# Q1论文试图解决什么问题？

移动设备上显示分辨率的不断提高，增加了对GPU渲染细节的高需求。糟糕的硬件支持和细粒度的呈现细节的挑战往往会让用户不满意，特别是在调用高帧率的场景时，例如，游戏。

# Q2这是否是一个新的问题？

移动应用程序中的GPU功率优化并不是一个新问题。

# Q3这篇文章要验证一个什么科学假设？

1. 在设备之间设计一个移动协作的GPU渲染机制可以有效帮助优化高帧率场景时的GPU优化。
2. 剪裁掉那些用户视野之外，渲染了却未显示的纹理渲染结果的部分将进一步优化移动图形的渲染。

# Q4有哪些相关研究？如何归类？谁是这一课题在领域内值得关注的研究员？

1. **GPU功率优化。**目前，在桌面/服务器[7]、[24]、[29]、[30]的建模和进一步优化GPU功耗方面有很多工作，重点是动态频率缩放。[31]和[32]分别通过调整显示分辨率和刷新率来研究桌面平台的游戏体验。研究结果表明，较低的帧率可能会导致系统功率的降低。这些技术通常都适用于智能手机。然而，GPU的频率扩展是在供应商提供的驱动程序中控制的，即使通过调用操作系统级界面，也很难被移动应用程序配置。然而，在ButterFly中，我们提出了在移动设备上不修改ROM的情况下实现高质量渲染的特定技术。
2. **GPU和GPGPU卸载。**

2.1通过在设备和更强大的服务器基础设施之间划分程序来克服移动设备的资源限制是一项挑战。Chroma [33]为程序员提供了指定程序分区和采用这些分区的条件，而MAUI [7]和CloneCloud [4]利用管理语言运行时的特性，根据运行时对分区的能量和性能效益的预测，自动对程序进行分区。然而，所有这些系统都是为通用的CPU工作负载而设计的，并且不适用于严重依赖于桌面/移动GPU的快速行动游戏等应用程序。为了解决这个问题，川外[8]采用了服务器端支持。然而，它却带来了大规模部署的巨大成本。

2.2利用大众的移动GPU（手机）来实现高质量的渲染。随着通用GPU（GPGPU）计算在移动平台上的普及，GPGPU很明显，低计算能力和有限的电池容量显然是用户面临的瓶颈。为此，研究人员提出了不同的解决方案，例如[34]，[35]，通过将重计算从低功耗设备迁移到更强大的机器上，来减轻低功耗设备的负担。同样，ButterFly也为高质量的移动GPGPU计算问题提供了一个显著的解决方案，它可以应用于大多数COTS安卓设备。

1. **重叠的图形问题**。GraalBench [36]是一个用于低端移动设备的3D基准测试套件，Ma等人[37]已经描述了手机游戏的功耗。虽然他们的方法很鼓舞人心，但没有人研究过OpenGL ES管道效率低下。尽量减少对移动GPU资源的浪费一直是一个热门话题。LPD [38]仅通过合成最近更改的不必要UI区域，从而减少了内存和显示界面流量。安卓系统配备了各种工具来解决GPU透支问题[39]。ButterFly的目标是与它们正交的，因为它通过裁剪GPU工作负载来避免不必要的纹理。

# Q5论文中提到的解决方案之关键是什么？

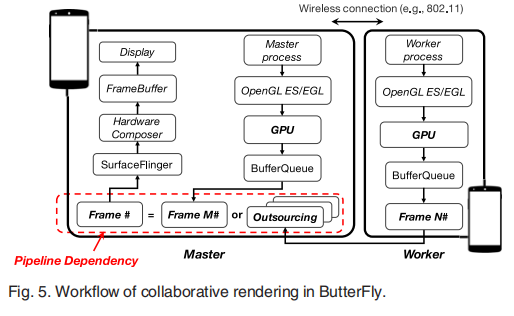
1. GPU协作渲染。并不是服务器端与移动端的协作，而是利用用户的多个移动端进行渲染协作。（个人主观性总结）
2. 从OpenGL ES管道出发，剪裁掉那些用户视野之外，渲染了却未显示的纹理渲染结果的部分。

# Q6论文中的实验是如何设计的？

**Butterfly**

**1.实验原理**

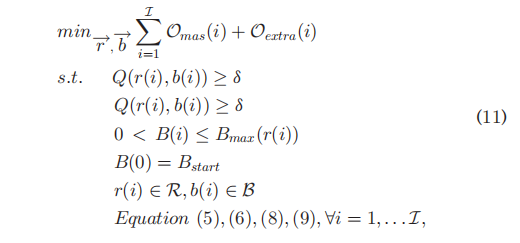
**1.1总体架构**



蝴蝶的概念工作流程。具体来说，它包括三个相关的阶段，即主处理、工作处理和管道依赖关系。

**1.2结合能量与延迟进行优化**

优化问题



* 优化的目标是在整个任务（完全I框架）中最小化总能量开销
* 限制1是要使得采取渲染决策r(i)与分辨率像素b(i)的每帧质量Q都要大于一个阈值以此保证质量
* 限制2是要使得buffer size要符合实际情况，即在[0，规定的分辨率像素r(i)容量决定的buffer size]中

**2.实现与评估**

**2.1涉及的软硬件**

|  |  |  |  |
| --- | --- | --- | --- |
| 实现 | OpenGL ES 3.0 | 安卓4.2+ | 改进的开源引擎  Rajawali  version 1.0.325 |
| 评估 | SamsungGalaxy S5 LTE-A, S4, Google Nexus 5 and Nexus 6 smartphones. | | |

**2.2细粒度能量测量法：**

将一个硬件功率计[17]附加到智能手机的电池上。

**2.3评估**

我们首先将butterfly的潜在好处分为两类。用户研究的第一类定性好处，即视觉质量和帧率评分，其结果需要与即时移动的移动用户相结合。第二类定量和客观分析是与其它四个系统对比。

**2.3.1用户研究：**

1）问卷的问题

“我认为它渲染质量很高（看起来不错）。”

“我认为它在渲染过程中很少掉帧（运行平稳）。”

“我周围有多少个设备可以作为渲染的worker？”

...等

1. 评价维度

“差”、“中等”、“好”、“优秀”、“相同”

1. 测试对象

40名参与者（15名女性和25名男性）在Rajawali引擎运行几种游戏渲染：《最终幻想》、《外星人射击》和《使命召唤》。每个参与者被要求使用butterfly和原始的方法至少20分钟。参与者的学科背景是多样的。

**2.3.2定量分析**

**宏基准（我理解的是与其它系统横向对比）**

1. 对比对象

原始方法；使用服务器端渲染而不考虑延迟问题；使用服务器端渲染并考虑延迟；butterfly，即使用butterfly最多三个worker（由上述研究确定）

1. 对比的指标

①FPS性能：四种分辨率四个对比对象进行对比

②电源开销：四种分辨率下butterfly与服务器端进行对比

**微基准（我理解的是butterfly内部参数对指标的影响）**

①测量了使用ButterFly渲染最终幻想游戏的平均每帧渲染时间（PFR）GPU渲染和总时间（1000帧）：不同数量的worker下（0-4）

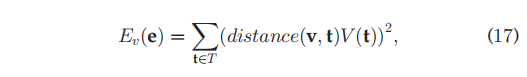
②能源开销：在不同worker数量下、不同渲染质量（分辨率）

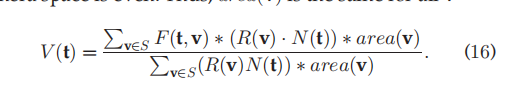
③802.11环境下的延迟响应/单设备的有效吞吐率：不同帧率、不同分辨率

**可见纹理简化VTS**

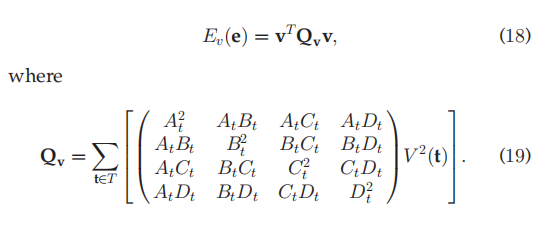
**1.原理**

可见性驱动的误差度量:





即



**VTS节能方法的评估**

**2.实现与评估**

**2.1涉及的软硬件**

|  |  |  |  |
| --- | --- | --- | --- |
| 三星Galaxy S5 LTE-A智能手机 | 2560x1440的默认显示分辨率 | 四核高通骁龙805 SoC | 600 MHz的核心速度GPU |
| Monson功率监测工具:测量系统的总能量 | | | |
| 使用十大流行的图形模型来评估VTS。这10个模型包括不同的复杂图形架构 | | | |

**2.2评估**

**2.2.1单独评估VTS的节能效果**

1. 对比对象

①基线，即默认的没有任何能量优化的三维纹理渲染；

②高节能强度的三维纹理渲染来自特定的视野视图，60%的纹理被裁剪；

③3D纹理渲染来自特定的视野视图，并有40%的纹理被裁剪；

④低节能强度，3D纹理渲染来自特定的视野视图，只有20%的纹理被裁剪。

1. 对比的指标

①标准化功率（PPF）：对以上四个对象进行对比

**2.2.2评估VTS在协同渲染中的效果**

1）评估的指标

①帧率FPS：协同渲染中是否使用VTS；worker个数；不同分辨率。

**2.2.3评估VTS的开销**

①系统延迟：使用“龙”模型（十个模型中最复杂的）

②内存开销：十个模型

③能源开销

# Q7用于定量评估的数据集是什么？代码有没有开源？

**1数据集**

**1.1Butterfly用于渲染的游戏：**

《最终幻想》、《外星人射击》和《使命召唤》

**1.2VTS用于渲染的模型获取途径：**

①Trimble 3D Warehouse：https://3dwarehouse.sketchup.com/

②Yobi3D

③Sketchfab：http://www.cgchannel.com/tag/sketchfab/

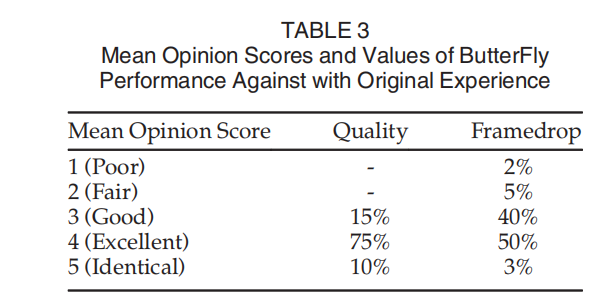
**2代码**

[Rajawali](https://github.com/Rajawali/Rajawali)：https://github.com/Rajawali/Rajawali

# Q8论文中的实验及结果有没有很好地支持需要验证的科学假设？

1. **Butterfly**

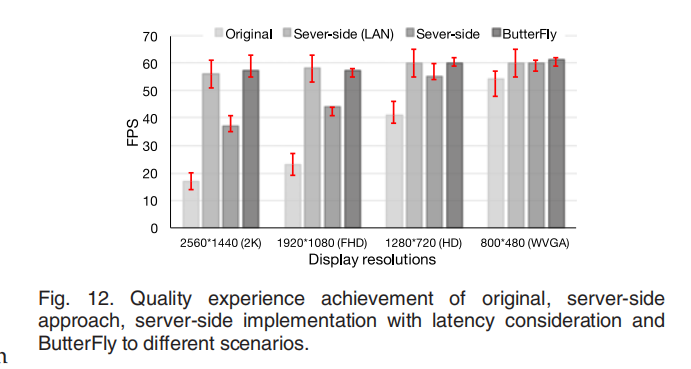
**1.1用户研究：**



**2.3.2定量分析**

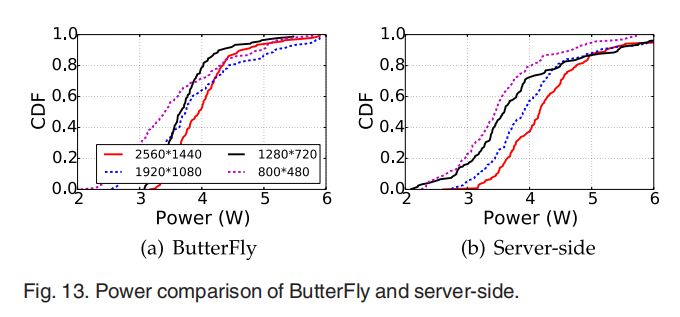
**宏基准**

①FPS性能：



Butterfly方法能够与不考虑网络状态的服务器负载方法相提并论甚至略胜一筹，说明效果是非常好的，毕竟成本更小。

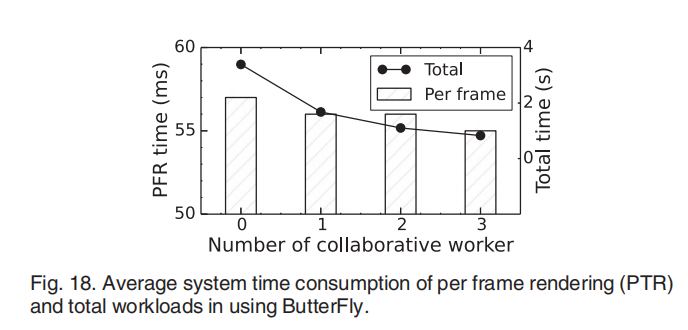
②电源开销：四种分辨率下butterfly与服务器端进行对比



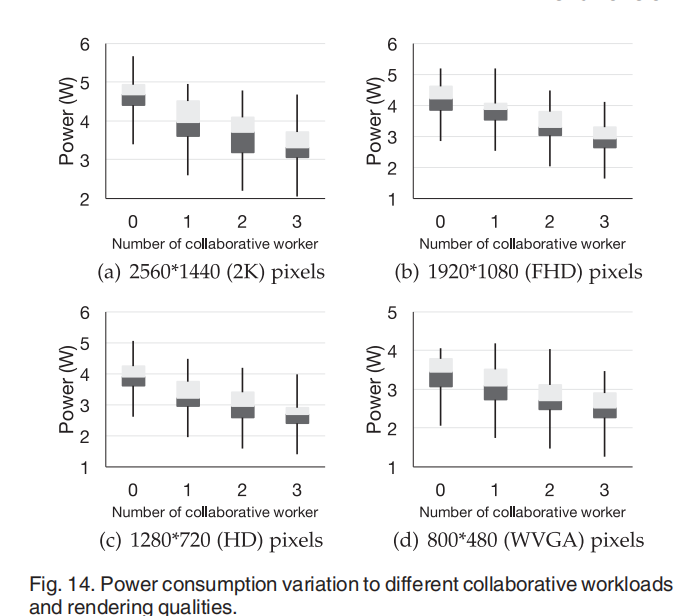
注：CDF是功率的累计概率分布，看这个图主要看CDF=0.5处的数据，它反应了平均功率情况。

**微基准（我理解的是butterfly内部参数对指标的影响）**

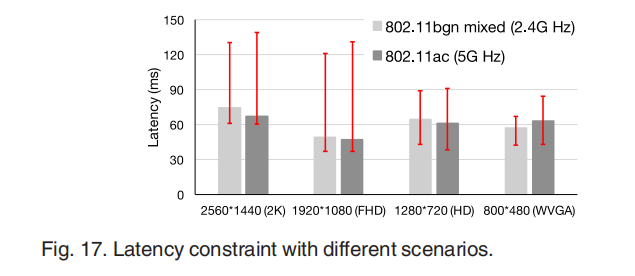
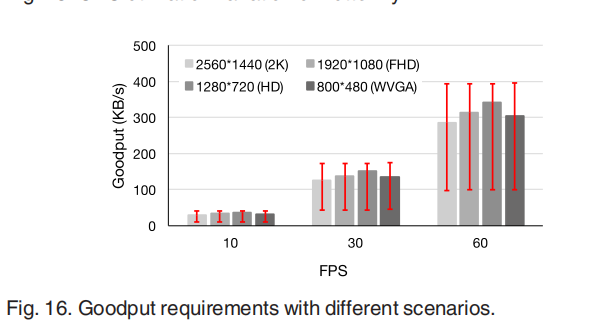
①测量了使用ButterFly渲染最终幻想游戏的平均每帧渲染时间（PFR）GPU渲染和总时间（1000帧）：不同数量的worker下（0-4）



②能源开销：在不同worker数量下、不同渲染质量（分辨率）



③802.11环境下的延迟响应/单设备的有效吞吐率：不同帧率、不同分辨率

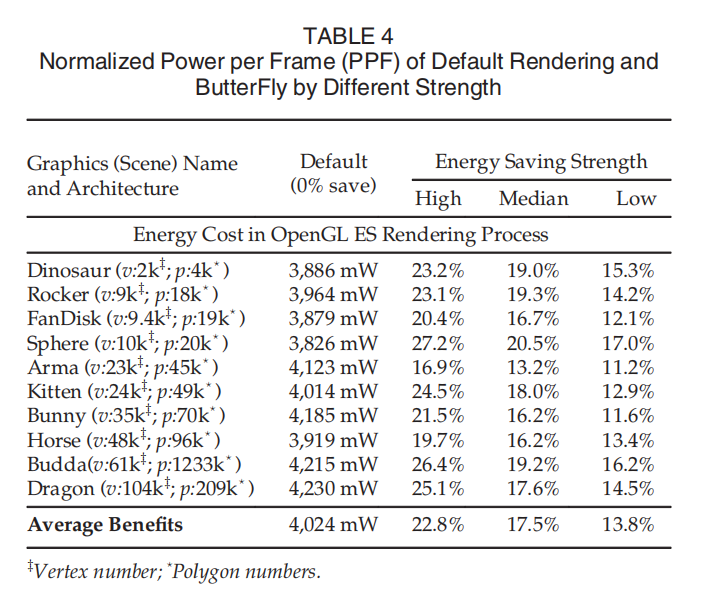


只是单纯对butterfly在各种分辨率下的性能测试，没有进行对比。

1. **VTS**

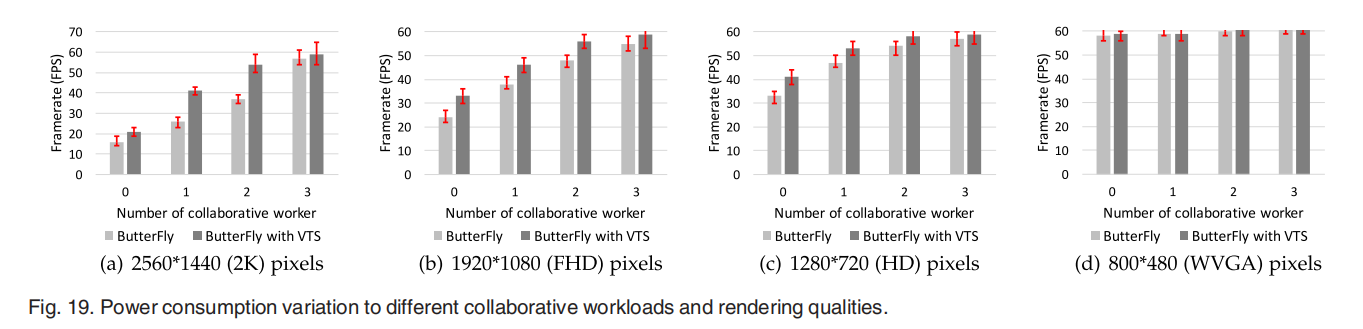
**2.2.1单独评估VTS的节能效果**

①标准化功率（PPF）：对以上四个对象进行对比



**2.2.2评估VTS在协同渲染中的效果**

①帧率FPS：协同渲染中是否使用VTS；worker个数；不同分辨率。

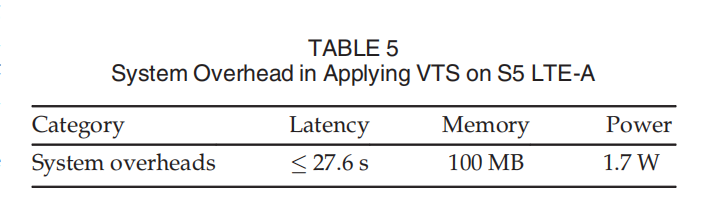


**2.2.3评估VTS的开销**

①系统延迟：使用“龙”模型（十个模型中最复杂的）

②内存开销：十个模型

③能源开销



# Q9这篇论文到底有什么贡献？

是第一篇为移动设备设计和实现协同渲染的论文

* 我们设计了一个系统级的协作渲染机制，并在其中提出了蝴蝶的设计和实现技术，它与OpenGL ES/EGL标准兼容。
* 我们联合了能量和RTT延迟考虑的全局优化的蝴蝶。
* 我们已经在现在的安卓智能手机上实现了蝴蝶原型，但硬件限制很少。
* 我们进行了全面的实验和用户研究，以确认我们的系统在优化用户体验和电源开销方面的有效性

# Q10下一步呢？有什么工作可以继续深入？

* 我们计划扩展我们目前的实现，以支持更灵活的移动图形应用程序，并将我们的实现移植到更多的平台上，包括Android、iOS和Windows，以进行更多的测试。
* VTS剪裁不必要纹理时时间延迟过大，可以通过预先剪辑潜在的纹理，并按需加载它们来解决这个问题。