压缩算法需要长宽尺寸是2的N次幂， 而iOS中的PVRTC则除了2的N次幂外还要求是正方形。例如原图尺寸为 228x380 的图片，转换成 PVRTC 的 ktx后，尺寸为 512x512。 我们来举个制作不太符合此规范的资源的例子，原始图像是：879 x 1242。那么：

格式	纹理压缩	像素	文件大小
JPG	无	879 x 1242	584KB
PVRTC	有	2048 x 2048	2049KB
ETC1	有	1024 x 2048	1025KB

可以看到，ETC1, PVRTC压缩后的文件体积比以前大很多（在考虑mipmap的情况下，还会增加30%左右）因此，在使用纹理压缩时，最为挑战开发者的一个问题就是如何让美术资源更符合压缩算法的标准尺寸。美术资源最好是2的N次幂，在iOS PVRTC下则需要长宽相等。

2. alpha通道透明度

Android下ETC1是不支持alpha通道的，因此还需要额外体积去多存储这部分alpha通道数据，而PVRTC则默认支持。

3. 兼容性

硬件相关，要考虑兼容性。 我们这里总结下常见的几种纹理压缩特性：

	Alpha	压缩率	OenGL要求	资源尺寸	设备支持
ETC1	N	6:1	OpenGL ES 2.0	2的幂次方	装有OpenGL ES 2.0及以上版本的所有安卓设备可以支持
ETC2	Y	6:1	OpenGL ES 3.0	2的幂次方	支持大部分高端Android机
PVRTC2BPI	Y	16:1	OpenGL ES 2.0	2的幂次方， 长宽相等	Android机（GPU：PowerVR系列），iPhone全系列机型
PVRTC4BPI	Y	8:1	OpenGL ES 2.0	2的幂次方， 长宽相等	Android机（GPU：PowerVR系列），iPhone全系列机型
ASTC	Y	可选压缩率	OpenGL ES 2.0	不要求图片长宽相等且为2的幂次方	支持部分高端Android机型，iPhone6及以上机型

微信小游戏开发者在研发过程中有什么技术疑问或建议，欢迎与我们交流！