Mobile GPU Power Consumption Reduction via Dynamic Resolution and Frame Rate Scaling

Kent W. Nixon[†‡], Xiang Chen[†‡], Hucheng Zhou[‡], Yunxin Liu[‡], Yiran Chen[†]

Microsoft Research[‡]
Beijing, China 100080

ECE Department, University of Pittsburgh[†]
Pittsburgh, PA, USA 15261

Abstract

The emerging industry trend of ever-increasing display density on mobile devices has dramatically increased workload placed on a mobile GPU's. Because mobile GPU power consumption increases almost linearly with workload, increasing the display density directly decreases battery life of a device. While this tradeoff is acceptable if user experience is improved, display densities beyond that which the human eye can perceive would result in decreased device battery life for no perceptible gain. Further, the workload imposed by such high density displays may invalidate the previous requirement that the interface always run at high frame rates.

In this paper, we show that the display on some modern devices already exceeds human perceptive capabilities, a feature which can be exploited at runtime in order to reduce GPU power consumption. Our proposed method includes both resolution and refresh rate scaling, which are shown to reduce mobile GPU power consumption by up to 33% and 38%, respectively. We conclude with a discussion of how such a system may be implemented on existing devices.

1. Introduction

The electronics industry has recently begun trending towards extremely high density displays. With the release of the iPhone 4 in 2010, Apple began to popularize "Retina displays," or displays on which individual pixels cannot be optically isolated by the human eye [1]. Most manufacturers are now participating in similar ventures, with many televisions, laptops, and recently phones, being released with heavy marketing emphasis being placed on the devices having QHD (2560x1400 pixels), WQXGA+ (3200x1800 pixels), or UHD (3840x2160 pixels) resolution.

While high density displays allow for sharper images and more content to be legibly displayed, they also come with larger system resource requirements. Modern systems must maintain multiple frame buffers for the display; Android utilizes as many as 3, requiring approximately 100MB of memory for a UHD display using 32 bit color - 4x the space required for FHD (1920x1080 pixels). This increased frame buffer size also requires higher memory bandwidth, which is a limited resource on mobile devices where both CPU and GPU share main memory. The addition of a high definition display also increases system power consumption due to the extra power required to update and illuminate the more complex display. This is in addition to the increased workload experienced by the GPU, which must render the content displayed on the now expansive screen.

The additional resources required by a high density displays can be justified if the result is an improved user experience - i.e. visibly sharper content and interfaces. However, even though the iPhone 4 utilized pixels which were purportedly too small to individually observe with the human eye at 326 pixels per inch (ppi), display densities have continued to increase [1]. As an example of this, the recently released LG G3 utilizes a display with 538 ppi [2], while the Galaxy S5 LTE-A utilizes a 577 ppi display [3]. Clearly, an examination of the tradeoff between screen density and power consumption is required.

In this paper, we utilize existing knowledge of the human visual system in an attempt to define the maximum observable display density in multiple use cases (Section 2). Using this, we then examine if a system able to dynamically adjust display resolution can be utilized to reduce system power consumption with the changes remaining imperceptible to a user (Section 3). We further the investigation by additionally examining how dynamically adjusting rendered frame rate while a system is being utilized can be used to accomplish similar goals (Section 4). Both proposed methods are examined exclusively from the perspective of a mobile device's GPU. We also briefly discuss recent, related works (Section 5), and how our proposed methods may be implemented on modern mobile devices (Section 6).

2. Human Visual Acuity and Display Density

Human visual acuity depends on three main factors: (i) the ability of the eye to correctly focus on an image, (ii) the functionality of the retina, and (iii) the ability of the brain to sample and process incoming data. As technology does not yet exist which can test each of these factors individually, existing measurements of human perceptive ability test combined functionality of all three. This is generally accomplished by utilizing a Snellen or Landolt C chart, on which multiple, similar characters of set sizes are printed. A user is then asked to stand a set distance from the chart and identify the smallest characters that are legible to them. A human adult is considered to

have normal vision when they are able to correctly identify 8.86 mm tall characters on the chart when standing 20 feet away.

While these tests provide a useful baseline in attempting to define expected perceivable detail, describing the objects in question in terms of both their real size and distance from an observer quickly becomes tedious, and is also of little use in this case as mobile devices are likely viewed from much closer than 20 feet away. Instead, a more useful method of describing and object's observed size is to define it in angular terms.

An object, depending on its actual size and distance from an observer, takes up a portion of the visual field which can be described in terms of angular size. The observed size of an object relative to a user can be easily and portably described using the equation

$$\delta = 2\tan^{-1}\left(\frac{d}{2D}\right) \quad (1)$$

where d is the actual size of an object or feature, D is its distance from the observer (both measured with the same unit), and δ is the angular size of the object in radians.

When the characters on the Snellen chart are described in this manner, each one subtends an area of 5 arc minutes (5/60 of a degree) by 5 arc minutes within the subject's field of view. However, the chart is designed so that the very similar characters can only be correctly identified if 25 critical, contrasting subregions within each one is clearly perceived. Each of these subregions is one square arc minute in size, and represents the smallest individually observable detail to a human with normal vision. This provides a quantitative basis for selecting an appropriate display density on a mobile device – each display pixel should, at its smallest, be one square arc minute in size; shrinking pixels beyond that would result in no perceptible benefit.

While it may be relatively simple to create a device with a display density high enough to be held, for example, 8 inches away from a user's face without any pixels being individually visible, recall that angular size of an object depends on both the actual size of the object as well as its distance from an observer. It is unlikely that a real-world user will always hold a device that exact same distance from their face. With mobile devices especially, the distance a user is from the screen likely varies depending on when and where it is being used. Further, different users likely have different preferences regarding interaction with a device, making the common viewing distance highly varied within a population. Because of this, a more accurate description of real-world use is to define a range of distances that a device is likely to be viewed from. These ranges are defined by industry guidelines and are based on device form factor and size. Table 1 lists these ranges, in addition to the minimum and maximum observable pixel density calculated from this range using Equation 1.

Figure 1: The test bench setup with Monsoon Power Monitor connected to Nexus 4 and Nexus 5.

Interestingly, Table 1 indicates that no matter the form factor or viewing distance of the display, pixels become smaller than the smallest human-observable feature long before the display reaches UHD resolution. This means that the emerging industry standard of "4K" resolution is much more beneficial to the marketing departments of electronics companies than it is to any end user.

The table also serves to illustrate that the level of discernable detail changes dramatically as the user moves closer to and farther from the display during normal use. For example, in the best-case scenario (device is farthest from the face), a tablet's GPU need only render images to 40% the number of pixels required in the worst-case scenario (minimum distance from the face) in order to maintain the same perceived level of sharpness.

3. Dynamic Resolution Scaling

Based on the calculations performed in Table 1, it is clear that existing mobile displays already allow for a level of

Device	Viewing Distance		Pixel Density		Screen Size	Display Resolution	
	Min	Max	Min	Max		Min	Max
Phone	8 inches	12 inches	286 ppi	430 ppi	4.7 inches	1173x660	1760x990
Tablet	10 inches	16 inches	215 ppi	344 ppi	8 inches	1498x843	2397x1348
Desktop	20 inches	40 inches	86 ppi	172 ppi	21 inches	1573x885	3146x1770
Television	84 inches	133 inches	26 ppi	41 ppi	70 inches	1577x887	2496x1404

Table 1: Maximum observable display densities when utilizing devices of varying form factors and holding them the industry recommended distance from the face [4][5]. The right side of the table also includes the display resolution required to achieve this density on an example screen of a given size.

detail beyond that perceptible to human beings. Even on devices with comparatively lower density displays, the rendered resolution is likely still over-provisioned when the device is held far from the user's face. This can be used as inspiration for a new power saving technique we term "Dynamic Resolution Scaling" (DRS). This technique reduces the workload placed on a device's GPU by dynamically adjusting the display resolution during use in order to render only the level of detail that is observable given the user's current viewing distance.

To determine if the expected power savings are significant, a test bench is constructed utilizing LG Nexus 4 and Nexus 5 development devices running Android 4.4.2. To record power utilization, each device's battery is removed allowing it to instead be powered by a Monsoon Power Monitor (Figure 1). In both cases, all non-essential applications are removed from the system.

An Android benchmarking application is created which allows for the generation of an arbitrary computational load for both the CPU and GPU. The former is accomplished though a simple busy loop, the latter by creating an OpenGL scene consisting of a single 3D cone composed of a user-designated number of vertices/triangles.

To isolate the power consumed by the GPU, the power utilized by the rest of the system while idling is first recorded. GPU power consumption is then determined by subtracting this baseline from the total system power consumption while a 3D scene is being drawn at 60 frames per second (fps).

Multiple modifications are made at both the application and system level in order to reduce measurement variation. First, the benchmarking application is coded to run with the display of the device turned off, removing any variance from advanced display power scaling techniques. Second, the CPU is kept in an active state via a wake lock, and all cores but one are disabled. Dynamic voltage and frequency scaling are disabled for both the CPU and GPU to provide a more stable power profile while the tests are being conducted. Finally, all tests are conducted with airplane mode enabled and sound output disabled on both devices.

As Android 4.4.2 is OpenGL ES 2.0 compatible, both the vertex and fragment shaders of the GPU are programmable. Vertex shaders govern calculations that are performed for each vertex in a 3D scene, while fragment shaders define operations performed for each output pixel. In the benchmarking application, vertex shaders perform 9000 floating point operations per vertex, while the fragment shaders simply assign a previously calculated color value to each pixel. The artificial load was added to the vertex shaders as without it, it was found that memory bandwidth became the GPU's bottleneck far before possible floating point operations per second (FLOPS). Conversely, fragment shader complexity was kept artificially low so that collected data would represent the lower bound of power reduction possible through DRS.

Figures 2 and 3 display the data gathered while utilizing the benchmarking application. In each figure, the GPU utilization when rendering to different density displays is recorded for scenes of varying complexity (number of triangles). As may be expected, a higher resolution display results in higher GPU utilization, indicating that a dynamic resolution scaling scheme would serve well to reduce system power consumption. It is interesting to note that, based on the utilization results in Figure 2, the graphics hardware or drivers on the LG Nexus 4 have been specifically optimized for rendering at the display's native resolution of 768x1280, or 317 ppi. In such a case, DRS would actually increase GPU power consumption, even if display density was reduced. However, as the data recorded from the Nexus 5 shows no sign of similar optimizations, it is concluded that this is either a special case, or that more recent and complex device designs have abandoned such low level optimizations.

Using the power consumption data recorded at the same time as GPU utilization, it is possible to compare the total power cost that increasing display density incurs on

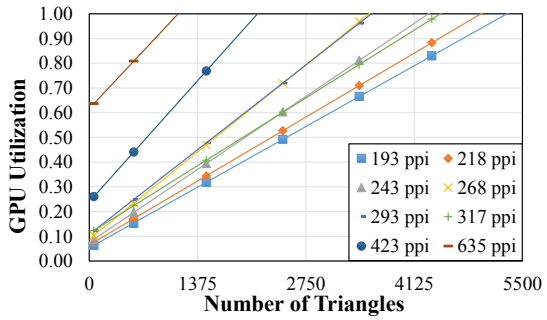

Figure 2: Recorded GPU utilization for Nexus 4 for different screen resolutions.

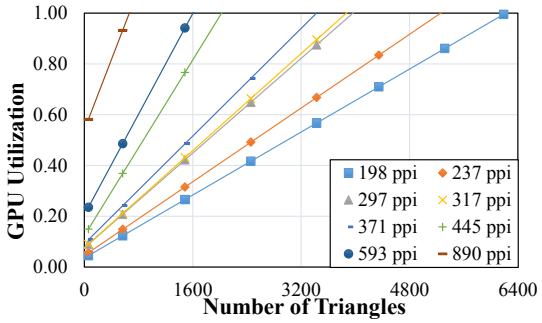

Figure 3: Recorded GPU utilization for Nexus 5 for different screen resolutions.

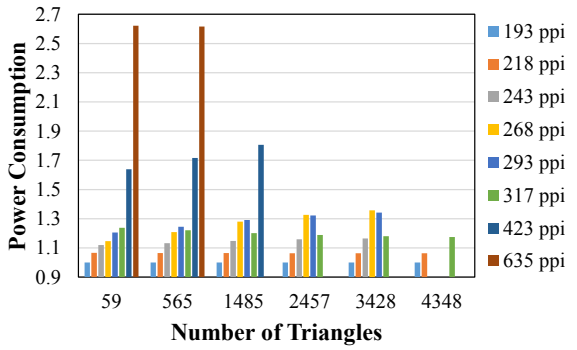

Figure 4: Relative GPU power consumption for different display resolutions on Nexus 4.

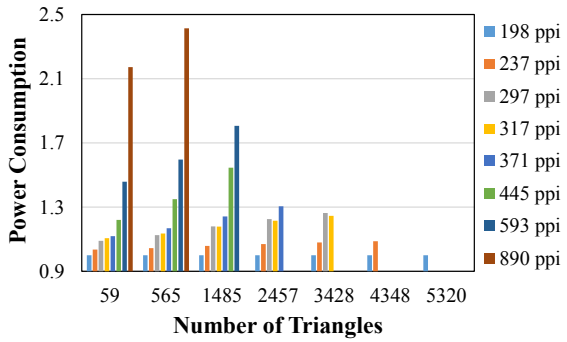

Figure 5: Relative GPU power consumption for different display resolutions on Nexus 5.

the GPU. Figures 4 and 5 show the power consumption of the Nexus 4 and Nexus 5 GPU when rendering at different densities. In each case, the power consumed by each device is normalized about the power required to render to the lowest tested display resolution. This is done in order to simplify comparison between the power levels, and because actual power consumption would likely vary greatly between different devices. Again, as expected, power consumption increases along with display density. It is interesting, however, to combine this data with the values in Table 1, as it is then possible to determine the difference in power cost from when each phone is held the minimum to the maximum distance away from the face (286 ppi to 430 ppi). For the Nexus 4, this incurs an increase of anywhere from 36-50% in the GPU's power envelope, and anywhere from 12-31% for the Nexus 5. Again, it is worth mentioning that these results are the lower bound of what would be observed in real-world situations as the fragment shaders have remained artificially simplified.

4. Dynamic Refresh Rate Scaling

A further source of GPU load is the target frame rate the system attempts to render at. By default, the Android operating system runs natively at 60 fps. This provides for a very smooth and "buttery" user experience, yet incurs a high overhead in power costs as the GPU must actively render the application or user interface 60 times per second. This problem is compounded with the with new, high resolution displays which require far more calculations and resources to populate.

Given these concerns, it may not be entirely required or even necessary to always update the display at 60 fps. While 60 fps may remain in place while a device has an excess of both system resources and battery power, the frame rate could be reduced in restrictive cases. A "Dynamic Frame Rate Scaling" (DFRS) system is proposed that will use information gathered regarding individual usage patterns to intelligently reduce system frame rate in order to maximize battery life. Such a case could occur when, for example, a mobile device falls below 10% battery power remaining. On this trigger, the system could reduce the target frame rate from 60 fps to 30 fps, impacting user experience but increasing the usable lifespan of the device.

While a reduction in frame rate from the proposed system may be noticed by a particularly perceptive user, the industry has shown that users will be willing to accept it if they gain something in return. For example, while most modern games run at 30 fps on console and films are shown at only 24 fps, users accept it as in return they gain increased image quality and graphical fidelity.

To verify that a significant amount of power would indeed be saved by such a method, the benchmarking application utilized to record the data in Section 3 was modified so that it could be set to render the 3D scene at either 15, 20, 30, or 60 fps. GPU utilization and power consumption were then recorded at each frame rate when rendering a scene at the device's native resolution, and are shown in Figure 6. As may be expected, rendering at 60 fps required 2x the processing time as rendering at 30 fps, 3x rendering at 20 fps, and 4x rendering at 15 fps.

Although rendering at 60 fps fairly predictably required 4x the processing time as rendering at 15 fps, it only required between 1.6x and 3.0x as much power, as can be seen in Figure 7. Even with this being the case, a reduction from 60 fps to 30 fps still causes, on average, a 38% decrease in GPU power consumption.

5. Related Work

In [6] and [7], the authors provide a detailed analysis of the effects of display resolution and refresh rate on users playing a PC game or watching television. In [8] it is proposed that lower frame rates may lead to system power reduction, while in [9] and [10] it is explicitly demonstrated that this is the case.

The methods in our paper more thoroughly investigate a mathematical relationship between perceived detail and display density. DRS and DFRS also more directly focus

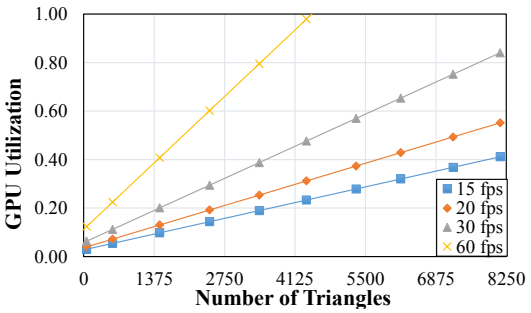

Figure 6: Relative GPU utilization for different refresh rates on Nexus 4.

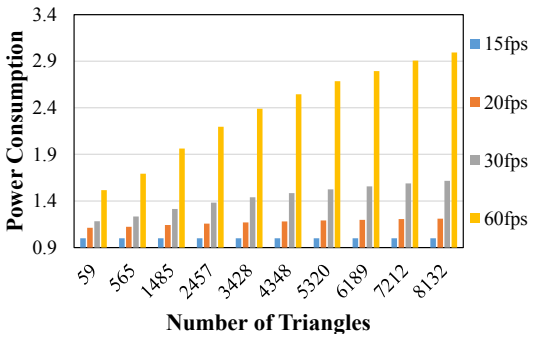

Figure 7: Relative GPU power consumption for different refresh rates on Nexus 4.

on reducing power consumption of the GPU by reducing workload, with emphasis on avoiding any negative impacts to user experience.

6. Implementation and Future Work

After it is verified that both DRS and DFRS can conserve meaningful amounts of power, it is investigated if it is possible to implement such mechanisms on existing mobile devices. As Nexus devices are used for testing up until this point, the target operating system is selected as Android 4.4.2. To implement DRS, relatively few changes are required to the operating system as Android already allows for a user to set both the display resolution and density. DFRS, however, requires the modification of Android's SurfaceFlinger binary and Settings application. In stock Android, SurfaceFlinger handles updating the display by synchronizing with a hardware v-sync signal and coordinating system draw events based on that. To implement DFRS, SurfaceFlinger is modified so that it instead updates the display based on a software interrupt which is user-definable through the Settings application.

Implemented in this manner, DFRS is entirely feasible to integrate into existing systems as there is no perceptible lag or stutter when a new target frame rate is set. DRS, however, would require a large system overhaul to work seamlessly on modern mobile devices, as the existing implementation requires restarting the SurfaceFlinger service, momentarily disabling the GUI altogether.

In future studies, we intend to implement a method of automatically determining a user's distance from their device similar to the one discussed in [11]. Use case studies utilizing the developed technique would provide insight as to how often users change their distance from a device, how much computational overhead is incurred by polling for such a change, and the seamlessness of density and frame rate changes in such a case.

References

[1] "Apple - IPhone." http://www.apple.com/iphone/features/retina-display.html. June 2010.

[2] "LG G3: Simple Is The New Smart." http://www.lg.com/us/mobile-phones/g3. June 2014.

[3] "Samsung Launches World's First Broadband LTE-A Smartphone." *Samsung Electronics America*. http://www.samsung.com/us/news/23327. July 2014.

[4] "OSHA Ergonomic Solutions: Computer Workstations ETool." https://www.osha.gov/SLTC/etools/computerworkstations/components_monitors.html. July 2014.

[5] "Get the Best Seat in the House." *Toshiba Direct*. http://www.toshiba.com/us/recommended-tv-viewing-distance. July 2014.

[6] Claypool, Mark, Kajal Claypool, and Feissal Damaa. "The effects of frame rate and resolution on users playing First Person Shooter games." *Electronic Imaging 2006*. International Society for Optics and Photonics.

[7] McCarthy, John D., M. Angela Sasse, and Dimitrios Miras. "Sharp or smooth?: comparing the effects of quantization vs. frame rate for streamed video." *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2004.

[8] Bhowmik, Achintya K., and Robert J. Brennan. "System-level display power reduction technologies for portable computing and communications devices." *Proc. IEEE Int. Conf. Portable Information Devices*. 2007.

[9] Pathania, Anuj, et al. "Integrated CPU-GPU Power Management for 3D Mobile Games." *Proceedings of the 51st Design Automation Conference*. ACM, 2014.

[10] Mochocki, Bren, Kanishka Lahiri, and Srihari Cadambi. "Power analysis of mobile 3D graphics." *Proceedings of the conference on Design, automation and test in Europe: Proceedings. European Design and Automation Association, 2006*.

[11] Konig, Immanuel, Philipp Beau, and Klaus David. "A new context: Screen to face distance." *Medical Information and Communication Technology (ISMICT), 2014 8th International Symposium on*. IEEE, 2014.