3D Res (2017) 8:2
DOI 10.1007/s13319-016-0111-x

3DR EXPRESS

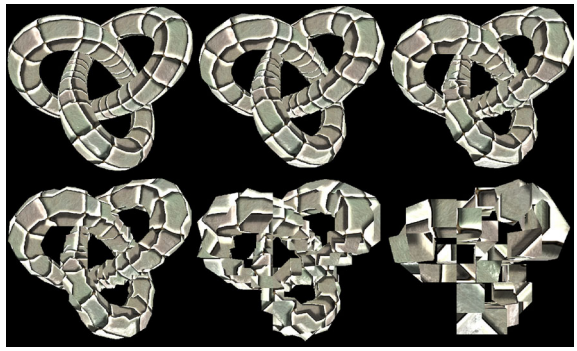
Simplification of 3D Graphics for Mobile Devices: Exploring the Trade-off Between Energy Savings and User Perceptions of Visual Quality

Jarkko Vatjus-Anttila · Timo Koskela · Tuomas Lappalainen · Jonna Häkkilä

Received: 18 August 2016 / Revised: 21 November 2016 / Accepted: 7 December 2016

Abstract 3D graphics have quickly become a popular form of media that can also be accessed with today's mobile devices. However, the use of 3D applications with mobile devices is typically a very energy-consuming task due to the processing complexity and the large file size of 3D graphics. As a result, their use may lead to rapid depletion of the limited battery life. In this paper, we investigate how much energy savings can be gained in the transmission and rendering of 3D graphics by simplifying geometry data. In this connection, we also examine users' perceptions on the visual quality of the simplified 3D models. The results of this paper provide new knowledge on the energy savings that can be gained through geometry simplification, as well as on how much the geometry can be simplified before the visual quality of 3D models becomes unacceptable for the mobile users. Based on the results, it can be concluded that geometry simplification can provide significant energy savings for mobile devices without disturbing the users. When geometry simplification is combined with distance based adjustment of detail, up to 52% energy savings were gained in our experiments compared to using only a single high quality 3D model.

Graphical Abstract



Keywords Mobile · Geometry · Three-dimensional · Energy consumption · Rendering · User experience

1 Introduction

In 1997, Shafi et al. [41] predicted a need for dense, efficient and compact energy storage (i.e. batteries). Such storage would enable the existence

J. Vatjus-Anttila (✉)
Cyberlightning Ltd, Teknologiantie 2, 90570 Oulu, Finland
e-mail: jarkko@cyberlightning.com

T. Koskela
Center for Ubiquitous Computing, Faculty of Information Technology and Electrical Engineering, The University of Oulu, Oulu, Finland

T. Lappalainen · J. Häkkilä
Faculty of Art and Design, University of Lapland, Yliopistokatu 8, 96400 Rovaniemi, Finland

Published online: 21 December 2016

of portable devices in the first place, and secondly, a decent operation time when not connected to any external energy source. We know how the era of mobile devices kicked off after 1997, which makes this prediction very accurate. In addition, Shafi et al. [41] estimated that the battery capacities will increase approximately by 4% annually, which was later verified by Belleville et al. [5] in 2009.

Gene's law describes how the technology evolves and makes processing of the data more energy-efficient gradually over time. This phenomenon in general is measured by energy/bit, and its decrement rate is approximately 1.6× annually [5]. The increment in the processing efficiency enables building more complex applications, and because of this, the total average energy consumption of the applications is not likely to decrease. Taking into account the annual increase of 4% in battery capacity and the tendency of the batteries to wear over time, it will not be enough to cover the need for additional energy by relying on the development of the battery technology only [38, 24].

In the field of 3D graphics, measuring energy consumption of a 3D application is a rather new aspect [29]. This is due to the fact that 3D graphics have been used for decades mainly with desktop computers that are connected to the grid. Only recently, the capabilities of battery-powered mobile devices have evolved to the level that enables fluent presentation of 3D graphics. Although energy efficiency as such is a rather new optimization parameter for 3D applications, a great deal of research has already been conducted on the simplification methods for 3D graphics to increase their suitability for devices with varying capabilities [22]. Firstly, 3D graphics are complex to visualize, and therefore, hundreds of studies have focused on lowering the rendering burden on the client [35]. Secondly, 3D graphics are typically large in file size, which has also inspired research to reduce their network bandwidth requirements [9]. For mobile devices in particular, use of appropriate simplification methods is essential due to the limitations in their processing capacities, wireless network bandwidth, and most of all, in the battery capacity [22].

In the simplification of 3D graphics, one can change the characteristics of mesh geometry [35], surface textures [43] and/or surface materials [27]. By simplifying any of these components of 3D graphics, energy savings can be achieved [22]. In this paper, we particularly focus on geometry simplification due to the two following facts: (1) the processing of the geometry has been discovered to be the most energy consuming part in the rendering process, at least with complex 3D scene structures [29]; and (2), when high quality textures are used, they can hide the artifacts in the simplified geometry [6]. For evaluating the visual quality of the simplified geometry, several objective (i.e. mathematical) metrics exist that measure, for instance, the root mean square error between the original and the simplified 3D model [26]. In few studies, the objective metrics have also been accompanied with subjective (i.e. user studies) metrics to benchmark the mathematical interpretations of visual quality against users' perceptions [6, 48]. However, no studies yet exist that examine the results of geometry simplification in terms of gained energy savings and reflect these gains with users' perceptions of visual quality of the simplified 3D models. As discovered by both Ickin et al. [19] and Heikkinen and Nurminen [14], energy consumption has a significant influence on users' experiences in mobile devices and applications.

In this paper, we investigate how much energy savings can be gained in the transmission and the rendering of 3D graphics by simplifying geometry. In addition, we examine users' perceptions on the visual quality of the simplified 3D models. For the simplification of the geometry, we implemented a modern method [28] that is very lightweight to execute at the client making it highly suitable also for mobile devices. By combining the results of both our studies, we can provide (1) new knowledge on the significance of energy savings that can be achieved through geometry simplification, and (2) guidelines on how much the geometry can be simplified for the sake of energy savings before the visual quality of 3D rendering becomes unacceptable for the mobile users. Based on our findings, we also propose new energy savings features for future 3D applications.

The rest of the paper is organized as follows. Section 2 presents the related work, Sect. 3 describes the experimental setup, Sect. 4 introduces the results of the experiments, and finally, Sect. 5 discusses the significance of the results and concludes the paper.

2 Related Work

2.1 Research on Evaluating the Visual Quality of Simplified 3D Models

In the literature, there already exist several studies that use subjective (i.e. user studies) and/or objective (i.e. mathematical) metrics for evaluating the quality of simplified 3D models. Next, the key findings of these studies are summarized.

Rushmeier et al. [40] use a subjective metric to examine the perceived quality of various representations of textured, geometric objects that were viewed under different lighting conditions. The various representations of the test objects were compared with the original object (the highest quality) and scored by users using a scale of 0–100. The score was then examined in relation to the file size of the simplified texture and geometry data. Rushmeier et al. [40] concluded that replacing geometry with texture can be very effective, but in some cases, the addition of texture may even reduce the perceived quality.

Rogowitz and Rushmeier [39] use a subjective metric to evaluate the perceived quality of simplified 3D models using both 2D images of the 3D objects and animated 3D objects. The users scored the various representations of the test objects using a scale of 0 to 10. According to their study, some differences between the scoring of the static images and the animated 3D objects were consistently visible, but as Rogowitz and Rushmeier [39] concluded, further studies on this topic should be conducted. What was clearly shown in their results is that the perceived quality of a 3D object was significantly higher when the object was animated if that 3D object was lit from the front. This is probably due to the fact that the human visual system is less sensitive to high spatial frequencies when an object is moving.

Lavoue and Corsini [26] present a comparative survey on different objective metrics for evaluating the quality of simplified 3D models. They conclude that perceptually-motivated metrics perform typically better than the standards geometric ones such as root mean square error or Hausdorff distance.

Bulbul et al. [6] present a taxonomy of both subjective and objective metrics that can be used for evaluating the quality of simplified 3D graphics. They conclude that use of subjective metrics outperforms the objective metrics, but their use is also more costly and may not be practical for all application scenarios. In addition, Bulbul et al. [6] point out that textures have a significant impact on the experienced quality of the 3D models, since they can be used for hiding the artifacts on the geometry.

Pan et al. [34] designed a user study and derived an objective metric that approximate the perceived quality of 3D graphics. They also reviewed the essential factors determining the perceived quality of 3D graphics. In their experiments, they used five different 3D objects and each object was represented by the total of nine levels: six levels of wireframe resolution and three levels of texture resolution. The task for the users was to compare the rating object to two referential objects. Pan et al. [34] conclude that users are more sensitive to the distortion of texture compared to that of surface geometry. They also discuss that fewer stimuli produce more reliable results and the large number of stimuli affect the fatigue and frustration of the participant.

Silva et al. [42] designed a new subjective metric to assess the perceived quality of polygonal meshes. In their experiment, Silva et al. [42] had five 3D models of which each was represented by six levels of simplification. The users were simultaneously presented four objects along with the referential object and the users were asked a preference order for the four objects. Silva et al. [42] also encountered user fatigue, which was caused by the 3D model repetition, but not by the number of comparison cases.

Watson et al. [48] examined both subjective and objective metrics for measuring and predicting visual fidelity of 3D models. In the experiment, 36 different 3D models were used of which each was varied with two different simplification algorithms. 3D models consisted of man-made artefacts and animals, none of the 3D models contained surface texture or color. In the experiment, the users had three tasks: first task was to name each 3D model as quickly and accurately as possible. Second task consisted of comparing and rating four 3D objects with different levels-of-detail (LODs). In the third task, the users were asked to compare pictures of the same object with different LODs and choose the picture which was the best in the set. Watson et al. [48] also discussed in their work that focus was on recognition of isolated objects, which is different from more natural scenes containing several objects in their context.

As shown in the related work, there already exist studies that combine both subjective and objective metrics for evaluating the visual quality of 3D graphics. However, none of these considers the degree of energy savings achieved through 3D graphics simplification in terms of 3D graphics delivery and/or 3D rendering. Therefore, this paper provides a novel viewpoint to the research field of 3D graphics simplification, particularly in the mobile realm.

2.2 Research on Geometry Simplification for Mobile Devices

Over the years, simplification of 3D graphics and geometry, in particular, has attained a significant amount of interest in the research community. The earliest works date back to the 1990′s, when Hoppe [15] presented the original edge collapse method that allows progressive delivery and reconstruction of geometry. Majority of the geometry simplification methods have been originally designed for desktop computers, but several of them are also suitable for mobile devices [22]. As the number of existing geometry simplification methods is vast, only the different categories of them are now briefly summarized together with a modern method particularly suitable for the mobile devices.

In general, the simplification of geometry can be done (1) by reducing the geometry (i.e. vertices) and/or the connectivity (i.e. triangles) data or (2) by compressing the geometry data without changing the connectivity data [35]. With the geometry simplification methods, it is possible to produce either discrete or progressive LODs [22]. With discrete LODs, a known number of geometry variants with different visual quality are created (e.g. [49, 50]). Discrete LODs are independent pieces of geometry that do not require any decoding on the CPU, but they typically contain significant amounts of redundant information, which may lead to high bandwidth and memory requirements overwhelming the capacities of a mobile device. With progressive LODs, geometry variants are determined by the percentage of original geometry that has been progressively transmitted to the client (e.g. [15, 28, 25]). Progressive LODs typically require substantial amounts of decoding on the CPU, which makes their use burdensome for mobile devices. The example methods mentioned focus on purely simplifying the geometry without the large scale contextual awareness about the geometry data. If the optimization method has awareness of the context (being for example a city 3D model), the mentioned methods can be tailored to be especially suitable for that particular dataset. This kind of approach has been shown by Zhang et al. [52] and Chang et al. [7] with urban city 3D models. Another specialized method tailored to terrain type geometry (typically presented as heightmap grid) has been introduced by Atlan and Garland [3]. The research shows that geometry optimization is an approach which can be tailored further by understanding the type of the geometry data and preferably the context in which it is being used.

Limper et al. [28] introduce a modern geometry simplification method that enables progressive delivery of geometry, but also decoding at the client without additional workload, which is very advantageous from the standpoint of mobile devices. Their method, however, is not purely a progressive one, since it does not decode the geometry data one vertex/edge/triangle at a time, but rather as groups of geometry. As a result, each new increment of geometry can also act as a new discrete LOD enabling, for instance, dynamic adjustment of LODs based on the viewing distance. With a typical progressive method, changing the LOD would require back and forth decoding and encoding of the geometry data by the CPU. For this paper, we implemented the method described by Limper et al. [28] in a custom 3D renderer software and created three test 3D objects (and scenes) for the experimentation. This allowed us to experiment with both (1) progressive loading of the 3D graphics to the renderer, and (2) switching of LODs during the rendering without additional decoding/encoding.

2.3 Research on Energy Consumption of Mobile Devices in 3D Graphics Rendering

According to Xiao et al. [51], (1) the mobile chipset (CPU, GPU and memory); (2) the screen; and (3) the wireless radio interfaces each roughly accounts for one-third of the total energy consumption of a mobile device. Therefore, the use of simplification methods can have a substantial influence on the overall energy consumption as it affects the use of both the mobile chipset and the wireless radio interfaces.

It should be noted, however, that the simplification methods are only as good as how well they suit for the

underlying hardware. Ma et al. [29] performed energy consumption measurements with three modern games on three different mobile devices. They split the energy consumption between various stages of the graphics pipeline, and came to a conclusion that vertex processing stage is the single most energy consuming activity during the rendering. This also suggests that the CPU might be a bottleneck for the whole process in large 3D scenes as the CPU needs to continuously feed new geometry for the GPU if (1) all of the 3D scene geometry does not fit into the GPU memory at once, and/or (2) the CPU needs to perform deformation to the geometry (for example, due to destruction simulation in the 3D scene). While this might be true for these particular cases, one should also acknowledge the influence of the heterogeneous nature of the mobile GPUs in the market.

Hosseini et al. [16] make a statement that the energy consumption of a mobile device can be optimized runtime by adjusting the content in use in the foreground application. First, they state that, with Nexus One mobile device, the maximum energy consumption of the display comprises roughly 60% of the total energy consumption. However, this figure does not include the GPS and the GPU, although the GPU is one of the most significant contributors to the total energy consumption. Hence, they address the display energy consumption indirectly by altering the rendering process lighting calculations (lighting limitation) and by adjusting certain texture files to darker tone (textural adjustment). The former method simplifies the rendering load by dropping away detailed lighting calculations and the latter one manipulates the textures images to be darker, as the darker colors tend to lower the display energy consumption. Their methods are simple, but their findings are nevertheless interesting since they claim to achieve 20–33% energy savings via this manipulation without the user noticing the difference.

Johnsson and Akenine-Möller [20] describe a non-intrusive, high frequency, external test method which they use to measure energy consumption of various devices from mobile (iPhone 4S) to desktop computers (Intel Core i7 class CPU). In their work, they use 3D rendering as a workload in their experiments. Johnsson and Akenine-Möller [20] use both unmodified 3rd party software, as well as synthetically build custom software. Their recommendation for energy measurement setup is to (1) record the energy consumption at a high frequency, (2) be aware that the operating system will interfere with the measurements (relates to understanding your platform used in the experiments), (3) be aware of the higher energy consumption taking place during the first few seconds of the application run and (4) avoid changes to the workload during the experiments. In general, they describe a comprehensive setup, but in real-life, not all behavior of the underlying OS is known to the application (especially when going into device driver details) and non-changing workloads do not match with the real world application behavior. However, all of the remarks Johnsson and Akenine-Möller [20] make should be acknowledged prior to conducting any measurements.

3 Experiment Setup and the Definition of Experiments

This section first describes the created experimental setup, including the test devices, the 3D renderer, the 3D test objects, the energy measurement setup and the user study setup. Second, the energy measurement experiments are explained in detail.

3.1 Test Devices

The energy measurements were conducted using three different mobile devices. The chosen test devices were Samsung Galaxy S2+ [11], Samsung Galaxy S3 [12] and Samsung Galaxy S5 [10].The user study was implemented using S5. The selected mobile devices represent three different generations, and hence three different performance levels. It can be defined that S2+ is a low-end mobile device with not much of processing power compared to nowadays standards, whereas S3 is a midrange and S5 is a high-end device. This applies from both available processing power and pricing viewpoints. Timewise, S2+ was released January 2013 and S3 in May 2012. S2+ is, however, a facelift of another low-end product (Samsung Galaxy S2) which was originally released in February 2011 [13] and is based on older generation hardware than S3. Finally, S5 is a device released to the market in February 2014.

Availability of mobile devices with different performance levels was important for the experiments. This is based on the fact that rendering load

optimization was expected to have a greater influence on the cases where the mobile device had difficulties in performing well with a given workload. The main differences of the selected devices are summarized in Table 1. It is worth noting that not only the plain comparison of the CPU and the GPU power matters in this experimentation. An important aspect as well is the display resolution. In the experiments, all 3D graphics were rendered in the device's native resolution, which leads to a fact, for example, that S2+ had to draw less than its counterparts only because it had to fill in fewer pixels on the display.

3.2 3D Renderer

From the rendering process perspective, it is important to understand the generic programmable GPU architecture (Fig. 1). The same architecture is in use for each test device and it is accessed through OpenGL ES 2.0 graphics API. The API exposes high level access to the programmable graphics hardware to allow the application to store data into the GPU memory, adjust rendering settings and issue draw calls. All geometry is first processed by the programmable vertex shader unit. After vertex transformations, a primitive assembly is performed in which some pieces of the 3D graphics (for example, off-screen geometry) are dropped from the pipeline. The fragment processing stage (the shader and per-fragment operations) are executed for the remaining pieces of the 3D graphics.

The 3D renderer used in all of the experiments was a custom-built OpenGL ES 2.0 C++ implementation. When implementing the renderer, its primary goal was to be as efficient as possible when dealing with the workloads of progressively downloaded geometry. Also, from the rendering process perspective, the methods of the OpenGL ES 2.0 API were utilized in a way that they caused as low overhead as possible (for instance, memory copying or duplication was minimized), but only those features were utilized which were supported by all test devices. The primary methods used by the renderer were (1) cached geometry with Vertex Buffer Objects (VBO) and (2) custom shaders which implemented basic illumination models per-vertex and per-fragment basis, and special effects such as texture mapping [45], normal mapping [8], and parallax occlusion mapping [36]. It should be noted that the VBO cache implementation of the renderer was built to fully support progressive downloading of the geometry. Hence, by knowing the focus of the implementation, the renderer was using caching as efficiently as possible by minimizing all memory (re-) allocations and copying. This was made to ensure the renderer takes full advantage of the availability of the progressive geometry data.

Additionally, the scene management associated with the renderer used regular methods for accelerating the scene management process, like view frustum culling [2]. Because the complexity of the 3D scene structure and the number of 3D objects in the test scenes were relatively small, there was no need for more complex scene management, such as using octrees [37]. An advanced rendering method of instanced rendering [31] was not used even though it would have been useful in this kind of case where multiple similar objects are rendered repeatedly. The main reason for not using the method was the lack of support for it with the older test devices (S2+ and S3). Thus, using it only with one device would not produce comparable results. It should be noted, however, that typically the advanced rendering methods, such as instanced rendering, are useful because they are likely

Table 1 The hardware-level comparison of the test devices

	S2+	S3	S5
CPU	1.2 GHz dual core Cortex-A9	1.4 GHz dual core Cortex-A9	2.5 GHz quad core Krait 400
GPU	VideoCore IV	Mali-400MP	Ardeno 330
RAM (GB)	1	1	2
Display size	4.3″	4.8″	5.1″
Display pixels	800 × 480 pixels	1280 × 720 pixels	1980 × 1080 pixels
Battery (mAh)	1650	2100	2800
Operating system	Android 4.2.2	Android 4.3	Android 5.1

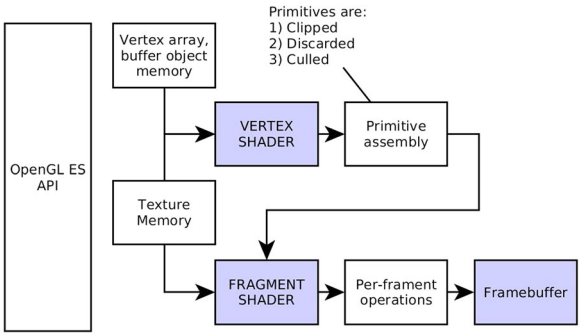


Fig. 1 A simplified programmable (OpenGL ES) graphics architecture

to reduce energy consumption. Instanced rendering, for instance, addresses the situation when multiple instances of the same geometry are rendered, the CPU can issue a single draw call to the GPU to draw all of the objects at once instead of looping through each 3D object separately. This lowers the CPU load significantly, particularly when there is a large number of the same 3D object to be rendered.

3.3 Test 3D Objects

Three different 3D objects were utilized in the experimentation: (1) Torus knot [33], (2) Stanford bunny [44] and (3) Sponza cathedral [30]. Torus knot and Stanford bunny were individual objects, while Sponza cathedral was an interior model illustrating a complete building. Regarding the complexity, Torus knot represents a relatively simple 3D object (2880 triangles and 1834 vertices in total) with additional texture maps, Stanford bunny is a more complex one (16,301 triangles and 48,903 vertices in total) but without additional texture maps, and finally, Sponza cathedral is a complex model (66,450 triangles and 61,378 vertices in total) with additional texture maps. In addition to the vertex data, Torus knot and Sponza cathedral included surface normals and tangents, whereas Stanford bunny included only surface normals. In all these 3D objects, triangles were stored as three 16-bit integers representing the corner point indices, and all other attributes (vertices, surface normals and surface tangents) were stored as vectors using 32-bit floating point numbers.

For the experiments, six distinct LODs were created for each 3D object using the quantization process presented by Limper et al. [28]. We chose six LODs to assure that (1) subsequent LODs are not too easily distinguished from each other; and (2) the participants of the user study do not get frustrated because of too large a number of alternatives. In previous user studies, listed by Bulbul et al. [6], from three to seven LODs were typically used. We also aimed to match the sizes (in bytes) of the LODs as closely as possible with the steady progressive loading process, i.e. the size difference between each subsequent LODs was kept as similar as possible.

The LODs, the sizes of LODs (in terms of bytes, triangles and vertices) are presented in Tables 2, 3 and 4 and sample images of each 3D object are presented in Figs. 2, 3 and 4, respectively. Additionally, Torus knot (1246 kB) and Sponza cathedral (3912 kB) use texture files in the surface material definitions.

3.4 Energy Measurement Setup

When externally measuring the total amount of energy consumed by a test device, the energy measurement setup needs to be non-intrusive in order to produce results with a minimal interference, i.e. external stands for the measurement equipment that are physically separated from the test devices. Non-intrusive, in turn, relates to the fact that the measurement equipment should be capable of measuring the energy consumption without interfering with the operation of a test device. Such a measurement setup is used also in this paper, and is comprehensively described, for example, by Johnsson and Akenine-Möller [21], Johnsson and Akenine-Möller [20] and Vatjus-Anttila et al. [46].

For measuring the energy consumption of the test devices, we used a power tool called Monsoon [32]. Monsoon was used to override the battery and to provide all energy to the mobile device. Hence, the results of this paper present the total energy consumption of the mobile device. During the experiments, all wireless radio interfaces were switched off and display brightness was manually turned to the minimum. This was done in order to minimize the effect of those peripherals on the experiment results. All energy consumption measurements were done using 3.7 battery voltage driven by Monsoon.

3.5 User Study Setup

This section presents the user study setup in detail, including the participants, the environment and the tasks conducted by the participants.

Table 2 Stanford bunny

LOD	Faces	Faces (%)	Increment (B)	Vertices	Vertices (%)	Increment (B)	Total increment (B)
1	1636	10.0	9816	785	1.6	18840	28656
2	3629	22.3	11958	1783	3.6	23952	35910
3	9083	55.7	32724	4515	9.2	65568	98292
4	14568	89.4	32910	7279	14.9	66336	99246
5	16297	100	10374	8145	16.7	20784	31158
6	16301	100	24	48903	100	978192	978216
Total loading (B)			97806			1173672	1271478

Table 3 Torus knot

LOD	Faces	Faces (%)	Increment (B)	Vertices	Vertices (%)	Increment (B)	Total increment (B)
1	466	16.2	2796	204	11.1	6528	9324
2	1904	66.1	8628	934	50.9	23360	31988
3	2790	96.9	5316	1393	76.0	14688	20004
4	2874	99.8	504	1437	78.4	1408	1912
5	2880	100	36	1440	78.5	96	132
6	2880	100	0	1834	100	12608	12608
Total loading (B)			17280			58688	75968

Table 4 Sponza cathedral

LOD	Faces	Faces (%)	Increment (B)	Vertices	Vertices (%)	Increment (B)	Total increment (B)
1	23024	34.7	138144	11572	18.9	370304	508448
2	31237	47.0	49278	15870	25.9	137536	186814
3	49083	73.9	107076	25214	41.1	299008	406084
4	59739	89.9	63936	31020	50.5	185792	249728
5	62519	94.1	16680	34267	55.8	103904	120584
6	66450	100	23586	61378	100	867552	891138
Total loading (B)			398700			1964096	2362796

3.5.1 Participants

To evaluate user perceptions on the visual quality of the 3D objects, a user study was organized using a within-subjects experimental design. Altogether 30 participants (64.3% male, 36.7% female) took part in the study in June 2015. Participants comprised of university staff and students, and their background was multidisciplinary. The participants were recruited through advertising on university notice boards, through social networks and some participants were recruited in situ. Participants' age varied between 24 and 59 years (median 32). 53.3% of the participants had corrected vision with eyeglasses or contact lenses, none of the participants reported having limited or impaired eyesight.

As background information related to the context of the study, participants were asked about the ownership of devices and the use of different applications with 3D graphics. 96.7% of the participants owned a

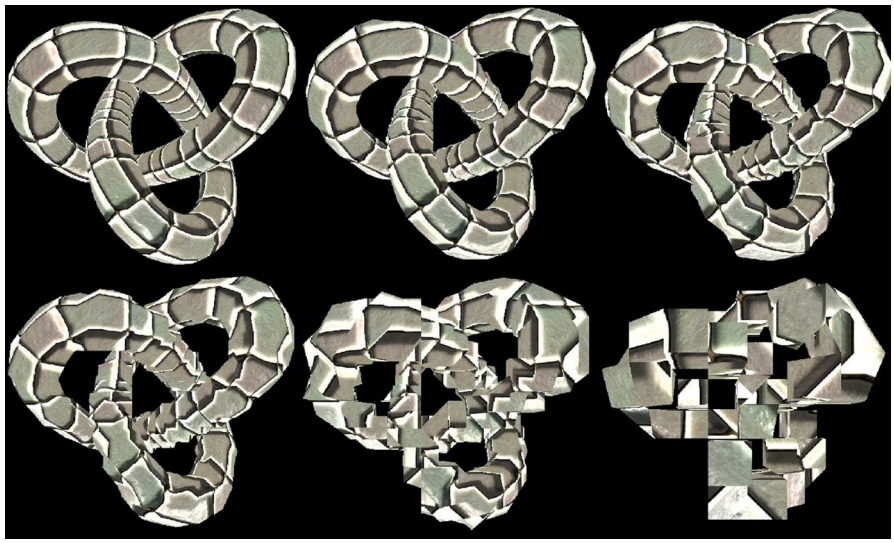


Fig. 2 Torus knot with six LODs

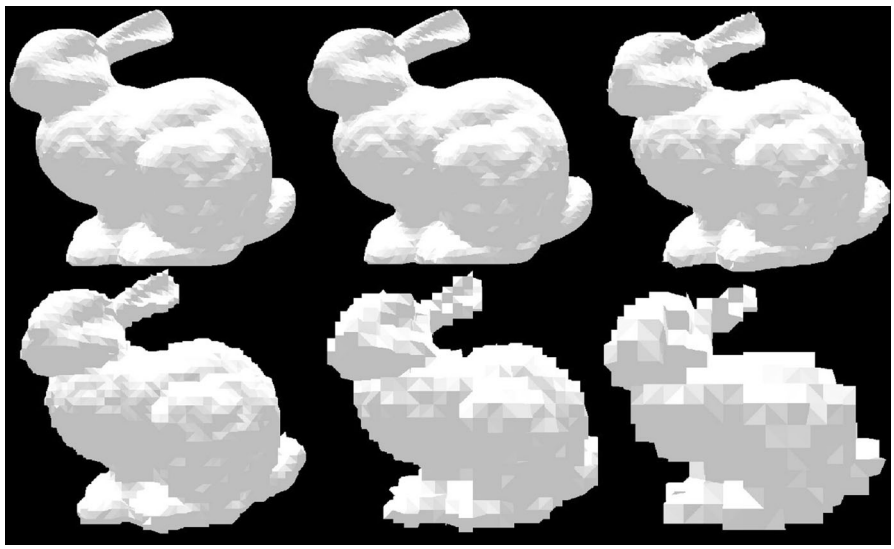


Fig. 3 Stanford bunny with six LODs

computer, 90% owned a smartphone and 56.7% owned a tablet. 33.3% of the participants reported playing 3D games on a computer and 6.7% played 3D games on a smartphone or a tablet. 30% reported using 3D modeling software on a computer and 13.3% used VR on a computer.

3.5.2 Environment

The study took place in a university classroom having an eight meters long corridor. Each session per participant lasted approximately 15 min and started with the introduction to the study. The participants first filled in a background questionnaire, including questions about prior experiences with 3D graphics on different devices. After the questionnaire, the participants proceeded with three different tasks. The order of the tasks was randomized, and before each task the participants were instructed to the task.

In tasks 1 and 2, the shown images were 2D projections of the 3D objects. In Task 3, the 3D objects were rendered in real-time. In all tasks, the used lighting conditions were static and identical for all the 3D objects. There was a single directional light source, which pointed to a direction that was 45 degrees away from each (positive) coordinate axis. As the camera

Fig. 4 Sponza cathedral with six LODs

always pointed toward the positive Z axis, the light appeared to come out behind the camera and to point towards the lower right corner of the display. The direction was chosen so that it emphasized the parallax map effect of the 3D objects, hence giving the participants an opportunity to see more details of the chosen models. However, the light source was chosen to be static so that it would not influence the perceived lighting conditions during the study, and hence did not steal the attention of the participants.

In the user study, S5 was used as the test device running custom developed test applications. S5 was chosen for this purpose as it was predicted that the visual differences between LODs would be more clearly recognizable on a large screen.

3.5.3 Task 1

The objective of Task 1 was to study how well the participants recognize the visual differences between LODs when compared side-to-side. In Task 1, the participants were shown simultaneously two images of the same 3D object with a different or the same LOD. The task was to choose which image in the comparison had the highest LOD or whether the images were equal in terms of visual quality (see, Fig. 5). Preferences have been used in the previous studies, as stated by Watson et al. [48], and as preferences represent a conscious decision, they have been proven useful for experimental measures. Participants were instructed to do the task as quickly and accurately as possible. Participants evaluated three different 3D objects, each with six different LODs, as described in Sect. 3.3. There was the total of 48 comparison sets and each 3D object was compared 16 times. Participants were sitting at a table while doing Task 1.

3.5.4 Task 2

The objective of Task 2 was to study whether it is more difficult to recognize the visual differences between LODs compared side-to-side while walking. The application in Task 2 was exactly the same as in Task 1, but the participants evaluated the 3D objects while walking. Participants walked in the corridor of the room which was approximately eight meters in length. Participants were instructed to walk their normal walking speed throughout the whole task.

3.5.5 Task 3

The objective of Task 3 was to study whether the participants were willing to use LODs with lower visual quality when they were given a choice. In Task 3, the participants could slide through six different LODs and they were asked to choose the lowest acceptable LOD for the 3D object. Participants evaluated the same three 3D objects as in Task 1 and Task 2, but here, the participants were shown real-time rendering of the 3D objects.

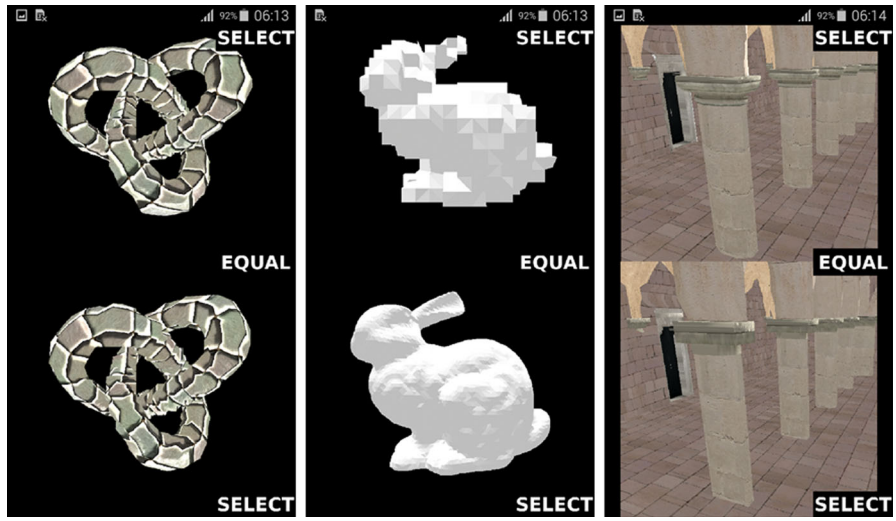


Fig. 5 Example screenshots of the application used in Task 1 and 2

3.6 Energy Consumption Experiments

The experiments concerning the energy consumption in 3D graphics rendering were performed with three different 3D scenes. The first two 3D scenes were exterior scenes built using the individual models (Torus knot and Stanford bunny). The third 3D scene was an interior scene built using Sponza cathedral. The purpose of exterior scenes was to mimic a typical scene structure of 3D applications, which use individual 3D objects to construct a full 3D scene. The interior scene creates the view using only one large 3D object, and the rendering takes place inside that object. The exterior scene structures allow exploiting, for example, distance based LOD adjustment, whereas in interior scenes this is not trivial. Not at least without splitting the interior 3D object into pieces and then applying a suitable LOD mechanism for each of those pieces.

For the exterior scenes, a total number of 24 experiments were run with a varying number of 3D objects (4, 25 and 64) and with different LOD mechanisms. These experiment variables are summarized in Fig. 6 and explained in more detail in the following paragraphs.

The experiments for the exterior scenes were organized in two ways: (1) the "crowded" scene organization, where the selected 3D object was multiplied densely around the origin; and (2) the "sparse" scene, where the 3D objects were in the same organization around the origin, but farther away from each other. It is illustrated in Fig. 7 how the two scene organizations were constructed in practice. In the "sparse" 3D scene, the proportion of the visible 3D test objects varied approximately between 20 and 80%, whereas in the "crowded" scene, the variance was higher, ranging from 0 to 100%. The primary purpose of the "crowded" scene was to exhaust the device during rendering and to examine the effect of each LOD mechanism in this kind scenario. The "sparse" scene aimed to be more like a typical 3D scene, where by design, not all content of the scene is rendered at the same time. During the measurements, we noticed that there were some differences in the energy consumption between "sparse" and "crowded" scenes, however, these were not as significant as we expected. Because of this, only the experiment results related to the "sparse" scene are presented in this paper.

The four LOD mechanisms used in the experiments are referenced as L1, L2, L3 and L4. The LOD mechanisms are summarized as follows:

- **L1**: "All min", all but the simplest LOD are ignored and this LOD is always used regardless of the viewing distance.

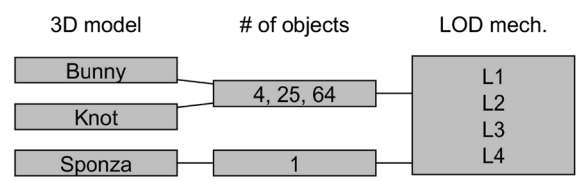


Fig. 6 All combinations of the executed test experiments

- **L2**: "Fully dynamic", takes full advantage of the dynamic LOD switching based on viewing distance.
- **L3**: "All max", all but the most complex LOD are ignored and this LOD is always used regardless of the viewing distance.
- **L4**: "Dynamic with the lowest acceptable LOD", dynamic LOD switching is used based on the viewing distance, but the highest LOD used was decided by our user study, where the users determined the lowest acceptable LOD for each test object.

Each of the experiments had two stages. (1) The first stage was the resource loading and the 3D scene construction. In this stage, all the assets (3D objects, texture files, material definitions, etc.) were loaded to the memory of the mobile device and the actual 3D scene was constructed as explained above. (2) The second stage was the actual rendering using a camera which moved through a circular route in the 3D scene. The camera circulated around the origin of the 3D scene in a way that it started with the maximum visibility to the 3D objects. In the midway of the rendering, the camera approached the minimum rendering load having the majority of the 3D objects behind the camera. Finally, the camera returned to the starting position. During its route, the camera did not turn, and hence, it was looking towards the positive Z axis all the time.

The camera circulation was adjusted in a way that the total test runtime was always 65 s providing approximately 5 s to manage the resource loading and 60 s to perform the rendering. 60 s was considered enough to stabilize the rendering load after the resource loading phase. The camera route is illustrated together with the scene organization in Fig. 7. Johnsson and Akenine-Möller [21] guideline that the resource loading part should be left out of the measurements when measuring the frame based load. In this case, we made a decision to include it, since it is an important part of the total energy consumption in progressive 3D graphics loading.

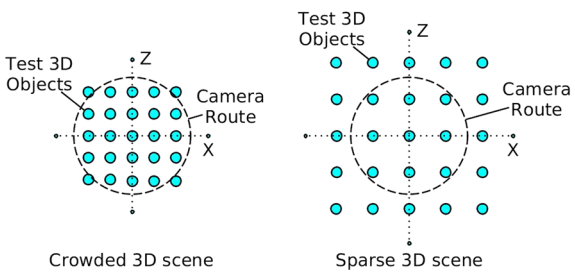


Fig. 7 3D scenes with 25 objects in the crowded (*left*) and the sparse (*right*) organizations

The interior 3D scene was constructed using a single Sponza cathedral 3D object, and the camera was placed directly inside the 3D object. In the experiment, the camera was turned around inside Sponza cathedral in a way that the whole 360-degree view of the interiors of Sponza cathedral was visible. The LOD mechanisms worked similarly to Torus knot and Stanford bunny models apart from the dynamic methods (L2 and L4). Because the viewing distance to the 3D object did not change, we were unable to switch between LODs based on the distance information. Instead, we forcefully changed the used LOD mechanism every 10 s, hence allowing each LOD to be rendered approximately for 10 s during each experiment. Figure 6 summarizes the experiment variables, also for the interior scene. In total, 4 experiments were run.

Taken all the explained test experiment combinations and LOD mechanisms into account, the resulting rendering complexity varied between the following absolute minimum and maximum vertex and face counts. Bunny: 3121-3129792 vertices and 6544-1043264 faces. Knot: 816-117276 vertices and 1864-184320 faces. Sponza: 11572-61378 vertices and 23024-66450 faces.

For each test device, the total number of 28 experiments were run. Each experiment was run twice and the results were averaged. As we used three test devices, this eventually resulted in the total of 168 individual 3D rendering experiments, totalling in 10,920 s of measurements. After each experiment, the following values were recorded: (1) the total number of rendered frames and (2) the total amount of energy consumed. These allowed us to follow the total energy consumption, but also how the invested energy per rendered frame varied between different experiments.

In addition to 3D rendering, the energy consumption of 3D graphics delivery in mobile networks (3G/UMTS and 4G/LTE) was also studied. This was done mathematically based on the estimated average download speed and average power consumption of mobile radios. The average download speed [18] and average power consumption [17] were retrieved from previous

studies, which examined the performance of 3G and 4G in real-life mobile networks.

4 Experiment Results

This section summarizes the results from the user study and the energy consumption measurements.

4.1 User Study

4.1.1 Results of Tasks 1 and 2

In Task 1 and Task 2, participants compared two images of the same 3D object having a different or the same LOD. In terms of decision time, there were no significant differences with Torus knot and Stanford bunny (see, Fig. 8). However, the comparisons appeared to be more time consuming with Sponza cathedral.

It can also be seen in Fig. 8 that the participants made the decisions faster while walking. This was potentially due to the difficulty to notice differences in the images and/or due to possible frustration towards the task as stated by one participant. This resulted in more incorrect answers on Sponza cathedral, as shown in Fig. 9. A Wilcoxon Signed-rank test shows that there is a significant difference on the decision time with Sponza cathedral while walking and while seated. ($Z = -2.00$, $p = 0.043$). It appears that the different LODs were easier to distinguish with individual models (Torus knot and Stanford bunny) as there were no statistically significant differences in decision time while walking or while seated.

As shown in Fig. 9, the number of incorrect answers was very low and nearly identical with Torus knot and Stanford bunny when comparing different LODs. For these models, there were no statistically significant diffences in the success rate while walking and while seated. However, with Sponza cathedral, different LODs were clearly more difficult to distinguish from each other. There was clearly a higher number of incorrect answers for Sponza cathedral, but there were no statistically significant differences in the success rate while seated and while walking.

As stated earlier, it is clear that distinguishing different LODs with Sponza cathedral was difficult for the participants and only the most obvious comparisons were easier (for instance, comparing LOD1 to

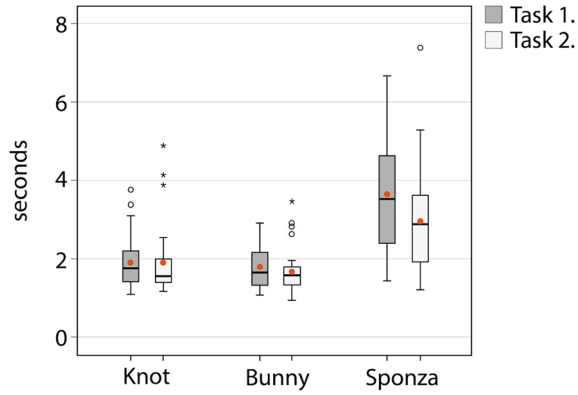


Fig. 8 Decision time in Tasks 1 and 2, *red dot* indicates the mean (n = 30). (Color figure online)

LOD6). In the case of Sponza cathedral, the success rate was very low when comparing through LODs from 1 to 4, as shown in Fig. 10. As previously stated, decision times were slightly faster while walking in the case of Sponza cathedral (see, Fig. 11). As expected, the decision times increased when the compared LODs were closer to each other.

4.1.2 Results of Task 3

As shown in Fig. 12, the participants decided lower acceptable LOD for Sponza cathedral (mean 3.80, SD 1.30) compared to Stanford bunny (mean 4.53, SD 1.04) and Torus knot (mean 4.97, SD 4.53). Both the decided lowest acceptable LODs and the selection times were analyzed using a Wilcoxon Signed-Rank test with a Bonferroni correction, resulting in a significance level set at $p < 0.017$. A Wilcoxon

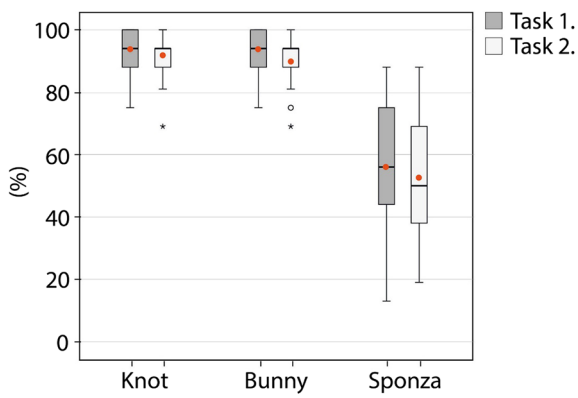


Fig. 9 Success rate of the comparisons in Tasks 1 and 2, *red dot* indicates the mean (n = 30). (Color figure online)

Signed-rank test shows that there is a significant difference between Torus knot and Sponza cathedral ($Z = -3.99$, $p < 0.001$), as well as Stanford bunny and Sponza cathedral ($Z = -2.91$, $p = 0.003$). There was no significant difference between Torus knot and Stanford bunny ($Z = -2.39$, $p = 0.023$). After the participants selected the lowest acceptable LOD, they were asked to comment and explain why they chose the particular LOD. Third of the participants (#1, #4, #5, #6, #11, #14, #16, #18, #24, #25) said that they chose the particular LOD based on the recognisability of the 3D object. In the case of Stanford bunny and Sponza cathedral, three of those participants (#1, #4, #24) mentioned Stanford bunny being recognizable with the lower LODs: *With bunny model, lower quality was enough, because the model stayed recognizable and the model could have been presented as pixel art* (*#24*). *The first picture was the most challenging, because the picture was abstract, but then the bunny and building were more expressive* (*#6*). Four participants (#1, #23, #28, #30) based their decision on the resolution, accuracy or sharpness of the 3D object. In this case, Torus knot was mentioned by one of the participant: *Texture is important, therefore, the highest detail* (*#1*). Torus knot was an abstract model and had surface texture so this might explain why Torus knot received the highest acceptable LOD compared to the other two models. In addition, it should be noted that Torus knot was an isolated 3D object on a black background. Sponza cathedral was also textured, but received the lowest acceptable LOD, which indicates the importance of the context.

Three participants (#13, #19, #24) stated that the chosen lowest acceptable LOD did not bother the eye: *Adequate level doesn't bother the eye* (*#24*) and two participants (#10, #21) mentioned that the chosen LOD pleased the eye. This notion of pleasing the eye can be linked to aesthetics and three participants (#2, #10, #20) stated that they based their decision on the aesthetics: *…the roughness of the surface texture can be viewed as aesthetic* (*#10*)*., Lower quality would've been clear, but not visually appealing* (*#20*). So aesthetics plays an important role in the user perceptions, as discovered also in previous studies [47]. Two participants (#4, #11) stated that the past experiences played a role in the decision making, but there were no statistically significant differences when comparing participants with more experience in 3D graphics or gaming with the participants that had limited experiences in 3D graphics or gaming. Three of the participants (#8, #15, #22) stated that they prefer fast and smooth performance rather than high visual quality: *I'm ready to lower the aesthetic factors for the sake of smoothness* (*#15*). One participant (#21) mentioned that the realism was an important factor on the decision. Two participants mentioned outlines as a reason for the selection (#7, #30): *Shape and outlines were distinguishable in acceptable level* (*#7*).

4.2 Energy Consumption in 3D Rendering

In this section, we present the results related to the energy consumption in 3D rendering. The total energy consumption was recorded for each experiment together with the maximum frame rate the device was capable to achieve. With these measurements, the per-frame energy consumption was calculated together with the normalized energy consumption per single display pixel.

4.2.1 Experiments: Stanford Bunny

The 3D model of Stanford bunny stresses especially the geometry part of the mobile GPUs, since it is a relatively complex 3D model as such, and does not use any additional textured surface materials. From the

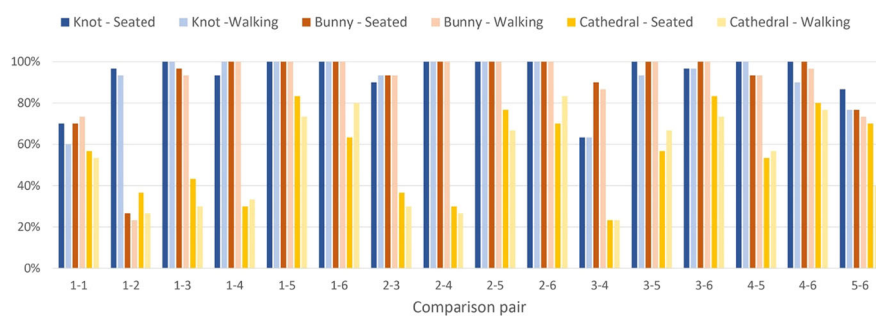


Fig. 10 Success rate per comparison pair (n = 30)

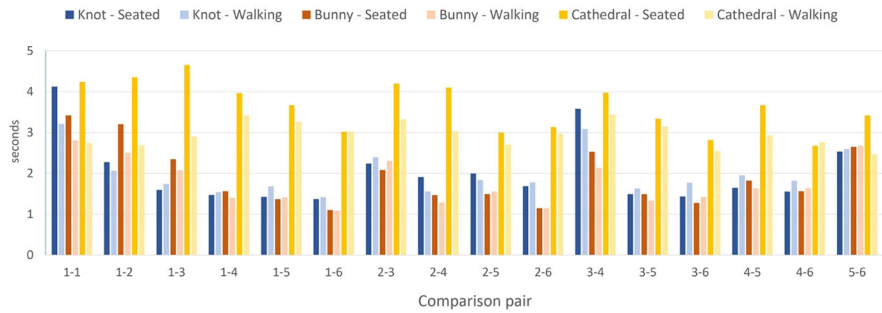


Fig. 11 The average decision time per comparison (n = 30)

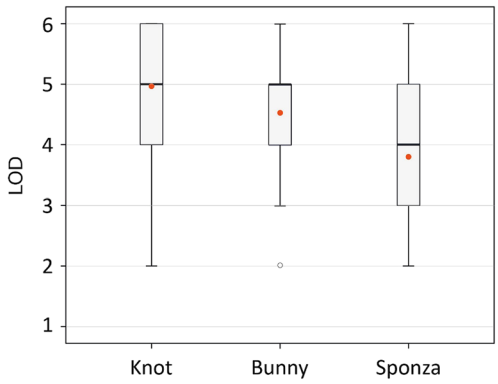


Fig. 12 The lowest acceptable LOD for each 3D object, *red dot* indicates the mean. (Color figure online)

mobile GPU architecture point of view, this means that Stanford bunny stresses the vertex shading, but less the fragment shading. When the number of 3D test objects is increased in the experiments, the number of triangles and vertices increases as well, rising up to the maximum of over 3 M vertices and 1 M triangles.

S2+ represents the low-end smartphone of the device triplet. The experiments scale the amount of geometry to quite high volumes, so it was expected that S2+ may not necessarily perform well with the highest rendering loads. Based on this assumption, it was surprising to see that S2+ performs more energy efficiently than S3, as shown in Fig. 13. Using the most complex 3D scene of 64 objects, it can be seen in Fig. 14 how the achieved average frame rate of S3 drops significantly compared to S2+. At the same time, the total energy consumed by S3 raises also approximately 10%, which further results in tremendous increase in the per-frame energy consumption compared to S2+ (see, Fig. 15). This is likely to be caused by the phenomenon of saturating the device capabilities, due to how much it can deliver real-time graphics rendering under the given circumstances. Because the main differentiating factor between the experiments is the number of vertices and triangles in the 3D scene, it is susceptible that the vertex processing stage of S2+ is working more efficiently compared to S3. The cause is likely related to the fact how the GPU processors of these devices work. In S2+, there are 4 common processors which can process either vertices or fragments, while in S3, there is one dedicated vertex unit and four fragment units. Presumably this allows S2+ to scale better to the heavy geometry workloads. However, S3 has a higher resolution display which increases the fragment processing workload which in turn might have an impact on the result. This effect presumably exists, but is not a dominating factor in this case as Stanford Bunny is especially complex in terms of geometry, and very simple in terms of surface details (lack of textures, per-vertex lighting) which means a simple fragment processing program to be executed. With the 3D scene of 64 objects, S5 also shows significant increase in the total energy consumption (see, Fig. 13). Despite this, S5 is able to maintain a steady 60 FPS for the duration of the whole experiment, as illustrated in Fig. 14.

In the experiments, a large number of objects are visible at the same time and the majority of them are potentially far away from the camera. Therefore, high detail LODs are needlessly rendered in the case of L3 method, which results in higher total energy consumption, particularly for S5 (see Fig. 13). Alternatively, clear benefits are gained using L1, L2 and L4 methods. Especially with L2 and L4, more consistent per-frame energy consumption is achieved for S2+ and S5, as shown in Fig. 15. L1, L2 and L4 open an opportunity to relatively large energy savings when compared to L3, which is especially emphasized with S2+ (see, Fig. 16). According to Fig. 14, S2+ can

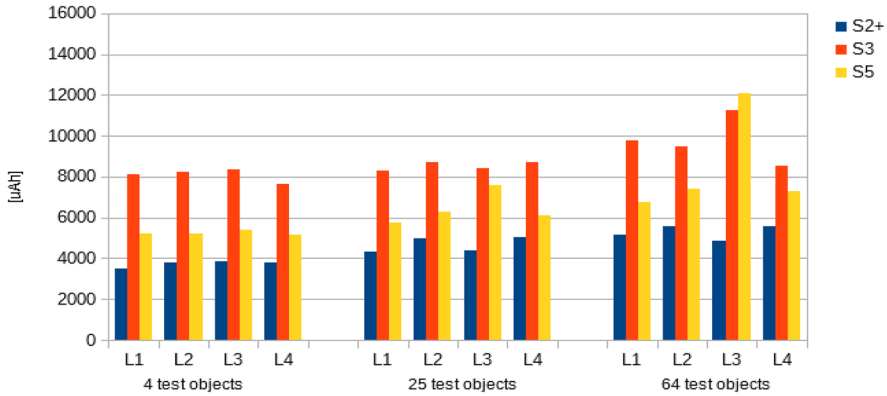


Fig. 13 The total energy consumed by each test device using Stanford bunny

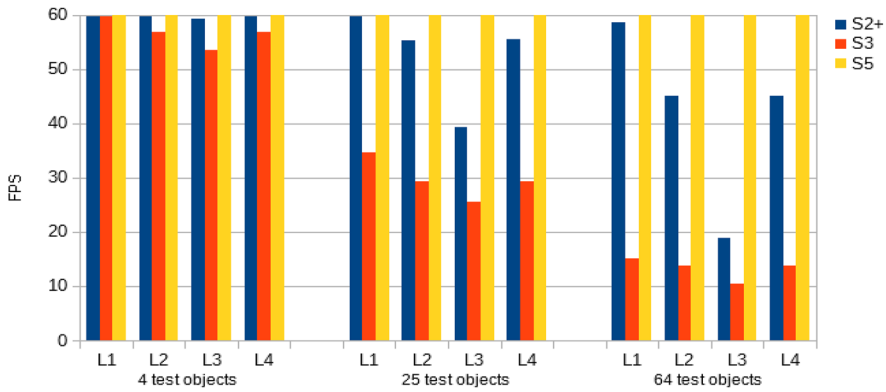


Fig. 14 Average FPS for each test device using Stanford bunny

gain at best over 100% better FPS with L2 and L4 compared to L3.

In general, the use of L2 and L4 provide the most substantial energy savings in comparison to L3, while making sure the quality of the 3D objects is not degraded too much, as happens with L1. When comparing the per-frame energy consumption between L2 and L4 using 64 test objects, it can be seen that with S2+ there is no difference, whereas S3 and S5 gain 3 and 1%, respectively, in favor of L4. The gain is not as much as when comparing with L3 directly, but it shows that stopping the loading to the lowest acceptable LOD has a measurable influence. As S3 struggles the most with the geometry processing in this experiment, the benefits are expectedly the most significant for this mobile device.

It should be noted in general that when evaluating the energy savings, looking at the total energy consumption (see, Fig. 13) over the test period is not enough. The geometry simplification methods enable the mobile devices to perform better, which is also seen in the number of the achieved FPS (Fig. 14). When one computes the average amount of energy invested in rendering a single frame (Fig. 15), one can clearly see the difference how the simplification affects the energy consumption.

4.2.2 Experiments: Torus Knot

Torus knot is much simpler in terms of geometry when compared to Stanford bunny. However, surface material definition of Torus knot uses a number of textures to achieve additional effects, such as per-fragment normal mapping and parallax mapping. Hence, from the rendering perspective, the workload of vertex processing stage is much less compared to that of fragment stage, in which all the lighting and object surface effect computations are done in this case.

The total energy consumption in the rendering of Torus knot is in general higher than with Stanford bunny. In the GPU, the vertex processing stage is done for each vertex only (which reside sparsely in the 3D space), but the fragment processing is done for each fragment what needs to be shown on the display. Now,

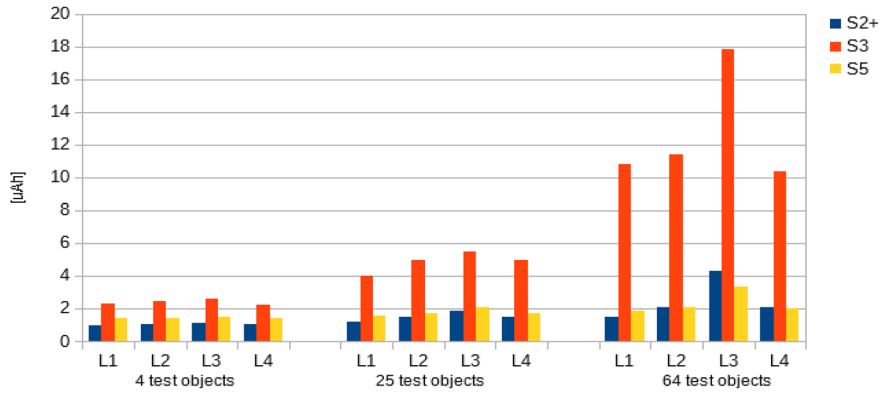


Fig. 15 Average energy consumption per frame for each test device with Stanford bunny

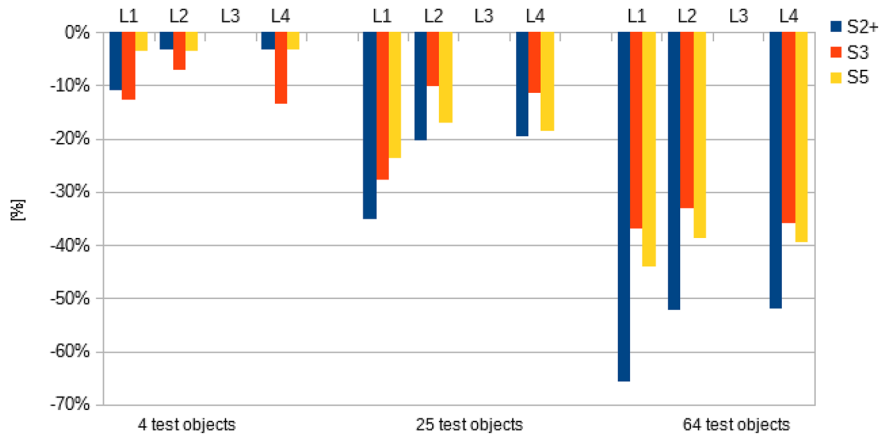


Fig. 16 Per frame energy savings compared to L3 (all max) method with Stanford bunny

when the complexity of the fragment processing increases, it also increases the workload for the GPU relatively fast, as can be seen in Fig. 17. At the same time, the achieved FPS is not as low as with Stanford bunny (see, Fig. 18). The fragment processing stage, though more complex, it is also highly parallelizable, as can be seen in Fig. 19.

When comparing to L3, L2 and L4 do not provide as large energy savings with Torus knot as with Stanford bunny. With Stanford bunny and 64 object test objects, the energy savings range from 32 to 52% averaging to 42% (Fig. 16), while with Torus knot and 64 object test objects, the energy savings are more uniform averaging to 20% (Fig. 20). The reason is that even though the geometry is simplified, the dominating factor remains the fragment processing stage and the simplification of the geometry does not reduce the number of fragments at the same proportion. Additionally, shifting the workload from the vertex processing stage to the fragment processing stage, sets S2+ and S3 in par in terms of FPS (Fig. 18), because the hardware constraints of S3 for vertex processing do not dominate anymore. The rendered 3D object still occupies roughly the same screen space, hence it requires roughly the same number of fragments to be evaluated, and in this case, the workload is proportional to the number of pixels on the display. The increased performance requirements can be seen in the results concerning S5. Although the geometry is simpler, even S5 is not able to sustain 60 FPS all the time when using 64 test objects and L3 (Fig. 18). When comparing L2 and L4 using 64 test objects, the results show that with S2+ there is no difference, whereas S3 and S5 gain 3 and 2%, respectively, in favor of L4.

4.2.3 Experiments: Sponza Cathedral

The Sponza cathedral scene is a 3D scene whose interiors consisted of a single 3D object, Sponza cathedral itself. In overall, Sponza cathedral is a relatively complex 3D object compared to Stanford

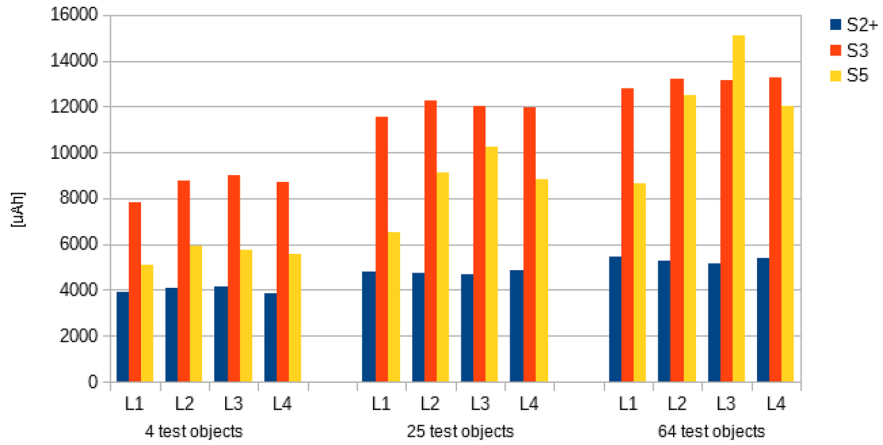


Fig. 17 The total energy consumed by each test device using Torus knot

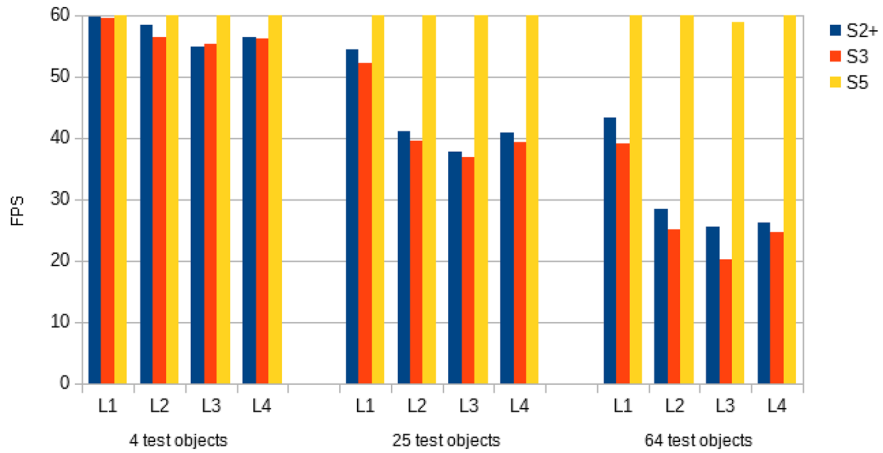


Fig. 18 Average FPS for each test device using Torus knot

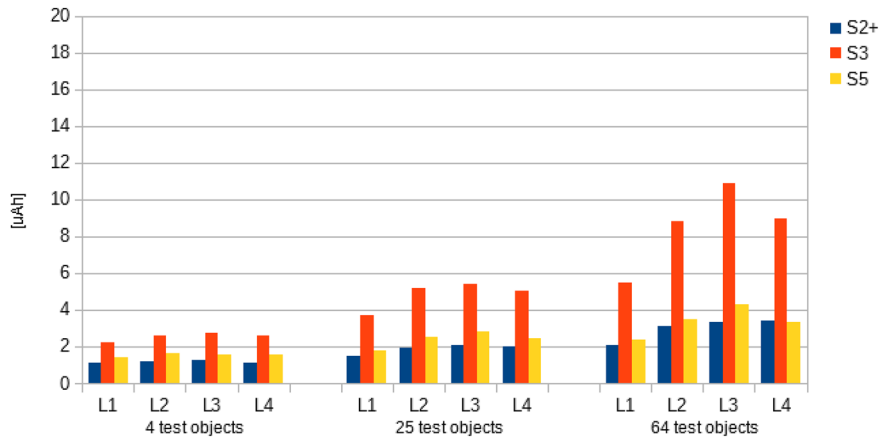


Fig. 19 Average energy consumption per frame for each test device with Torus knot

bunny and Torus knot. Like Torus knot, Sponza cathedral also has complex surface material definitions consisting of normal and parallax texture maps.

As earlier mentioned, the dynamic LOD mechanisms (L2 and L4) were not usable on Sponza cathedral directly, so the LOD switching was implemented artificially. While the camera turned over its

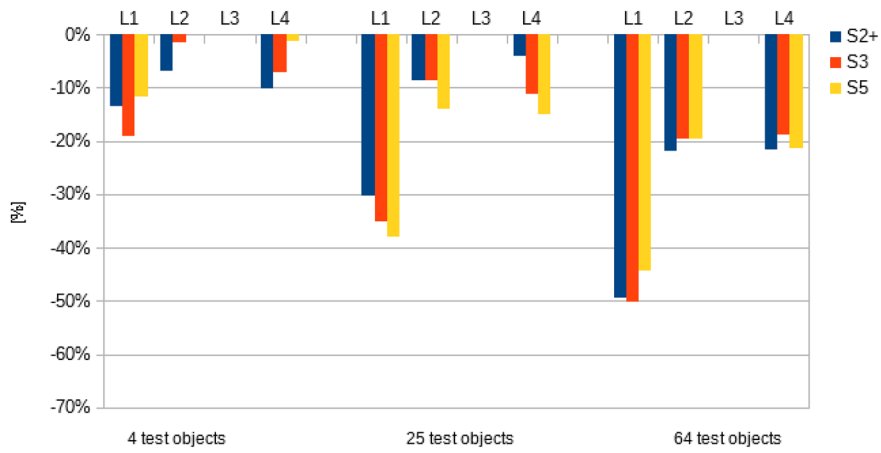


Fig. 20 Per frame energy savings compared to L3 (all max) method with Torus knot

axis to view all interior parts of Sponza cathedral, LODs were switched periodically with L2 and L4 in a way that each LOD was visible an equal amount of time during each experiment. As illustrated in Fig. 21, the total energy consumption was approximately equal for each test device across all LOD methods. The same applies for FPS as shown in Fig. 22 and the average energy consumption per frame, as shown in Fig. 23. When comparing L1 with L3 in terms of FPS, there is a slight increase of 9% for S2+ and S3 in favor of L1. For S5, there is no difference. The LOD method comparison in Fig. 24 shows that even when using only a single 3D object, L2 and L4 can provide energy savings. When comparing L2 and L4, the results show that with S2+ there is no difference, whereas both S3 and S5 gain 1% in favor of L4. In turn, when comparing L4 to L3, the gains for S2+, S3 and S5 are 1, 5 and 2%, respectively.

In overall, it can be stated that dynamic LOD methods with correctly split 3D objects can be useful also for the interior 3D scenes. Even with the lowest acceptable LOD (L4), the savings are measurable. This experiment is especially interesting since the users picked a relatively simple 3D model as the minimum viable choice. The chosen lowest acceptable LOD was much simpler than the choice with Stanford bunny or Torus knot, in terms of the number of vertices and triangles. This is an indicative result of, first of all, that the degradation in the visual quality of 3D graphics is not that obvious for the end-users with the interior type 3D models, and secondly, this gives room for the use of all kinds of geometry simplification methods (including POP buffers), which can reduce the energy consumption in the 3D rendering.

4.2.4 Energy Consumption in Each Frame per Pixel

Each of the test devices has been equipped with different displays with varying resolutions, starting from 800 × 480 pixels (S2+) up to 1920 × 1080 pixels (S5). The size of the display is proportional to the requirements the GPU needs to fulfill for each frame when the complete display is redrawn. During the rasterization phase, the color of each pixel needs to be recomputed, which takes place in the fragment program. Even when rendering the same workload, the test devices are executing a completely different number of fragment operations due to the difference in the size of their displays. Hence, the differences in the rendering performance may not be so obvious by merely evaluating the achieved FPS or the total energy consumption. As Akenine-Möller and Johnsson [1] propose, normalizing the total executed workload to pixel granularity may be a better overall view to the varying performance characteristic of the test devices.

Figures 25, 26 and 27 summarize the per-pixel energy consumptions for each experiment and for each test device. In this viewpoint, S3 is the most inefficient among the three, and its inefficiency is emphasized when more complex 3D scenes are being rendered. The inefficiency of S3 is emphasized most with Stanford bunny, where the vertex processing bottleneck is the most obvious, as already explained. Of the three test devices, S5 is clearly the most efficient one. When the display size is taken into account, the figures show how the per-pixel energy invested is only a fraction compared to S2+ and S3.

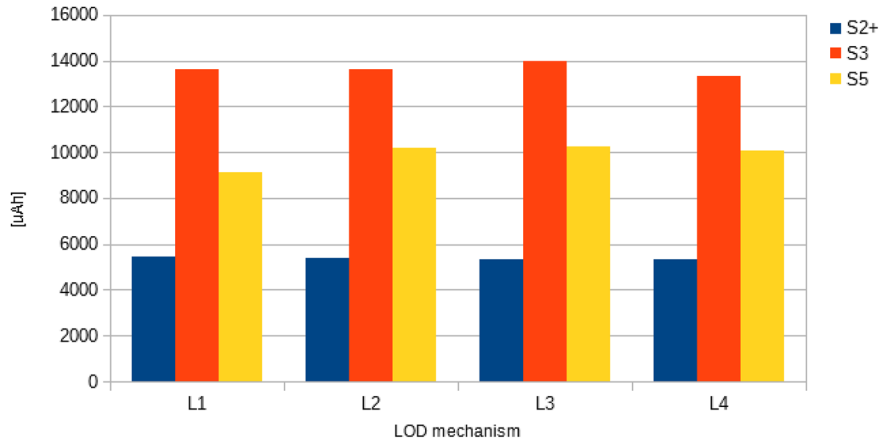


Fig. 21 The total energy consumed by each test device using Sponza cathedral

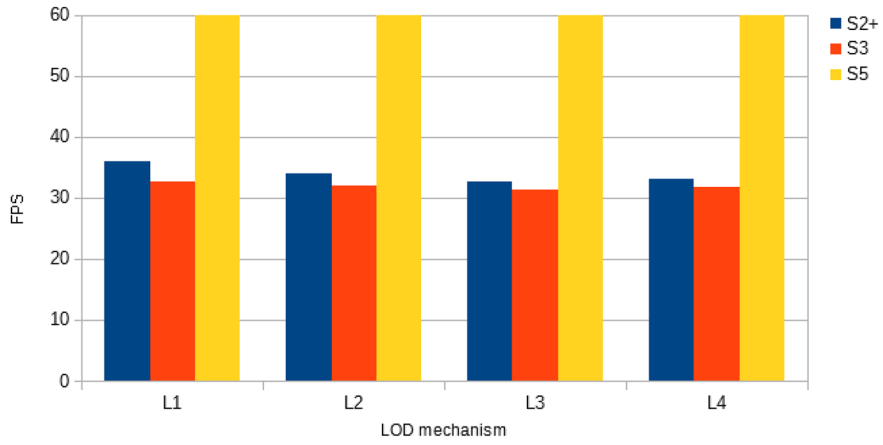


Fig. 22 Average FPS for each test device using Sponza cathedral

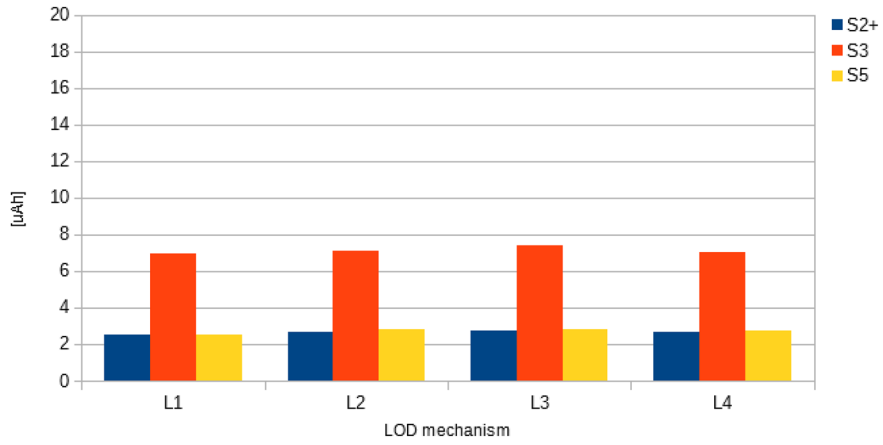


Fig. 23 Average energy consumption per frame for each test device with Sponza cathedral

4.2.5 Energy Consumption with FPS Limit

All of the 3D objects were rendered without any kind of FPS limit in effect. The only limit came from the operating system (eventually from the graphics driver) which limits the maximum FPS to 60 on the Android operating system (OS). It is important to understand that via simplification of the geometry (or any other 3D data type for that matter) it is possible to reduce the rendering workload of the mobile device, and as a

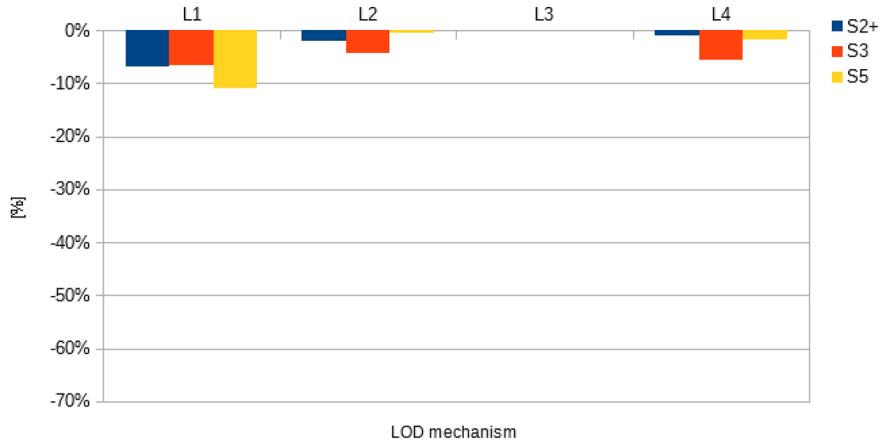


Fig. 24 Per frame energy savings compared to L3 (all max) method with Sponza cathedral

consequence, increase the achievable frame rates of the 3D application. If the maximum of 60 FPS is not needed by the 3D application, the achieved simplification results can in theory be turned into energy savings if the rendering frame rate is limited and the gained time is used to put the hardware into sleep states. However, proper activation of the hardware specific sleep states may be impossible to execute by the 3D application, since those features are typically in the control of the OS.

During this experiment, we examined how limiting the rendering speed to 10 FPS affects the energy consumption. Based on the earlier experiments, limiting the frame rate to 10 FPS should give room for all of the mobile devices to reach sleep states during the execution of each experiment. The CPU sleep was implemented after completion of each frame by using usleep() system call. The set limit of 10 FPS results in 100 ms frame rendering time. First, the frame was rendered as fast as possible, and the remaining time was slept using the usleep() system call to make the duration of single frame to match 100 ms, hence resulting in 10 FPS.

The effect of the 10 FPS limit was examined with all three test devices using Stanford bunny with 25 objects and L3 ("all max"). This particular experiment was chosen because it delivered a considerable amount of workload on all test devices and each device achieved clearly higher FPS than 10 on average. This means that using the 10 FPS limit would potentially result in energy savings. Also, the 10 FPS limit was considered to be fast enough to still create a fluent animation through the experiment.

The results of the experiment are shown in Figs. 28, 29, 30, 31, 32 and 33. There are two phenomena to pay attention to in the figures. Firstly, when the usleep() is invoked, the graph of the consumed current starts oscillating, while the energy consumption without the

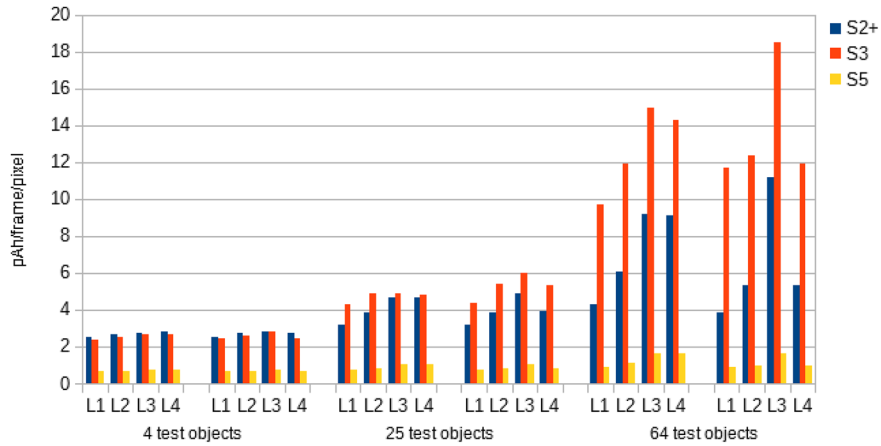


Fig. 25 Energy consumed in each frame per pixel with Stanford bunny

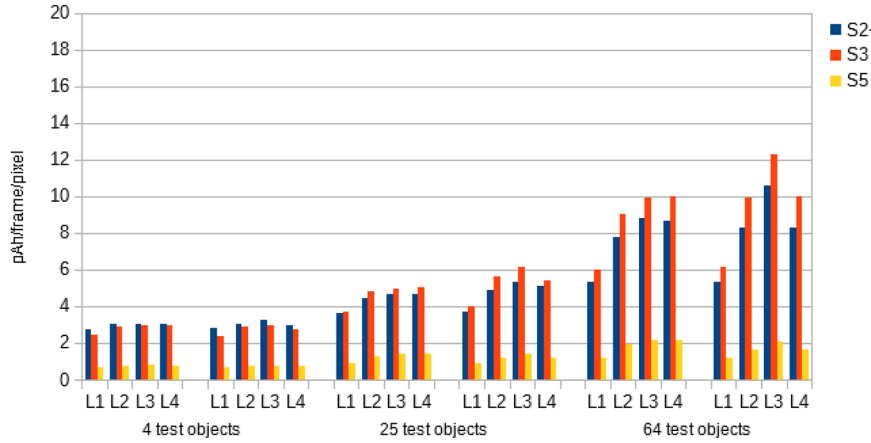


Fig. 26 Energy consumed in each frame per pixel with Torus knot

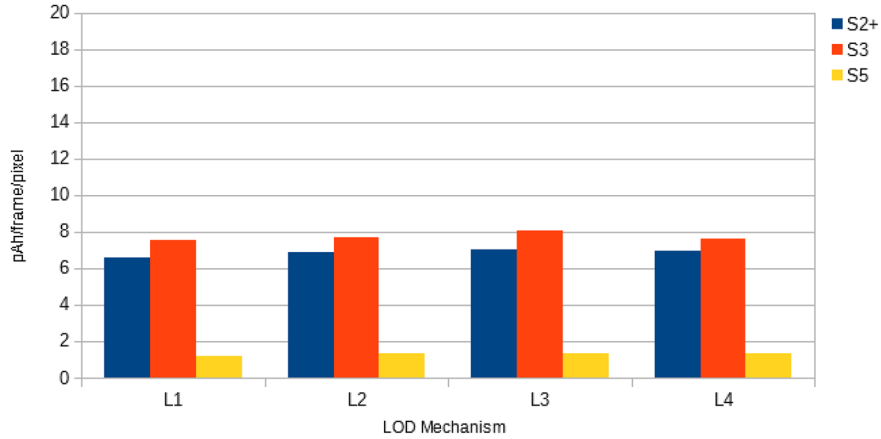


Fig. 27 Energy consumed in each frame per pixel with Sponza cathedral

FPS limit is much more stable. The hypothesis for this is that the OS invokes power saving schemes which are not visible directly for the 3D application. Possibly, the OS turns off and on parts of the hardware, which eventually can be measured as an oscillating consumption curve. The verification of the exact phenomenon would require more specific tools developed particularly for the test devices. Secondly, when the FPS limit is applied, it in general lowers the total energy consumption. However, with S2+, the oscillation is for some reason happening so violently that the cumulative energy consumption with the FPS limit is actually higher than without it. With the FPS limit, S3 and S5 consume significantly less energy. Based on the results, it can be seen that the more the specific mobile device has processing capabilities, the more it gains from the application level sleep invocation. By comparing the total energy consumption with and without the FPS limit, the effect for the mobile devices used in the experiments was as follows: S2+: +13.31%, S3: −15.65% and S5: −43.91%.

Lowering the maximum FPS has also an impact on the rendered 3D graphics and how the 3D application behaves. This would have an impact on the user experience as well, but it was not examined within the scope of this paper. It should be pointed out that these experiments give an insight of the proportion of the energy savings that can be achieved using application level sleep invocation. In the future, the FPS limit could potentially be adaptively adjusted according to the 3D application usage scenarios. However, this would require an implementation of such adaptive FPS limiter and comprehensive user studies of how the limitations translate into user experience and how they compare with the achieved energy savings.

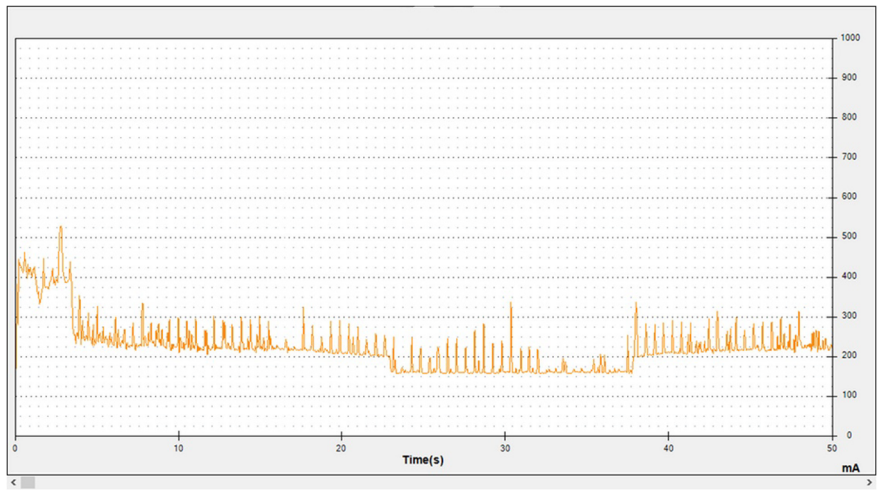


Fig. 28 Energy consumption graph with the maximum rendering speed (S2+: L4 with 25 objects)

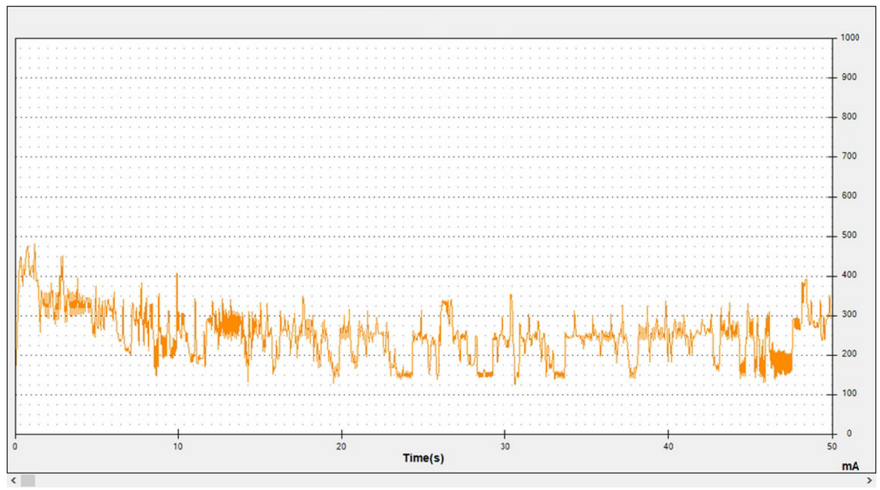


Fig. 29 Energy consumption graph with the FPS limit of 10 (S2+: L4 with 25 objects)

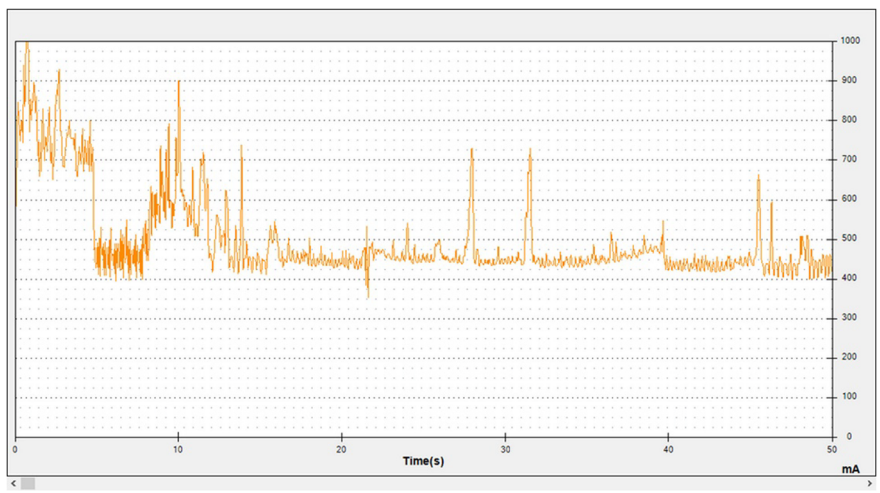


Fig. 30 Energy consumption graph with the maximum rendering speed (S3: L4 with 25 objects)

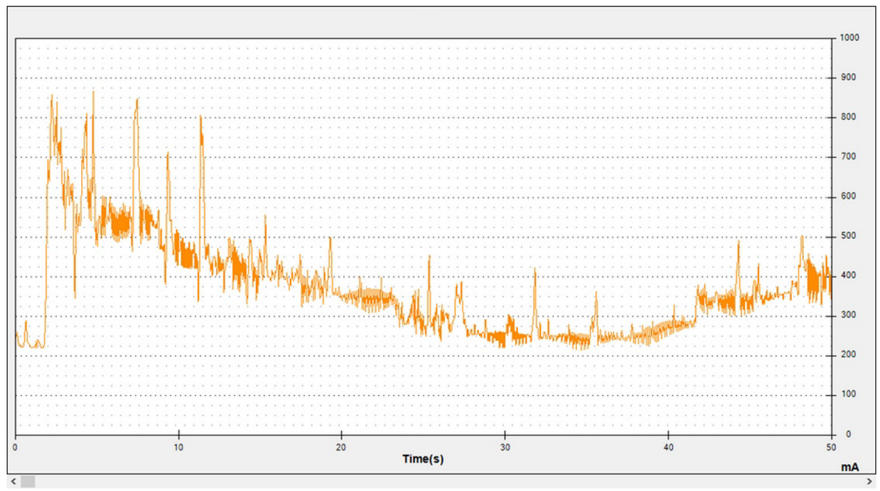


Fig. 31 Energy consumption graph with the FPS limit of 10 (S3: L4 with 25 objects)

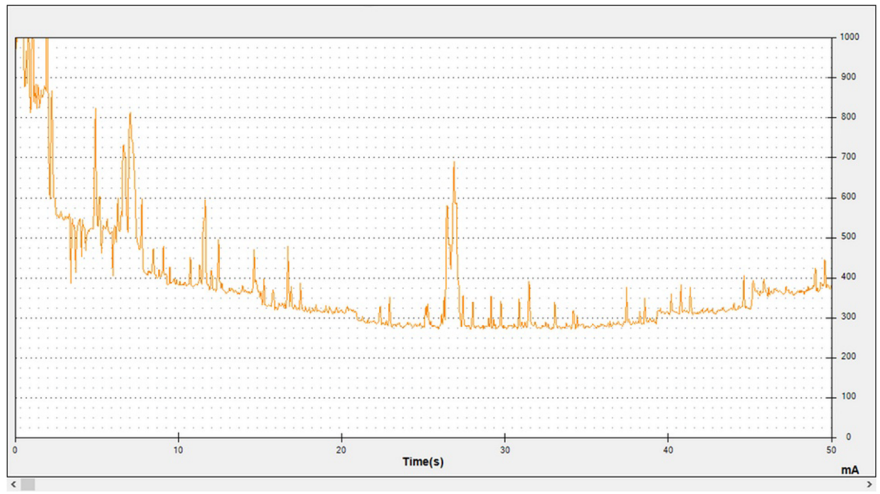


Fig. 32 Energy consumption graph with the maximum rendering speed (S5: L4 with 25 objects)

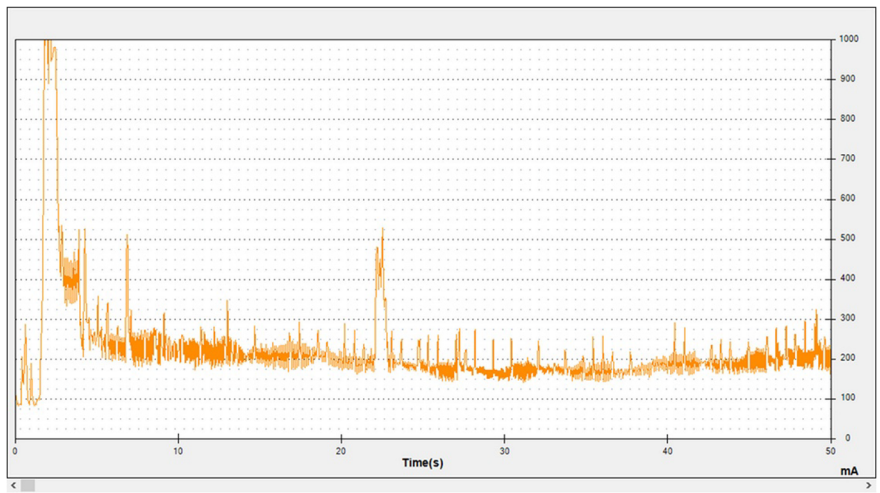


Fig. 33 Energy consumption graph with the FPS limit of 10 (S5: L4 with 25 objects)

4.3 Energy Consumption in 3D Graphics Delivery

According to Xiao et al. [51], the use of the wireless radio interfaces roughly accounts for one-third of the total energy consumption of a mobile device. In this section, we estimate the energy consumption when delivering the three test objects in mobile operator networks.

It was clearly shown in Sect. 4.2 that the 3D rendering performance of smartphones has increased rapidly during the recent years. At the same time, the battery capacity has increased at a much slower pace (4% annually [5]), which emphasizes the importance of energy saving in 3D graphics deployment on mobile devices. Avoiding the transmission of high-quality LODs whose additional details the user does not distinguish in the small screen of a mobile device, saves energy not only in the 3D graphics rendering, but also in the 3D graphics delivery.

The total energy consumption of 3D graphics delivery in mobile operator networks is a sum of promotion, data transfer, tail and idle energy [4, 17]. Promotion stands for the energy required in promoting the radio from idle to high power mode. Tail stands for the energy used after a completed data transfer when the radio still lingers in high power modes before going idle. Idle stands for the energy used when the radio is turned off.

The energy consumption in the 3D graphics delivery was estimated for each test object based on the download speed [18] and power consumption [17] measurements conducted in real-life 3G (UMTS) and 4G (LTE) networks. In Table 5, the average download speed of 3G and 4G are shown with the corresponding power consumption rate.

Based on Table 5, Tables 6, 7, and 8 present the estimates for the average transfer time and the energy consumption when downloading the test objects over mobile operator networks. For the consumed energy, two estimates are provided: (1) data transfer energy only; (2) the sum of promotion, data transfer and tail energy. This is to facilitate the comparison between 3G and 4G radios as well as between different test objects. The effect of promotion and idle energy [17] is marginal, but the tail energy can have a significant effect on the overall energy consumption. However, tail energy may vary substantially based on the application traffic patterns [4]. Based on the measurements conducted by Huang et al. [17], the promotion and tail energy are for 3G and 4G at the maximum approximate 550 and 960 μAh respectively. The values were retrieved by promoting the radio from idle into high power state and by waiting for the radio to return to the idle state after completing the transmission of a single packet. Finally, it should be noted that the estimates do not include the energy consumed by the uplink as it is only used for transmitting requests for the 3D objects. Therefore, the effect of the uplink usage is only marginal on the total energy consumption. Asterix (*) denotes the lowest acceptable LOD based on the user study.

From the viewpoint of energy consumption, downloading a single test object even with the highest LOD (see, Tables 6, 7, 8) is less significant than rendering the same object for 65 s as done in the 3D rendering test cases in Sect. 4.2 (see, Figs. 13, 17, 21). This is particularly visible with 4G that has a high average throughput. However, with the increasing number of LODs, the transfer time starts to increase significantly with 3G, as seen with Stanford bunny in Table 6 and particularly with Sponza cathedral in Table 8. The use of the lowest acceptable LOD instead of the maximum LOD results in 77% shorter transfer time and 48% lesser energy consumption (promotion and tail included) with Stanford Bunny, and in 43 and 28% with Sponza Cathedral, respectively. In real-life applications, it is typical that multiple objects are successively downloaded, and therefore, the energy savings are expected to be even higher as the energy consumed mainly results from the actual data transfer rather than tail.

It should also be noted that if the 3G radio is not already in a high-power state, the promotion to high power state and the overall RTT (round trip time) may also delay the start of the download by additional 1–2 s. Therefore, in the worst case, the user might be forced to wait for more than 7 s (see, Table 8) before the first LOD of the Sponza cathedral could be rendered on the screen. This emphasizes the need for considered creation of LODs based on the expected capacities of end-user devices. With 4G, the delay caused by promotion and RTT are however less significant [17]. Finally, it should be noted that additional delay can also be caused by TCP slow start, which typically prevents the maximized use of available bandwidth in the beginning of the 3D graphics download.

Table 5 Average download speed and power consumption of 3G (UMTS) and 4G (LTE) radios

	Average download speed	Power consumption (downlink)
3G (UMTS)	763 kbps	900 mW
4G (LTE)	9185 kbps	1800 mW

Table 6 Stanford bunny

LOD		1	2	3	4	5*	6
Object data [B] (cumulative)		28656	64566	162858	262104	293262	1271478
3G (UMTS)	Transfer time [s] (cumulative)	0.30	0.68	1.71	2.75	3.07	13.3
	Consumed energy [μAh]	23	46	116	186	207	899
	(cumulative)	576	596	666	736	757	1449
4G (LTE)	Transfer time [s] (cumulative)	0.02	0.06	0.14	0.23	0.26	1.1
	Consumed energy [μAh]	3	8	19	31	35	149
	(cumulative)	963	968	979	991	995	1109

Table 7 Torus knot

LOD		1	2	3	4	5*	6
Object data [B] (cumulative)		9324	41312	61316	63228	63360	75968
3G (UMTS)	Transfer time [s] (cumulative)	0.10	0.43	0.64	0.66	0.66	0.80
	Consumed energy [μAh]	7	29	44	45	45	54
	(cumulative)	557	579	594	595	595	604
4G (LTE)	Transfer time [s] (cumulative)	0.01	0.04	0.05	0.06	0.06	0.07
	Consumed energy [μAh]	2	5	7	8	8	10
	(cumulative)	962	965	967	968	968	970

Table 8 Sponza Cathedral

LOD		1	2	3	4*	5	6
Object data [B] (cumulative)		508448	695262	1101346	1351074	1471658	2362796
3G (UMTS)	Transfer time [s] (cumulative)	5.33	7.29	11.55	14.17	15.43	24.77
	Consumed energy [μAh]	360	493	781	957	1043	1673
	(cumulative)	910	1043	1331	1507	1593	2223
4G (LTE)	Transfer time [s] (cumulative)	0.44	0.61	0.96	1.18	1.28	2.06
	Consumed energy [μAh]	59	83	130	159	173	279
	(cumulative)	1019	1043	1090	1119	1133	1239

5 Discussion and Conclusions

In this paper, we explored the trade-off between user perceptions of visual quality and the gained energy savings in the simplification of 3D graphics. Our focus was particularly on simplification of geometry, since the processing of the geometry has been discovered to be the most energy consuming part in the rendering [29], and furthermore, the artifacts in the simplified geometry can be potentially hidden with high quality textures [6].

Progressive delivery of 3D graphics allows the 3D application to quickly access the first LODs, and hence, to minimize the time the user has to wait before anything meaningless is rendered in the 3D application. Before the introduction of modern geometry simplification methods (e.g. Limper et al. [28]), the use of progressive LODs has not been feasible for mobile devices due to their very high processing requirements. It should be emphasized that the availability of LODs also enables distance-based adjustment of detail for all 3D objects in a 3D scene, which greatly reduces energy consumption in the rendering.

In a typical process of authoring 3D graphics, the artist first creates the highest LOD, which can be further used for creating a required number of lower quality variants using an automated process. Based on our study, users are ready to accept lower quality LODs, and in some cases, they may not be capable of recognizing the difference between two adjacent LODs even though the geometry has been heavily simplified. How much the geometry can be simplified before it is noticed by the users is, however, highly dependent on the content itself, the user, and the usage context. Our study was conducted in a laboratory environment, in which the users primarily focused on finding the flaws in the simplified 3D objects. However, in a real-life use of 3D applications, this kind of use behavior is not typical as also stated by Watson et al. [48]. Furthermore, the 3D objects were evaluated against black background using optimal lighting conditions. These two design choices were deliberately taken to emphasize the artifacts particularly in the edges of the 3D objects. Therefore, in real-life mobile 3D applications, the visual quality of 3D graphics could presumably be decreased even further for the sake of energy savings without distracting the user.

In our study, we also noticed that the used interior model (i.e. Sponza Cathedral) could be simplified more heavily compared to individual exterior models (i.e. Torus knot and Stanford bunny). This emphasizes the importance of context in which the 3D objects reside. According to Koulieris et al. [23], users have higher tendency to recognize the lack of detail in individual and isolated 3D objects than in groups of 3D objects. This tendency also seems to apply to large interior models, where the users are not so focused on any small details, but rather on the whole 3D scene itself.

Another interesting finding in our study is related to users on the move. It seems that when a user is walking and simultaneously using the mobile device, the perception on the visual quality of 3D graphics declines. This same phenomenon was also discovered by Rogowitz and Rushmeier [39], however, from the opposite angle. In their study, Rogowitz and Rushmeier [39] concluded that the perceived quality of a 3D object is significantly higher when the 3D object is moving and is lit from the front. They stated that this is probably due to the fact that the human visual system is less sensitive to high spatial frequencies when the 3D object is on the move. Using movement detection (e.g. accelerometer of the mobile device) might allow temporary use of lower quality LODs without a user noticing the change. However, this would require further user studies and experiments regarding the achievable energy savings.

In our experiments, we also investigated whether further energy savings could be achieved by forcefully limiting the FPS of the 3D application. The results of our experiment indicate that by using the limit of 10 FPS, significant energy savings can be gained (in our example case up to 44%). It should also be noted that the energy consumption curves started to oscillate when the FPS was limited to 10. This behavior was presumably caused by either the OS or the device drivers, but it was something that could not be controlled by the 3D application. This finding opens up room for further studies on how well an intelligent and adaptive frame rate control would work in 3D rendering context. The frame rate could be intelligently adapted, for instance, in cases (1) when there is a low number of changes detected in the view of the 3D application, and/or (2) when the user is on the move and less sensitive to fine details of the 3D application.

Based on our experiments in overall, it can be concluded that geometry simplification can really provide significant energy savings for mobile devices without disturbing the users. When geometry simplification is combined with dynamic adjustment of LODs based on the viewing distance, up to 52% energy savings were gained in our experiments compared to using only a single high quality 3D model. Furthermore, by avoiding the use of high-quality LODs whose additional details the user may not distinguish in the small screen of a mobile device, saves energy not only in the 3D graphics rendering, but also in the 3D graphics delivery.

Acknowledgements This work has been supported by the CADIST3D project (268905) funded by the Academy of Finland. In addition, the Strategic Research Council at the Academy of Finland is acknowledged for financial support of the COMBAT project (293389).

References

1. Akenine-Möller, T., & Johnsson, B. (2012). Performance per what? *Journal of Computer Graphics Techniques, 1,* 37–41.
2. Assarsson, U., & Moller, T. (2000). Optimized view frustum culling algorithms for bounding boxes. *Journal of Graphics Tools, 5,* 9–22. doi:10.1080/10867651.2000.10487517.
3. Atlan, S., & Garland, M. (2006). Interactive multiresolution editing and display of large terrains. *Computer Graphics Forum, 25*(2), 211–223. doi:10.1111/j.1467-8659.2006.00936.x.
4. Balasubramanian, N., Balasubramanian, A., & Venkataramani, A. (2009). Energy consumption in mobile phones In *Proceedings of the 9th ACM SIGCOMM conference on internet measurement conference—IMC'09* (p. 14). New York: ACM. doi:10.1145/1644893.1644927.
5. Belleville, M., Cantatore, E., Fanet, H., Fiorini, P., Nicole, P., Pelgrom, M., et al. (2009). *Energy autonomous systems: Future trends in devices, technology, and systems*. Paris: Catrene.
6. Bulbul, A., Capin, T., Lavoue, G., & Preda, M. (2011). Assessing visual quality of 3-D polygonal models. *Signal Processing Magazine, 28,* 80–90. doi:10.1109/MSP.2011.942466.
7. Chang, R., Butkiewicz, T., Ziemkiewicz, C., Wartell, Z., Pollard, N., & Ribarsky, W. (2008). Leglible simplification of textured urban models. *IEEE Computer Graphics and Applicatons*. doi:10.1109/MCG.2008.56.
8. Dmitriev, K., & Makarov, E. (2011). Generating Displacement from Normal Map for use in 3D Games In *Proceedings of ACM SIGGRAPH 2011 Talks*. New York: ACM, Vancouver.
9. Gil, B., & Trezentos, P. (2011). Impacts of data interchange formats on energy consumption and performance in smart phones, In *Workshop on open source and design of communication—OSDOC'11* (pp. 1–6). New York: ACM.
10. GSMARENA. (2014). Samsung S5 [WWW Document]. http://www.gsmarena.com/samsung_galaxy_s5-6033.php.
11. GSMARENA. (2013). Samsung S2+ [WWW Document]. http://www.gsmarena.com/samsung_i9105_galaxy_s_ii_plus-5213.php.
12. GSMARENA. (2012). Samsung S3 [WWW Document]. http://www.gsmarena.com/samsung_i9300_galaxy_s_iii-4238.php.
13. GSMARENA. (2011). Samsung S2 [WWW Document]. http://www.gsmarena.com/samsung_i9100_galaxy_s_ii-3621.php.
14. Heikkinen, M.V.J., & Nurminen, J.K. (2010). Consumer attitudes towards energy consumption of mobile phones and services, In *2010 IEEE 72nd vehicular technology conference—Fall* (pp. 1–5). Piscataway, NJ: IEEE. doi:10.1109/VETECF.2010.5594115.
15. Hoppe, H. (1996). Progressive meshes, In *Proceedings of the 23rd annual conference on computer graphics and interactive techniques—SIGGRAPH'96* (pp. 99–108). New York: ACM.
16. Hosseini, M., Fedorova, A., Peters, J., & Shirmohammadi, S. (2012). Energy-aware adaptations in mobile 3D graphics, In *20th ACM international conference on multimedia* (pp. 1017–1020). Nara.
17. Huang, J., Quian, F., Gerber, A., Mao, Z. M., Sen, S., & Spatscheck, O. (2012). A close examination of performance and power characteristics of 4G LTE networks. *MobiSys*. doi:10.1145/2307636.2307658.
18. Huang, J., Quian, F., Guo, Y., Zhou, Y., Xu, Q., Mao, Z.M., Sen, S., & Spatscheck, O. (2013). An in-depth study of LTE: Effect of network protocol and application behavior on performance, In *SIGCOMM* (pp. 363–374).
19. Ickin, S., Wac, K., Fiedler, M., Janowski, L., Hong, J.-H., & Dey, A. K. (2012). Factors influencing quality of experience of commonly used mobile applications. *IEEE Communications Magazine, 50,* 48–56. doi:10.1109/MCOM.2012.6178833.
20. Johnsson, B., & Akenine-Möller, T. (2015). A performance and energy evaluation of many-light rendering algorithms. *The Visual Computer, 31,* 1671–1681.
21. Johnsson, B., & Akenine-Möller, T. (2014). Measuring per-frame energy consumption of real-time graphics applications. *Journal of Computer Graphics Techniques, 3,* 60–73.
22. Koskela, T., & Vatjus-Anttila, J. (2015). Optimization techniques for 3D graphics deployment on mobile devices. 3D Research 6. doi:10.1007/s13319-015-0040-0.
23. Koulieris, G.A., Drettakis, G., Cunningham, D., & Mania, K. (2014). C-LOD: Context-aware material level-of-detail applied to mobile graphics. Computer Graphics Forum 33, No. 4.
24. Kularatna, N. (2011). Rechargeable batteries and their management. *IEEE Instrumentation and Measurement Magazine, 14,* 20–33. doi:10.1109/MIM.2011.5735252.
25. Lavoué, G., Chevalier, L., & Dupont, F. (2013). Streaming compressed 3D data on the web using JavaScript and WebGL, In: *18th international conference on 3D web technology* (pp. 19–27) San Sebastian.
26. Lavoue, G., & Corsini, M. (2010). A comparison of perceptually-based metrics for objective evaluation of

geometry processing. *IEEE Transactions on Multimedia, 12,* 636–649. doi:10.1109/TMM.2010.2060475.
27. Lefohn, A. E., Sengupta, S., & Owens, J. D. (2007). *Resolution-matched shadow maps*. Graph: ACM Transactions. **26**.
28. Limper, M., Jung, Y., Behr, J., & Alexa, M. (2013). The POP buffer: Rapid progressive clustering by geometry quantization. *Computer Graphics Forum, 21,* 197–206.
29. Ma, X., Deng, Z., Dong, M., & Zhong, L. (2013). Characterizing the performance and power consumption of 3D mobile games. *Computer, 46,* 76–82.
30. McGuire. (2011). Sponza cathedral [WWW Document]. URL http://graphics.cs.williams.edu/data/meshes.xml.
31. Millan, E., & Rudomin, I. (2006). Impostors and pseudo-instancing for GPU crowd rendering, In *The 4th international conference on computer graphics and interactive techniques in Australasia and Southeast Asia* (pp. 49–55).
32. Monsoon, (2015). Monsoon power monitor [WWW Document]. URL https://www.msoon.com/LabEquipment/PowerMonitor/.
33. Ogre3D, (2015). Torus Knot.
34. Pan, Y., Cheng, I., & Basu, A. (2005). Quality metric for approximating subjective evaluation of 3-D objects. *IEEE Transactions on Multimedia, 7,* 269–279. doi:10.1109/TMM.2005.843364.
35. Peng, J., Kim, C. S., & Kuo, C. C. J. (2005). Technologies for 3D mesh compression: A survey. *Journal of Visual Communication and Image Representation, 16,* 688–733.
36. Policarpo, F., & Oliveira, M.M. (2006). Relief mapping of non-height-field surface details. In symposium on interactive 3D graph games—SI3D'06 1, 55.
37. Puech, C., & Yahia, H. (1985). Quadtrees, octrees, hyper-octrees: A unified analytical approach to tree data structures used in graphics, geometric modeling and image processing In: *The first annual symposium on computational geometry* (pp. 272–280).
38. Rao, R., Vrudhula, S., & Rakhmatov, D. N. (2003). Battery modeling for energy-aware system design. *Computer, 36,* 77–87. doi:10.1109/MC.2003.1250886.
39. Rogowitz, B.E., Rushmeier, H.E., 2001. Are image quality metrics adequate to evaluate the quality of geometric objects? In B.E. Rogowitz & T.N. Pappas (Eds.), *Photonics west 2001-electronic imaging* (pp. 340–348). International Society for Optics and Photonics. doi:10.1117/12.429504.
40. Rushmeier, H.E., Rogowitz, B.E., & Piatko, C. (2000). Perceptual issues in substituting texture for geometry, In B.E. Rogowitz & T.N Pappas (Eds.), *SPIE Human Vision and Electronic Imaging* (pp. 372–383).
41. Shafi, M., Hashimoto, A., Umehira, M., Ogose, S., & Murase, T. (1997). Wireless communications in the twenty-first century: A perspective. *Proceedings of the IEEE, 85,* 1622–1638. doi:10.1109/5.640770.
42. Silva, S., Santos, B.S., Madeira, J., & Ferreira, C. (2008). Perceived quality assessment of polygonal meshes using observer studies: A new extended protocol, In *SPIE Human Vision and Electronic Imaging XIII*.
43. Sloan, P.-P. J., Weinstein, D. M., & Brederson, D. J. (1998). Importance driven texture coordinate optimization. *Computer Graphics Forum, 17,* 97.
44. Stanford, 2015. Standard Bunny [WWW Document]. URL http://graphics.stanford.edu/data/3Dscanrep/.
45. Turk, G. (2001). Texture synthesis on surfaces. In *Proceedings* of the 28th *annual conference* on *Computer graphics and interactive techniques*—SIGGRAPH'01 pp. 347–354.
46. Vatjus-anttila, J., Ventä-Olkkonen, L., & Häkkilä, J. (2013). On the edge of a virtual world—investigating users preferences and different visualization techniques, In *International joint conference on ambient intelligence*. New York: Springer.
47. Vatjus-Anttila, J.M., Koskela, T., & Hickey, S. (2013). Effect of 3D content simplification on mobile device energy consumption, In *Proceedings of International Conference on Making Sense of Converging Media, AcademicMindTrek'13* (p. 263). New York: ACM.
48. Watson, B., Friedman, A., & McGaffey, A. (2001). Measuring and predicting visual fidelity. In *Proceeding 28th annual conference on Computer graphics and interactive techniques* (pp. 213–220).
49. Wei, J., & Lou, Y. (2010). Feature preserving mesh simplification using feature sensitive metric. *Journal of Computer Science and Technology, 25,* 595–605.
50. Willmott, A. (2011). Rapid simplification of multi-attribute meshes In *High-performance graphics* (pp. 151–158). Lyon.
51. Xiao, Y., Bhaumik, R., Yang, Z., Siekkinen, M., Savolainen, P., & Ylä-Jääski, A. (2010). A system-level model for runtime power estimation on mobile devices, In *IEEE/ACM international conference on cyber, physical and social computing* (pp. 27–34).
52. Zhang, M., Zhang, L., & Wang, H. (2011). Geometry and texture coupled generalization of urban buildings. *Digital media and digital content management 2011 workshop*. Los Alamitos, CA: IEEE. doi:10.1109/DMDCM.2011.69.