

渲染优化（一）

真机：iPhone X 处理器 A11

XCode 版本13.4.1

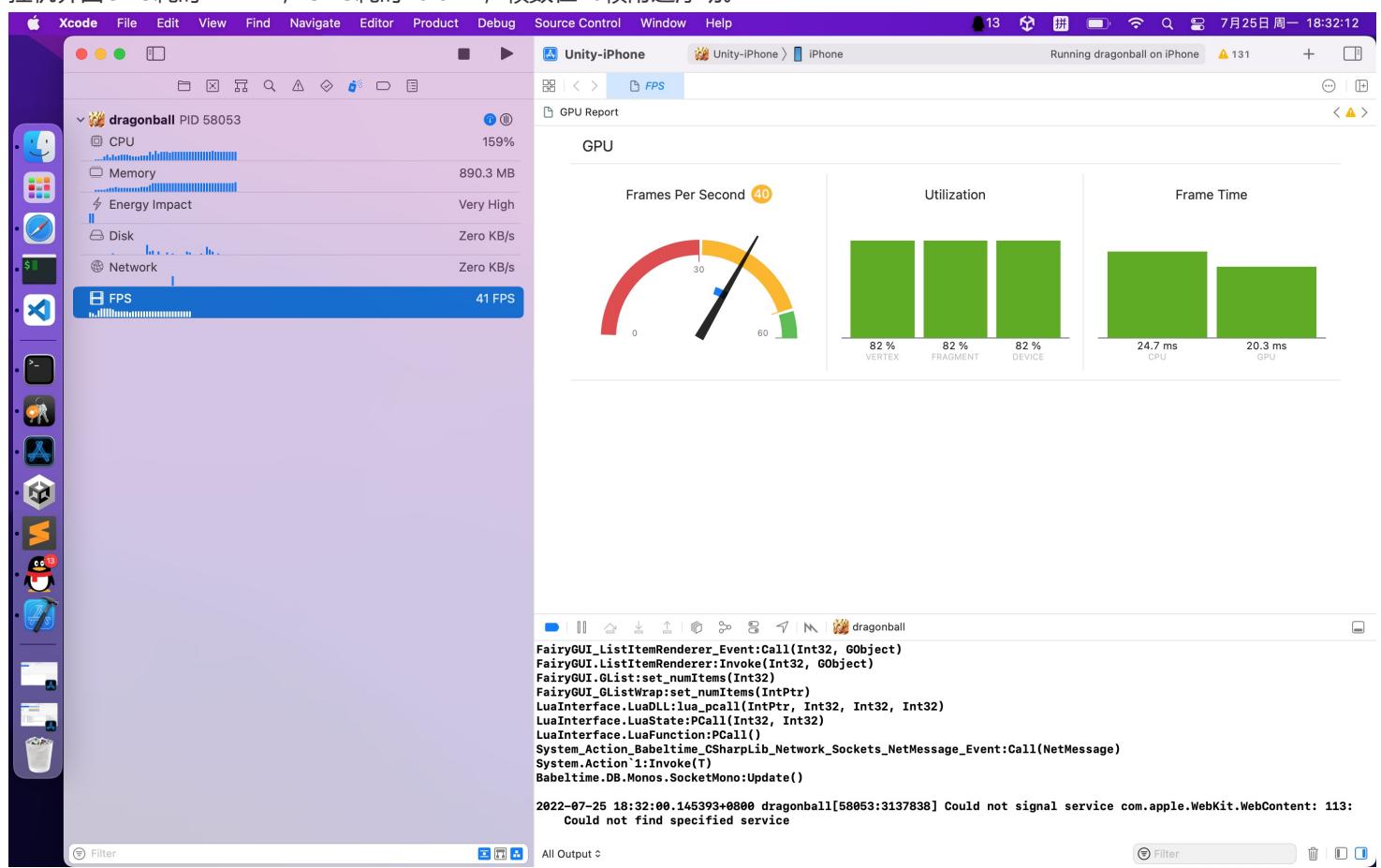
• 结论

- 需要进项目确认的问题是，Bloom进行了几次，后处理都做了什么。4Textures这个shader做了什么。SMAA是否为最适合的抗锯齿方案。FAE插件是否需要优化
- 可优化的点是，草的渲染采用GPU实例化，树的渲染将噪声函数改为噪声贴图，光照计算尽可能使用half。

挂机场景（悟空家）

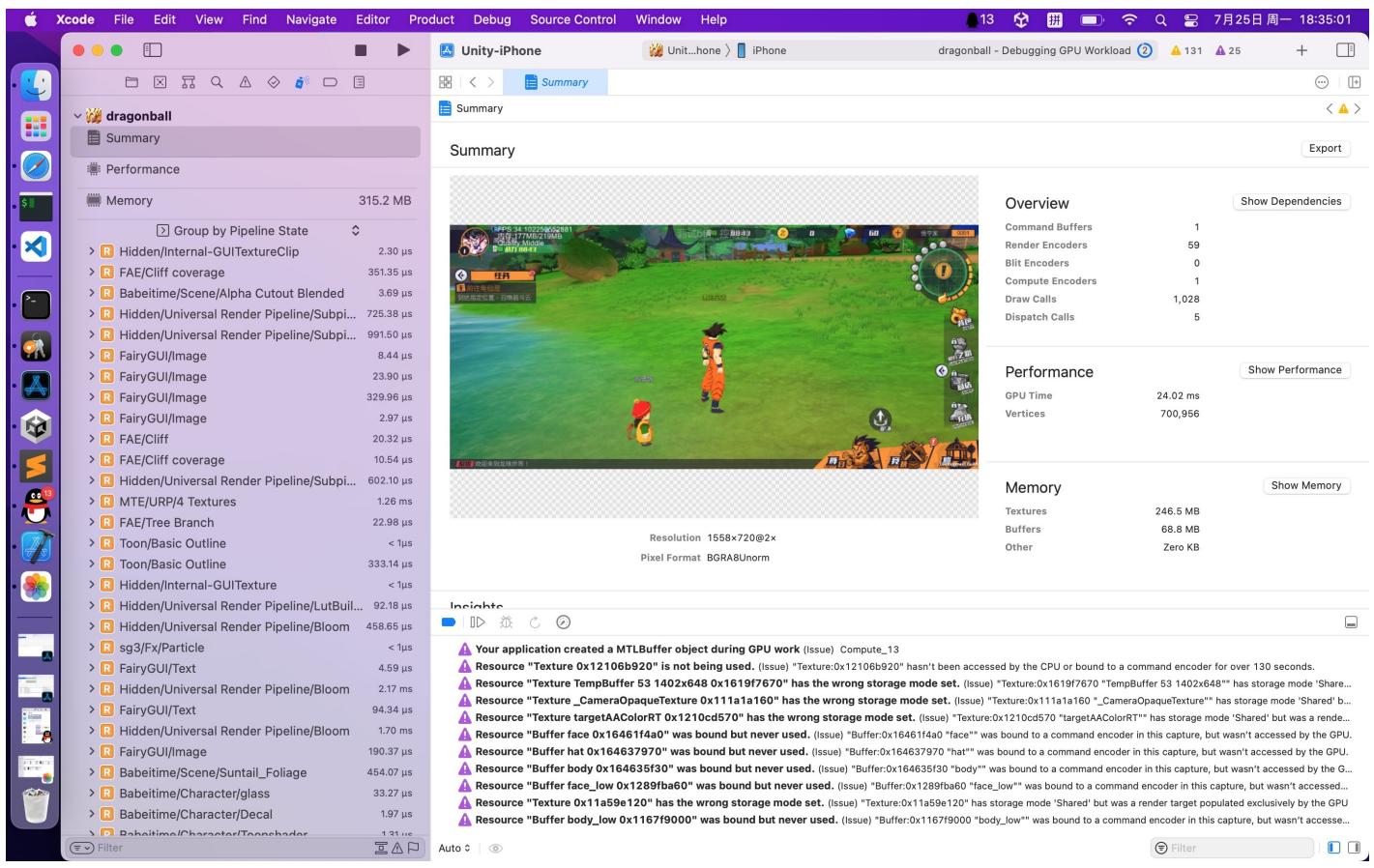
性能总览

挂机界面CPU耗时24.7ms，GPU耗时20.3ms，帧数在40帧附近浮动。



抓帧数据

1. 数据总览：

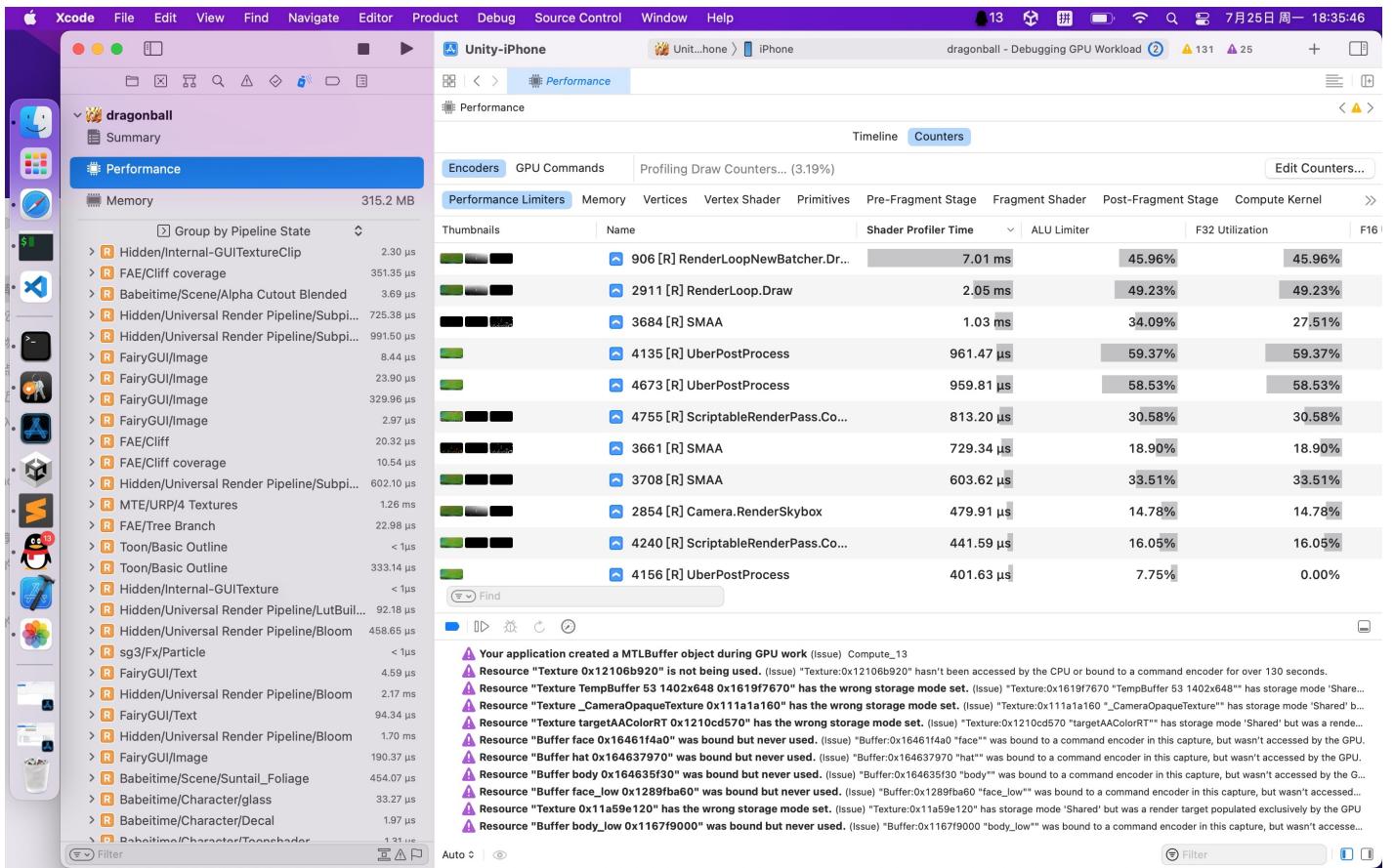


名称	数据
DC	1028 个
顶点数	700956 个
占用显存	315.3 MB
纹理占用	246.5 MB
Buffers	68.8 MB

从这个数据来看，最大的问题为DC数量太多有1028个

1. 渲染事件耗时概览(由大到小排列)

如图，渲染事件接近1ms的有五个



渲染事件	耗时	所属渲染队列	该队列总耗时	渲染相机	渲染相机耗时
RenderLoopNewBatcher	7.01ms	DrawOpaque	7.01ms	MapCamera	16.39ms
RenderLoop	2.05ms	DrawTransparent	2.99	MapCamera	16.39ms
SMAA	1.03ms	PostProcessing	5.81ms	MapCamera	16.39ms
UberPostProcess	0.96ms	PostProcessing	5.81	MapCamera	16.39ms
UberPostProcess	0.96ms	PostProcessing	4.68ms	UIPostProcessingCamera	5.28ms

后处理有两次，一次在MapCamera,一次在UIPostProcessingCamera,主渲染在MapCamera很奇怪（名字起错了？）

1. 场景渲染相机排序(由大到小排列)

相机名称	渲染耗时
MapCamera	16.39ms
UIPostProcessingCamera	5.28ms
UICamera	1.14ms
Base_Camera	0.74ms

2. 场景shader耗时排序（所有使用该shader的DC累加，由高到底，超过1ms的）

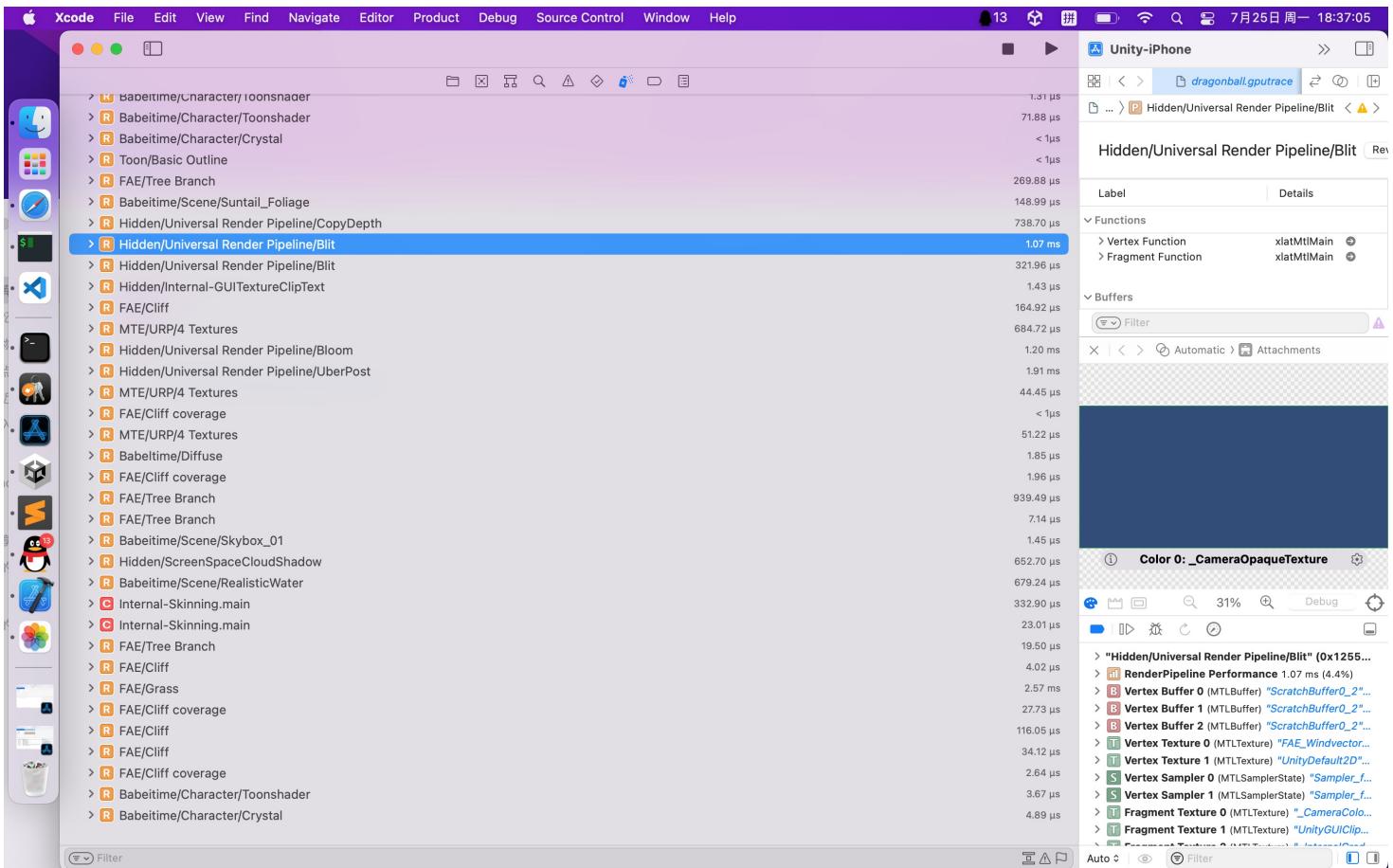
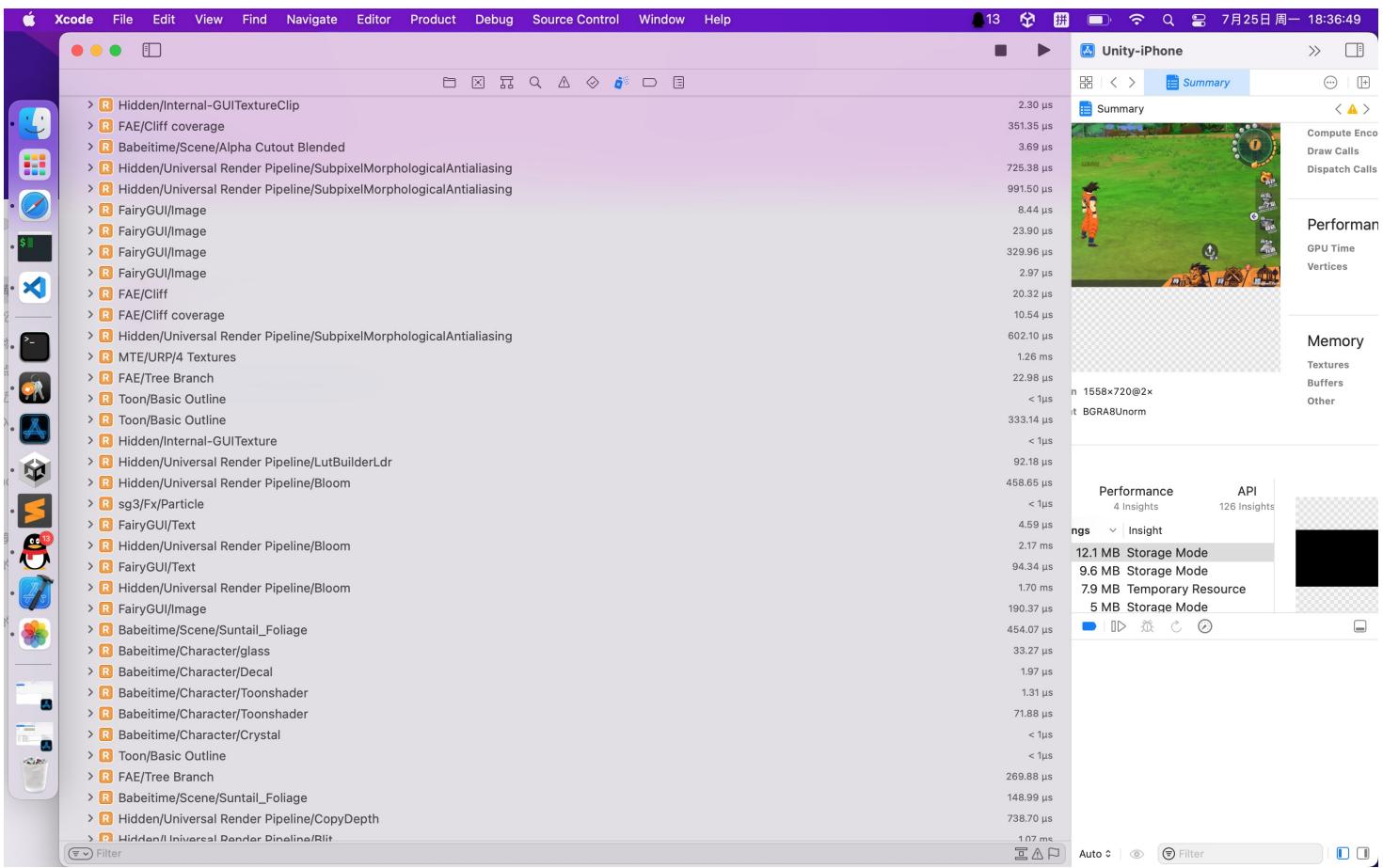
shader名称	引用数量 (DC)	总耗时	平均单个DC耗时	Vs耗时比率	Ps耗时比率
FAE/Grass	647	2.71ms	0.004ms	35%	65%

shader名称	引用数量 (DC)	总耗时	平均单个DC耗时	Vs耗时比率	Ps耗时比率
Bloom1	12	1.87ms	0.15ms	32%	68%
UberPost	2	1.49ms	0.745ms	0.1%	99%
Bloom2	12	1.46ms	0.12ms	44%	56%
Suntail_Floage	8	1.29ms	0.16ms	1%	99%
Bloom3	12	1.10ms	0.09ms	27%	73%
SubPoxlMorphologicalAntialiasing	1	1.05ms	1.05ms	0.1%	99.9%
Blit	4	1.02ms	0.255ms	0.1%	99.9%
Suntail_Floage(LOD)	91	0.85ms	0.0093ms	25%	75%
FAE/Cliff	35	0.80ms	0.022ms	10%	90%
CopyDepth	2	0.74ms	0.37ms	50%	50%
SMAA	1	0.73ms	0.73ms	0.1%	99.9%
4Textures	1	0.70ms	0.7ms	0.01%	99.9%

场景中一共62个shader，这里列出的13个shader耗时15.81ms，也就是说**20%**的 shader消耗了 **70%** 时间。
这些shader是该场景中超过700ms的，引用数量太多说明需要进行合

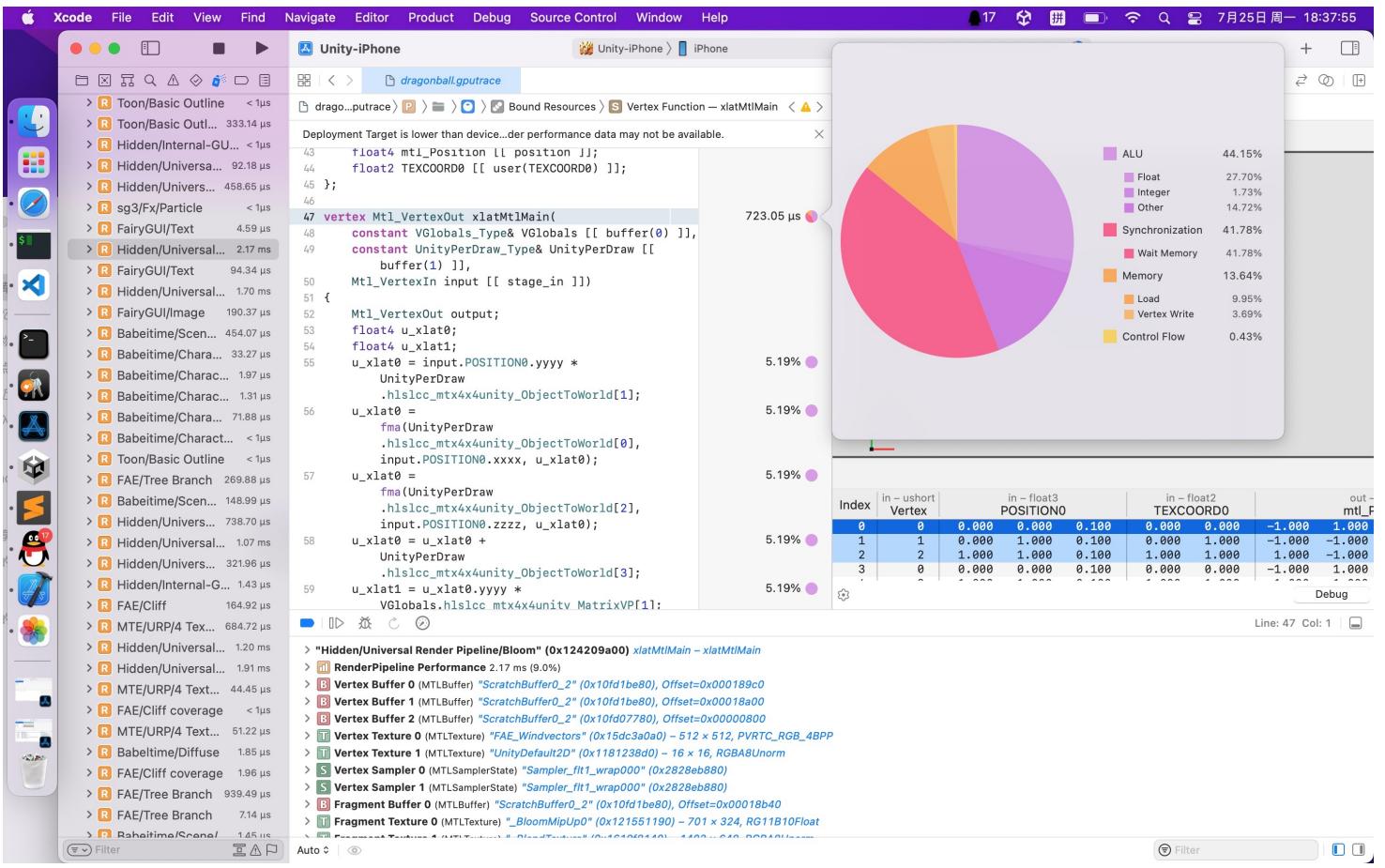
批，Grass shader引用数量有647个DC，草的渲染可以使用GPU实例化，减少DC，SRPBatcher只减少setpass，不减少DC。
从这里依然可以看出多次后处理的问题。

单个DC耗时可以评估一个shader的效率，如果单个耗时太高，则需要针对shader优化。



)

Bloom多次计算且VS阶段耗时



从图中可以看出，ALU计算耗时44%，同步等待耗时41%。
且有float向half的隐式转换