PowerGS: Display-Rendering Power Co-Optimization for Foveated Radiance-Field Rendering in Power-Constrained XR Systems

WEIKAI LIN, University of Rochester, USA
SUSHANT KONDGULI, Reality Labs Research, Meta, USA
CARL MARSHALL, Reality Labs Research, Meta, USA
YUHAO ZHU, University of Rochester, USA

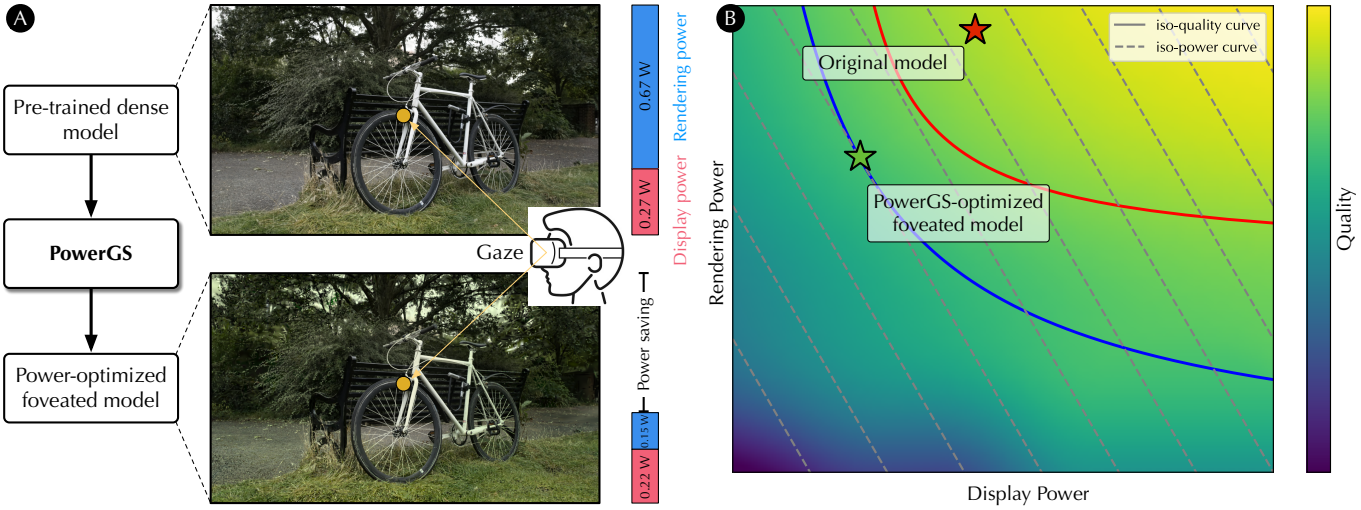


Fig. 1. (**A**): This paper presents PowerGS, a framework that translates a pre-trained, dense 3D Gaussian Splatting (3DGS) model to a power-optimal, foveated model. PowerGS for the first time jointly optimizes the rendering power and display power, the two main power consumers in power-constrained Extended Reality devices. (**B**): Given a quality constraint, PowerGS first estimates the iso-quality curve in the display-vs-rendering power landscape and then identifies the minimal-power point on the curve, giving a model that minimizes the total power while meeting the perceptual quality requirement. PowerGS integrates Foveated Rendering by using different models for different quality regions, each of which is optimized independently using the same PowerGS framework.

3D Gaussian Splatting (3DGS) combines classic image-based rendering, point-based graphics, and modern differentiable techniques, and offers an interesting alternative to traditional physically-based rendering. 3DGS-family models are far from efficient for power-constrained Extended Reality (XR) devices, which need to operate at a Watt-level. This paper introduces PowerGS, the first framework to *jointly* minimize the rendering and display power in 3DGS under a quality constraint. We present a general problem formulation and show that solving the problem amounts to 1) identifying the iso-quality curve(s) in the landscape subtended by the display and rendering power and 2) identifying the power-minimal point on a given curve, which has a closed-form solution given a proper parameterization of the curves. PowerGS also readily supports foveated rendering for further power savings. Extensive experiments and user studies show that PowerGS achieves up to 86% total power reduction compared to state-of-the-art 3DGS models, with minimal loss in both subjective and objective quality. Code is available at https://github.com/horizon-research/PowerGS.

CCS Concepts: • **Computing methodologies** → **Rendering**; **Image-based rendering**; • **Human-centered computing** → **Ubiquitous and mobile computing**.

Additional Key Words and Phrases: Foveated rendering, power optimization

ACM Reference Format:
Weikai Lin, Sushant Kondguli, Carl Marshall, and Yuhao Zhu. 2025. PowerGS: Display-Rendering Power Co-Optimization for Foveated Radiance-Field Rendering in Power-Constrained XR Systems. In *SIGGRAPH Asia 2025 Conference Papers (SA Conference Papers '25), December 15–18, 2025, Hong Kong, Hong Kong*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3757377.3763851

Authors' Contact Information: Weikai Lin, University of Rochester, USA, wlin33@ur.rochester.edu; Sushant Kondguli, Reality Labs Research, Meta, USA, sushantkondguli@meta.com; Carl Marshall, Reality Labs Research, Meta, USA, csmarshall@meta.com; Yuhao Zhu, University of Rochester, USA, yzhu@rochester.edu.

SA Conference Papers '25, December 15–18, 2025, Hong Kong, Hong Kong

ACM ISBN 979-8-4007-2137-3/2025/12
https://doi.org/10.1145/3757377.3763851

1 Introduction

Extended Reality (XR), including Virtual Reality (VR) [Meta 2022] and Augmented Reality (AR) [Meta 2024; Microsoft 2019; XREAL 2023], is seen as the next ubiquitous personal computing platform. While XR demands high visual quality, its system power is severely limited by battery capacity. Unfortunately, there is no Moore's law for battery — the energy density of battery technology is fundamentally limited [Halpern et al. 2016; Leng et al. 2019; Schlachter

2013]. This increasingly creates a tension between visual quality and power consumption, the latter of which consists of both the rendering power and the display power.

Meanwhile, radiance field-based rendering using NeRF [Mildenhall et al. 2020] or 3D Gaussian Splatting (3DGS) [Kerbl et al. 2023; Radl et al. 2024; Yu et al. 2024a] is emerging as a promising rendering paradigm for XR [Gafni et al. 2021; Luo et al. 2024; Saito et al. 2024; Xu et al. 2023]. This paper investigates how to design power-optimal 3DGS models under quality requirements.

Prior work focuses exclusively on reducing either the rendering or the display power. On the rendering side, recent effort to prune 3DGS models [Fan et al. 2024; Fang and Wang 2024; Franke et al. 2024; Girish et al. 2024; Lee et al. 2024b; Lin et al. 2025] demonstrates that pruning points can effectively accelerate 3DGS and, thereby, reducing the rendering power. Similarly, display power can be minimized by dimming pixels [Shye et al. 2009; Yan et al. 2018] or color adjustments [Chen et al. 2024; Duinkharjav et al. 2022].

The single-minded optimization of only one power consumer likely leads to sub-optimal results. For instance, one can train a model with fewer Gaussian points but the resulting model might use pixel colors that consume high display power. Similarly, a display-power optimization, e.g., a post-processing filter that adjust pixels to using power-efficient colors based on human color discrimination [Duinkharjav et al. 2022], could have little effect on the rendering power, because no rendering work is saved.

This paper proposes PowerGS, a framework that jointly optimizes the display and rendering power for 3DGS-family models. We give a general problem formulation that minimizes the total power consumption under a quality constraint. The problem amounts to 1) identifying the iso-quality curve(s) in the landscape subtended by the display and rendering power and 2) identifying the power-minimal point on such a curve. The right-most plot in Fig. 1 offers a visual intuition: each curve is an iso-quality curve for a given quality constraint, and each dash line is an iso-power line ($x + y = P$, where P is the total power). The goal is to identify the lowest iso-power line that just intersects with a desired iso-quality curve.

Obtaining the iso-quality curves in the power landscape is challenging because of the non-differentiable nature of power consumption w.r.t. model parameters. We approach this using a sample-and-reconstruct framework. We first sample a set of 3DGS models that have similar quality via pruning a dense model — by differentially allocating the quality budget between reducing display vs. rendering power until a quality requirement is met. We then propose a parameterization of the iso-quality curve. This parameterization allows the curve to be robustly reconstructed from just about 5 samples of the pruned models and gives rise to a closed-form solution for finding the power-minimal model.

We extend PowerGS to support foveated rendering (FR). Without losing generality, we assume a FR paradigm demonstrated in Lin et al. [2024, 2025], where the Field-of-View (FoV) is divided into quality regions, each rendered by a separate 3DGS model. We show that the PowerGS framework can be applied to FR in a plug-and-play manner — by optimizing each region/model independently using the method described above *without* changing the FR architecture.

We validate our method through both subjective user study and objective power and quality metrics. We show that with little quality degradation, PowerGS achieves 63% (Mip-NeRF 360 dataset) and 52% (Synthetic NeRF dataset) total power reduction compared to Mini-Splatting [Fang and Wang 2024] and 3DGS [Kerbl et al. 2023], respectively. PowerGS will be open-sourced. To summarize, this work makes the following key contributions:

- We propose a general formulation for jointly optimizing rendering and display power for radiance-field rendering.
- We propose a sample-and-reconstruction framework to obtain iso-quality functions, using which we can identify the power-minimal models through a closed-form solution.
- We extend our approach to foveated rendering.
- Through subjective and objective measurements, we show that 3DGS models generated by PowerGS generally have similar or higher perceptual quality at the same total power consumption compared to existing models.

2 Related Work

2.1 Efficient (Neural) Radiance-Field Rendering

Pioneered by Neural Radiance Fields (NeRF) [Mildenhall et al. 2020], radiance-field rendering combines classic image-based rendering with modern differentiable rendering techniques and offers a compelling alternative to photorealistic rendering. Recent Point-Based Radiance-Field Rendering techniques [Kerbl et al. 2023; Radl et al. 2024; Yu et al. 2024a], exemplified by 3D Gaussian Splatting [Kerbl et al. 2023], accelerates NeRF-family models by replacing implicit radiance field representations with explicit points-based rendering primitives [Gross and Pfister 2011]. A huge amount of recent work on 3DGS models improve the rendering quality and widen its applicability [Dong et al. 2024; Duan et al. 2024; Gao et al. 2024; Huang et al. 2024; Jiang et al. 2024; Li et al. 2024; Liang et al. 2024; Lyu et al. 2024; Mujkanovic et al. 2024; Peng et al. 2024; Radl et al. 2024; Rao et al. 2024; Wang et al. 2024a; Yang et al. 2024a; Yu et al. 2024b].

However, deploying 3DGS-family models on power-constrained XR devices remains challenging. In addition to rethinking the rendering pipeline for efficiency [Duckworth et al. 2024; Kerbl et al. 2024; Moenne-Loccoz et al. 2024; Tong and Hachisuka 2024; Yang et al. 2024b], many existing methods that accelerate 3DGS models use point pruning as a key technique [Fan et al. 2024; Fang and Wang 2024; Girish et al. 2024; Lee et al. 2024b]. Pruning points reduces the amount of work a model executes and, thus, reduces the rendering power. However, existing pruning methods ignore the display power and do not aim at total power minimization.

2.2 Display Power Optimization

Displays [Koulieris et al. 2019] constitute another important component of the XR device power consumption. In an AR device, the rendering power budget is around 1 W [Philip Berne 2023] and the display power consumption is around 0.3 W [GooDisplay [n. d.]], which could be even higher in scenarios where the luminance of the rendered images has to rival with that of the real scene.

In emissive displays such as Organic Light-Emitting Diode (OLED) displays that are becoming popular in consumer devices, the display power consumption is dictated by pixel colors (luminance + chromaticity). Shye et al. [2009] and Yan et al. [2018] leverage light

adaptation [Wandell 1995, Chpt. 5] to gradually reducing the luminance (i.e., uniforming reducing the pixel values). Duinkharjav et al. [2022] uses the eccentricity-dependent nature of human color discrimination [MacAdam 1942] to design a filtering technique to adjust pixel chromaticity to save display power without introducing visual artifacts. Chen et al. [2024] compares a number of different techniques to reduce display power and their quality implications.

These methods optimize only for the display power. In contrast, we formulate a joint display-rendering power optimization problem such that the resulting models have lower total power while providing a similar perceptual quality (Sec. 6.2).

2.3 Foveated (Radiance-Field) Rendering

Foveated rendering (FR) is a classic technique that leverages the eccentricity-dependent visual acuity falloff in human vision to reduce rendering quality in the visual periphery, significantly improving rendering speed and/or power consumption [Guenter et al. 2012; Patney et al. 2016; Sun et al. 2017; Ujjainkar et al. 2024; Wang et al. 2024b; Ye et al. 2022; Zhang et al. 2024]. FR has seen its use in commercial visual displays [Meta 2024] and is supported in common graphics programming models [Nvidia 2018].

FR techniques have been applied to (neural) radiance-field rendering. For instance, FoV-NeRF [Deng et al. 2022], VRS-NeRF [Rolff et al. 2023] and Shi et al. [2024] integrate foveated rendering into NeRF; Franke et al. [2024]; Lin et al. [2024, 2025] apply FR to accelerate 3DGS. This paper does *not* propose a new FR method; rather, we show that our joint power optimization can be easily integrated into FR (using Lin et al. [2024] as an example) to reduce the power consumption of FR.

3 Problem Formulation

This section formulates a general problem that integrates quality relaxation, rendering power, and display power. The next section develops a practical solution for it.

3.1 Display and Rendering Co-Optimization

Unlike previous works that trade quality solely for rendering or display power, minimizing the total power requires solving the following constrained optimization problem[1]:

$$\arg\min_{\mathcal{P}} \left[\overline{P_{\mathrm{D}}}(\mathcal{P}) + \overline{P_{\mathrm{R}}}(\mathcal{P}) \right] \quad \text{subject to } \overline{Q}(\mathcal{P}) \geq Q_{\min}, \tag{1}$$

where $\mathcal{P}$ is a 3DGS model to be optimized for; $\overline{P_{\mathrm{D}}}(\mathcal{P})$, $\overline{P_{\mathrm{R}}}(\mathcal{P})$, and $\overline{Q}(\mathcal{P})$ denote the average display power, rendering power, and quality when running the model $\mathcal{P}$, respectively, and are expressed as:

$$\overline{P_{\mathrm{R}}}(\mathcal{P}) = \frac{1}{|\mathcal{T}|} \sum_{T \in \mathcal{T}} P_{\mathrm{rend}}(\mathcal{P}, T), \tag{2a}$$

$$\overline{P_{\mathrm{D}}}(\mathcal{P}) = \frac{1}{|\mathcal{I}|} \sum_{I \in \mathcal{I}} P_{\mathrm{disp}}(I), \tag{2b}$$

$$\overline{Q}(\mathcal{P}) = \frac{1}{|\mathcal{I}|} \sum_{I \in \mathcal{I}} Q(I), \tag{2c}$$

where $\mathcal{T}$ represents the sampled poses over, e.g., a dataset, while $\mathcal{I}$ represents images rendered from these poses. $P_{\mathrm{rend}}(\mathcal{P}, T)$ represents the rendering power when running $\mathcal{P}$ on a pose T, $P_{\mathrm{disp}}(I)$ represents the display power of displaying the image I rendered by $\mathcal{P}$ on T, and $Q(I)$ represents the perceptual quality of I.

While the perceptual quality metrics are firmly established in the literature (e.g., PSNR, SSIM, HVSQ [Walton et al. 2021]), the rest of this section will discuss how the display power and rendering power are modeled.

3.2 Display Power Modeling

For emissive displays (e.g., OLEDs) commonly used in XR devices, power consumption is determined by the pixel values of the displayed image. Actual display measurements show that the display power $P_{\mathrm{disp}}(I)$ for an image I can be modeled as a linear combination of the average R, G, and B channel values expressed in a linear (sRGB) color space [Chen et al. 2024; Duinkharjav et al. 2022]:

$$P_{\mathrm{disp}}(I) = \alpha \overline{R}(I) + \beta \overline{G}(I) + \gamma \overline{B}(I) + s, \tag{3}$$

where $\overline{R}(\cdot)$, $\overline{G}(\cdot)$, and $\overline{B}(\cdot)$ denote the average linear sRGB-space values of the R, G, and B channels across all pixels in the image I, respectively. The coefficients α, β, and γ represent the power consumption costs associated with each channel, while s accounts for the static power of peripheral circuitry, which is independent of the display content. The four coefficients are usually experimentally fit with measurement data by sampling a set of colors.

3.3 Rendering Power Modeling

Ideally, the rendering power model should be built from measurements on real hardware just like how the display power model is built. However, today's mobile computing hardware, such as Nvidia's Jetson series [NVIDIA 2018] and the Qualcomm's XR2+ platform [Qualcomm 2022] (used in Meta Quest Pro), all consume upwards of 10 to 20 Watts, in our measurements, when running 3DGS models, making them ill-suited for future XR platforms whose power budgets are usually at the Watt level [Berne 2024; LiKamWa et al. 2014; McLellan 2019; XREAL 2023]. The high power consumption is primarily due to the general-purpose nature of the computing hardware (CPUs or GPUs), where a huge amount of power is wasted on moving data rather than the actual computation [Hameed et al. 2010; Qadeer et al. 2013].

Dedicated Application-Specific Integrated Chips (ASICs) are known to address the power inefficiencies in general-purpose processors, and have been widely used in application domains such as deep learning [Jouppi et al. 2017], video processing [Ranganathan et al. 2021], and robotics [Murray et al. 2016; Wan et al. 2021]. Recently

[1] We considered reformulating the optimization problem in Eqn. 1 using the Lagrange multiplier, where the quality constraint is expressed as an additional term in the objective function. But still the objective function, especially the rendering power term, lacks an analytical form, so we could not directly apply the Lagrange method to solve the optimization problem. Our current formulation makes it explicit that our goal is to minimize power under the quality constraint.

researchers have started exploring ASICs for neural rendering and 3DGS [Feng et al. 2024a; Fu et al. 2023]. Lin et al. [2025] and Lee et al. [2024a] demonstrate that ASICs can achieve 15×–50× energy reductions over GPUs. Our power modeling is thus based on ASICs to target futuristic XR devices.

We develop a power estimation model for 3DGS-based ASICs, following a widely adopted approach in modeling ASIC power consumption [Sze et al. 2020; Yang et al. 2018, 2017]. Specifically, the energy consumption of ASICs is divided into three components: the energy of executing floating-point operations (FLOPs), energy of retrieving data from on-chip memory (called the Static RAMs, or SRAMs), and the energy of retrieving data from off-chip memory (called the Dynamic RAMs, or DRAMs). Each energy component itself is estimated by the product of the unit energy (i.e., the energy consumption per FLOP, per Byte from SRAM/DRAM) and the number of total FLOPs/Bytes retrieved from the memory.

The total rendering power of running a model $\mathcal{P}$ at a pose T is:

$$P_{\text{rend}}(\mathcal{P}, T) = \left(e_{\text{FLOP}}\#_{\text{FLOP}} + e_{\text{SRAM}}\#_{\text{SRAM}} + e_{\text{DRAM}}\#_{\text{DRAM}}\right) \cdot \text{FPS} \quad (4)$$

where e_{FLOP}, e_{SRAM}, and e_{DRAM} are the unit energy, which are hardware dependent but independent of $\mathcal{P}$ and T; $\#_{\text{FLOP}}$, $\#_{\text{SRAM}}$, and $\#_{\text{DRAM}}$ represent the total number of FLOPs, SRAM accesses, and DRAM accesses *per frame*, which depend on $\mathcal{P}$, T, and the hardware architecture. Scaling the total energy per frame by the FPS (which we assume to be 60) gives us the average rendering power (energy per second).

In general, $\#_{\text{FLOP}}$, $\#_{\text{SRAM}}$, and $\#_{\text{DRAM}}$ cannot be expressed analytically with respect to the parameters of the model $\mathcal{P}$ because of the discrete nature of both the software (e.g., model parameters are defined per point) and hardware execution (which proceeds in steps of clock cycles). Instead, we obtain the statistics by simulating how a 3DGS model is actually executed on the hardware. This model is employed for rendering power estimation, as described in Sec. 6.1. We omit the hardware and its simulation for simplicity sake, but refer interested readers to the Supplemental Material A for details.

4 The PowerGS Method

Solving Eqn. 1 is challenging. It lacks a closed-form solution, and the non-differentiable rendering power model (Eqn. 4), which generally cannot be analytically related to the model parameters, makes numerical methods like gradient descent inapplicable.

The key observation is that any 3DGS model eventually casts to a 3D landscape subtended by the rendering power, display power, and quality of the model. Fig. 1(B) shows a 2D visualization, where the x-axis and y-axis represent the display and rendering power, respectively, and the colorscale represents the quality (PSNR here). Our job is to identify the iso-quality curve in the display-vs-rendering power plot given a particular quality constraint (Q_{min} in Eqn. 1) and then identify the power-minimal point on the curve.

We approach this problem through a sample-and-reconstruct approach, where we first sample a few models that have the same/similar quality (Sec. 4.1) and then reconstruct the iso-quality curve from the samples (Sec. 4.2). Given the iso-quality curve (Sec. 4.3), the generally intractable problem in Eqn. 1 is then cast to a simple convex optimization with a closed-form solution (Sec. 4.4).

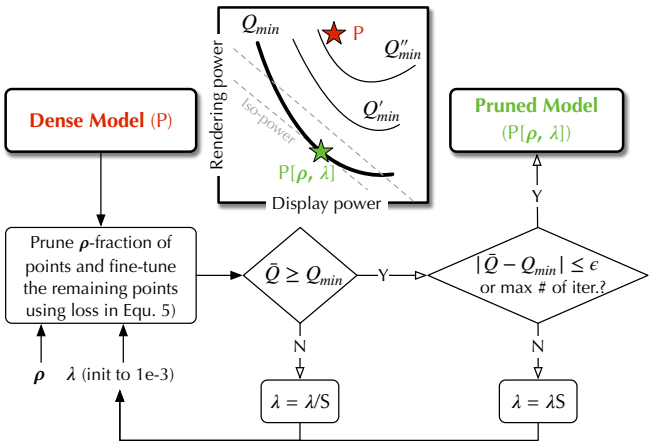


Fig. 2. PowerGS turns a pre-trained, dense 3DGS model into a power-optimal model given a quality constraint Q_{min}. An iso-qualiry curve closer to the top-right corner has a higher quality (i.e., $Q''_{min} > Q'_{min} > Q_{min}$). Given a ρ, PowerGS iteratively identifies a λ that, together with ρ, results in a pruned model $P[\rho, \lambda]$ that lands on a given iso-quality curve. The iso-quality curves are parameterized to be convex, and identifying the power-minimal power on a given iso-quality curve has a closed-form solution.

4.1 Sampling Iso-Quality Models

The first step is to sample models that have a similar quality but differ in power. Searching the entire model parameter space is clearly intractable, because a model involves millions of points, each with many attributes. We observe that the main factor that affects the visual quality and rendering power is the number of points in a model [Fan et al. 2024; Fang and Wang 2024; Girish et al. 2024; Lee et al. 2024b; Lin et al. 2024, 2025]. So we start from a pre-trained dense model and prune the points, modulated by the pruning ratio ρ: a higher ρ reduces the rendering quality/power, and vice versa.

To modulate the display power, we weight the visual quality and display power in a loss function:

$$-\overline{Q}(\mathcal{P}) + \lambda \overline{P_{\text{D}}}(\mathcal{P}). \quad (5)$$

The parameter λ controls the weight of display power. Increasing λ shifts pixels toward more power-efficient colors, reducing the display power. However, this color shift also affects rendering quality $\overline{Q}(\mathcal{P})$, which, when operating under a fixed quality budget, translates to less aggressive rendering power reduction. This is why the loss in Eqn. 5 allows us identify iso-quality models that vary in display-vs-rendering power consumptions. We quantify the trade-off between quality and display power as λ varies in Sec. 6.7.

Fig. 2 shows how the loss is used. For a sampled ρ, we first prune ρ proportion of points and then fine-tune of the remaining points using Eqn. 5, during which we monitor the quality $\overline{Q}(\mathcal{P})$ every certain number of iterations. The idea is that if the quality is higher than Q_{min}, we increase λ to trade the surplus quality for lowering the display power. We adjust λ as follows:

$$\lambda \leftarrow \begin{cases} \lambda \cdot S, & \text{if } \overline{Q}(\mathcal{P}) \geq Q_{\text{min}}, \\ \lambda / S, & \text{if } \overline{Q}(\mathcal{P}) < Q_{\text{min}}, \end{cases} \quad (6)$$

where $S > 1$ controls the adjustment scale (we use $S = 2$). In practice, S is gradually reduced to 1 using cosine annealing, which decreases the fluctuations of λ and helps convergence. The whole process terminates when the model quality is just ϵ-higher that Q_{min} (or the maximum fine-tuning iteration has been reached). Each resulting, pruned model lands on the corresponding iso-quality curve with a display power (x-axis) and rendering power consumption (y-axis).

4.2 Parameterizing and Reconstructing Iso-Quality Curve

Given a ρ we can now quickly arrive at a λ that, together with ρ, gives us a model that lands on the iso-quality curve. We parameterize the iso-quality curve and approximately reconstruct it by analytically fitting to the samples. We represent the iso-quality curve in the display-vs-rendering power graph as a parametric equation:

$$P_r = \mathcal{R}(\rho) \tag{7}$$

$$P_d = \mathcal{D}(\rho), \tag{8}$$

where P_r and P_d represent the rendering power (y-axis in Fig. 2) and display power (x-axis in Fig. 2), respectively; they are parameterized by the pruning ratio ρ through the function $\mathcal{R}(\cdot)$ and $\mathcal{D}(\cdot)$, respectively, both of which are specific to a particular quality constraint.

Intuitively, as the pruning ratio increases the rendering power would reduce but the display power would have to increase to make up for the quality loss. In practice, we find that the reduction of rendering power (increase of the display power) as ρ reduces follows the inverse Michaelis–Menten kinetics [Michaelis and Menten 1913], which is commonly used in biological sciences to describe the incremental saturation observed when a stimulus property changes [Baylor et al. 1974; Dugdale 1967; Schneeweis and Schnapf 1995]. Specifically:

$$\mathcal{D}(\rho) = 1 - \frac{V_d(1-\rho)}{K_d + (1-\rho)}, \quad \mathcal{R}(\rho) = 1 - \frac{V_r\rho}{K_r + \rho}, \tag{9}$$

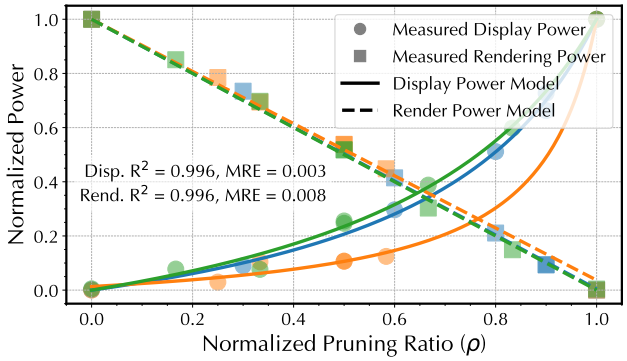
where V_d, K_d, V_r, and K_r are free parameters fit to data. Given that both models are parameterized by only two free parameters, we can afford to sample only about 5 ρs to derive an iso-quality curve.

4.3 Model Results and Discussion

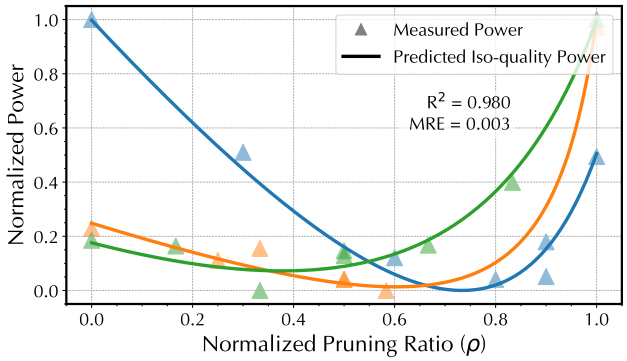
Fig. 3a shows the accuracy of our parameterization and regression using three traces from the Mip-Nerf360 [Barron et al. 2022] and the Synthetic NeRF dataset [Mildenhall et al. 2020]. The markers represent the measured data under different values of ρ and the solid and dashed lines represent the regressed display and rendering power function, respectively. The mean relative errors (MRE) of the two regressions are 0.003 and 0.008, respectively, indicating high regression accuracy.

Empirically, display power rises with pruning ratio ρ. This is because we are operating under a fixed quality budget, which can be allocated between rendering and display power reduction. If more of this budget is used for reducing rendering power (pruning), less remains for display power reduction, increasing display power.

Interestingly, the rendering power function is almost linear with respect to ρ. This is because we use the efficiency-aware pruning proposed in Lin et al. [2025] (which is shown to better reduce the rendering cost compared to prior methods [Fan et al. 2024; Fang and Wang 2024]), which prioritizes pruning points that intersect with more tiles (thus consume more power), so the rendering power is roughly proportional to pruning ratio.



(a) We regress the parameterized display power model and rendering power model (Eqn. 9) using about 5 samples of ρ.



(b) The total power model (curves) reconstructed from summing the rendering and display power models predicts the measured data (markers) well.

Fig. 3. Accuracy of the rendering and display power models (top) and the iso-quality curves (bottom). `bicycle` (blue), `flower` (orange), and `hotdog` (green) scenes are used here. The low coefficients of determination (R^2) and the mean relative errors (MRE) show the accuracy of our models. Power numbers and pruning ratios are normalized to the [0, 1] range on a per-scene basis for both visualization and regression stability (see Supplementary Material B.1). Each ρ has an associated λ (see Fig. 2) but not shown.

Given the parameterized display and rendering power models, the total power model is just the sum of the two. Fig. 3b shows that the resulting total power model (continuous curves) predicts the measured data (markers) well. Recall that each total power function represent an iso-quality curve, so all the points on each curve share the same quality. The non-monotonic nature of the iso-quality curves suggests that there exists an optimal ρ that minimizes the total power, confirming the need for joint rendering and display power optimzations.

Why not a ML Proxy Model? We also experimented with an alternative method that trains a machine-learning–based proxy model that learns to map 3DGS parameters to power. Our results

show that its accuracy fall flat (next to 0) given the same training time as our method (about 2.5 hrs; see Sec. 6.1). The proxy model is fundamentally inefficient to learn because the rendering power is dictated by the *number* of Gaussian points, much less by exactly *which* Gaussians are used and their parameters. Our method, instead, parameterizes the rendering power directly as a function of the pruning ratio, requiring only about 5 samples and giving rise to a closed-form solution (without SGD), as we will show next.

4.4 Deriving Power-Minimal Model

The general problem in Eqn. 1 is now reduced to the following:

$$\arg\min_{\rho} \mathcal{D}(\rho) + \mathcal{R}(\rho), \tag{10}$$

which gives us a ρ, using which we prune the dense model (Sec. 4.1) to obtain a concrete, power-minimal model.

A nice property of the Michaelis–Menten parameterization is that both $\mathcal{D}(\rho)$ and $\mathcal{R}(\rho)$ are convex, so $\mathcal{D}(\rho) + \mathcal{R}(\rho)$ is convex, too. In fact, Eqn. 10 has a closed-form solution; see Supplementary Material B for details. Finding the solution amounts to identifying the lowest point in Fig. 3b or, equivalently, shifting the iso-power line in Fig. 2 until it is tangential to the iso-quality curve.

5 Supporting Foveated Rendering (FR)

Our power optimization can be integrated into FR in a plug-and-play manner. We assume RTGS, an FR algorithm for 3DGS [Lin et al. 2024, 2025], as the FR baseline. We claim no novelty of the basic FR method. Our contribution is to show PowerGS can be trivially extended to jointly optimize display and rendering power in FR. We focus on the main idea here; see Supplementary Material C for more implementation details.

RTGS represents the multi-model FR paradigm commonly applied to radiance-field rendering [Deng et al. 2022; Franke et al. 2024], where 1) the Field-of-View (FoV) is divided into multiple quality regions (each covering a range of eccentricities) and 2) each quality region is rendered by a separate 3DGS model. In particular, the model in each region is pruned from a dense model: low-quality regions are more heavily pruned and vice versa.

The nature of "one model per quality region" lends itself to our power optimization. The idea is to apply the PowerGS method to each model/quality region independently. Specifically, we start with a dense model and prune it using the method described in Sec. 4 to obtain the model for region 1 (R_1), which is then used as the starting point for pruning to obtain the model for region 2 (R_2). This process is repeated for subsequent, progressively lower-quality regions.

The quality constraint used during pruning (Q_{min} in Eqn. 1) must be eccentricity dependent. As with RTGS, we use a ventral metamerism-inspired [Freeman and Simoncelli 2011], eccentricity-dependent quality metric, termed Human Vision System Quality (HVSQ) metric [Walton et al. 2021, 2022] (see Supplementary Material C.3 for details of HVSQ). When pruning models for each region, we align the HVSQ loss across all the quality regions, ensuring uniform perceptual quality in the FoV.

6 Evaluation

6.1 Experiment Setup

Power Estimation. For the display power model (Eqn. 3), we use parameters obtained from a real OLED measurement used in prior work [Chen et al. 2024; Duinkharjav et al. 2022].

The unit energy numbers used in the rendering power model (Eqn. 4) are based on real hardware measurements reported in prior work [Ahn et al. 2024; Tortorella et al. 2022]. Briefly, the energy per FLOP is 0.53 pJ, SRAM access consumes 0.24 pJ per byte, and DRAM access is 10.88 pJ per byte. We implement a hardware simulator to get the operation counts. The hardware is modeled after GSCore [Lee et al. 2024a], a recent 3DGS hardware accelerator. See Supplemental Material A for the hardware modeling details.

Dataset. We evaluate our method on two datasets: a real-world dataset Mip-NeRF360 [Barron et al. 2022] and the Synthetic NeRF dataset [Mildenhall et al. 2020]. The total (display + rendering) power ratio between the two datasets is approximately 3.5:1. We use Mini-Splatting-D [Fang and Wang 2024] as the pre-trained dense model for Mip-NeRF360 dataset and we use 3DGS [Kerbl et al. 2023] as a dense model for the Synthetic NeRF dataset, as they achieve the highest quality in the respective datasets. Supplemental Material D contains additional results on Tanks&Temples [Knapitsch et al. 2017] and Deep Blending [Hedman et al. 2018], two widely used datasets in neural rendering.

Variants and Overhead. Our variants include: PowerGS-H, PowerGS-M, and PowerGS-L, with decreasing quality and total power consumption. The R_1 PSNR and SSIM requirements for these variants are set to be 99%, 98%, and 97%, respectively, of the dense model. For levels beyond 1, the eccentricity-dependent HVSQ metric [Walton et al. 2021, 2022] is set to match that of level 1 in pruning the models. See Supplemental Material C for more training details.

Power optimization is done once for each model with a *one-time, offline* cost of about 2.5 hours on an RTX-4090 machine without additional run-time and power overhead. This does mean that the power model/iso-quality curves are scene specific, but this is inherent to/consistent with any radiance-field rendering method, where a model is learned specifically for a scene.

Baselines. We compare against following baselines:

- Dense 3DGSs: 3DGS [Kerbl et al. 2023], the first 3DGS model, and Mini-Splatting-D [Fang and Wang 2024], a recent improvement upon 3DGS. They optimize solely for quality, resulting in high power consumption.
- Pruned 3DGSs: LightGS [Fan et al. 2024], which is pruned from 3DGS, and Mini-Splatting [Fang and Wang 2024], which pruned from Mini-Splatting-D. They ignore the display power and do not support FR. We provide three variants of LightGS (H, M, L) by varying ρ (66%, 76%, 86%).
- FR methods optimizing only for rendering power (FR-Render) and only for the display power (FR-Display) in each region. FR-Render is RTGS [Lin et al. 2024, 2025], and FR-Display is a variant of PowerGS that fine-tunes the pixel values of a dense model *without* pruning. They also have the three H, M, and L variants like PowerGS.

All FR methods are implemented with blending across region boundaries using the method described in Guenter et al. [2012].

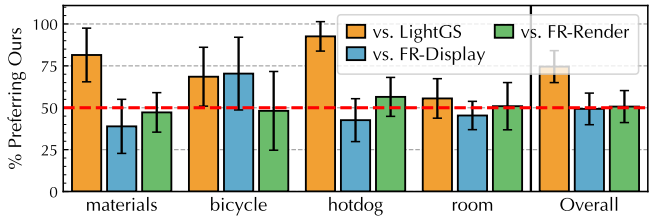


Fig. 4. 2IFC results showing the fraction of times users prefer our method over the baselines. Ours is significantly better than LightGS ($p < 0.01$) and has an insignificant difference compared to the other two baselines ($p > 0.8$ in both cases).

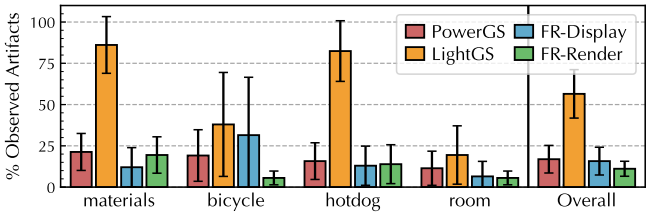


Fig. 5. 2AFC results showing the proportion of times users report seeing artifacts in each method. Users are significantly less likely to report seeing artifacts in ours than in LightGS ($p < 0.01$); ours has an insignificant difference compared to FR-Display ($p > 0.57$) and to FR-Render ($p > 0.07$) in all scenes except `bicycle`.

6.2 User Study

We conduct an IRB-approved user study to show that PowerGS provides a subjective quality no worse than other baseline methods while having a lower power consumption. We compare with LightGS-L, FR-Render-H, and FR-Display-L, over which PowerGS-H saves 16.8%, 12.3%, and 50.6% power, respectively. We do not include dense methods like 3DGS due to their excessive power consumption (2.7 × higher than PowerGS-L) and low frame rate (~5 FPS on Jetson Xavier) on XR-suitable mobile devices.

Procedure. We select the `bicycle` and `room` scenes from the Mip-NeRF360 dataset and the `hotdog` and `materials` scenes from the Synthetic NeRF dataset for evaluation. We interpolate the sparse poses in the dataset interpolation to generate 1,350 poses per scene, forming three 5-second clips for each scene at 90 FPS.

We recruit 9 participants (5 male and 4 female; age 20 to 30), all with normal or corrected-to-normal vision. We first conduct a classic preference-based Two-Interval Forced Choice (2IFC) study commonly used in psychophysical experiments for perceptual rendering [Chen et al. 2024; Deng et al. 2022; Guenter et al. 2012; Perez-Ortiz et al. 2019; Rolff et al. 2023; Shi et al. 2024; Walton et al. 2021]. For each comparison, we present a clip from PowerGS-H and a clip from a baseline method in random order and ask participants to select their preferred version. A 1-second interval is added between videos in each comparison. Each comparison is repeated 4 times, resulting in 144 randomized (4 scenes × 3 clips × 3 comparison × 4 repeats) total comparisons (288 videos) for each participant. The experiment takes approximately 1 hour per participant.

We also perform a Two-Alternative Forced Choice (2AFC) experiment, where each participant is asked to respond whether they see artifact in each video, following the procedure used by a prior display-power optimization work [Duinkharjav et al. 2022].

Result. Fig. 4 shows the 2IFC results, where y-axis shows the fraction of times participants prefer PowerGS rendering compared to each of the three baselines (chance level is 0.5). The error bars represent the standard errors. PowerGS performs significantly better than LightGS (two-sided binomial test $p < 0.01$). The difference between PowerGS and the other two baselines is statistically insignificant ($p > 0.8$ in both cases).

Fig. 5 shows the 2AFC results, where the y-axis shows the proportion of reported artifacts in each method. PowerGS has a significantly lower artifact ratio compared to LightGS (two-tailed two-proportion z-test $p < 0.01$). The differences between PowerGS and FR-Display are insignificant ($p > 0.57$). The difference w.r.t. to FR-Render is insignificant ($p > 0.07$) except in `bicycle`.

Fig. 13 shows one `bicycle` frame rendered by PowerGS and all the baselines. LightGS-L in the foveal region has the artifact around the tire that many participants notice. Both variants of FR-Display, to save display power, shade the foveal region overwhelmingly green that many report. PowerGS shades the peripheral region (which is originally achromatic) in a yellow-green hue compared to FR-Render, which leads to a higher proportion of reported artifacts. See Sec. 7 for further discussions of this color shift.

6.3 Quality-Power Trade-offs

We now compare the quality-power trade-offs using objective quality metrics. For the FR variants, the quality is obtained by applying the model trained for the foveal region (R1) *globally* to the entire frame so that we can make a fair comparison with other non-FR methods: standard metrics (PSNR, SSIM, LPIPS) target foveal quality and all FR methods already align HVSQ across regions. Fig. 6 compares the total power (x-axis) and SSIM (y-axis) of our method and the baselines on the Mip-NeRF360 dataset (top) and the Synthetic NeRF dataset (bottom). PowerGS, LightGS, and the two FR baselines all have three variants each. The trends on PSNR and LPIPS metrics are similar (Fig. 7 and Fig. 8).

PowerGS achieves the best trade-off among all methods in both datasets. PowerGS variants Pareto-dominate existing pruned models (LightGS and Mini-Splatting). Compared with FR-Render and FR-Display, which optimize only for the rendering power or display power, PowerGS variants consume 13.1% and 52.5% less power, respectively, while staying within 0.005 PSNR/SSIM loss. Compared with the dense models (Mini-Splatting-D and 3DGS), PowerGS reduces over 40% power with < 0.01 PSNR/SSIM loss.

As discussed in Sec. 3.3, our rendering power is modeled analytically rather than directly from a measurement on existing XR hardware. However, our method is applicable to today's mobile GPUs too. For instance, PowerGS-L on Jetson Xavier GPU obtained a 13.7× rendering power saving (no display power) on `bicycle`.

6.4 Varying Display vs. Rendering Power Ratio

We perform a sensitivity study to understand how our power saving will vary given different display vs. rendering power ratios by

Fig. 6. SSIM-vs-power trade-offs between ours and the baselines on the Mip-NeRF360 (top) and the Synthetic NeRF (bottom) dataset.

Fig. 7. PSNR-vs-power trade-offs between ours and the baselines on the Mip-NeRF360 (top) and the Synthetic NeRF (bottom) dataset.

Fig. 8. LPIPS-vs-power trade-offs between ours and the baselines on the Mip-NeRF360 (top) and the Synthetic NeRF (bottom) dataset.

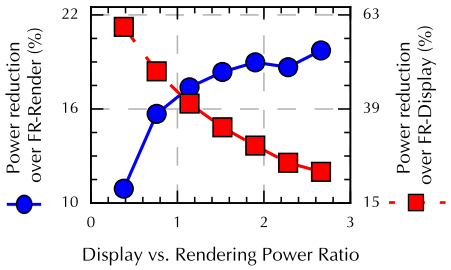


Fig. 9. Power saving of PowerGS over FR-Render (left y-axis) and over FR-Display (right y-axis) as the display vs. rendering power ratio varies (`bicycle` scene); all H variants.

sweeping the ratio from 0.38 to 2.66. Fig. 9 shows the power saving of PowerGS over FR-Render (left y-axis) and FR-Display (right y-axis) using the `bicycle` scene (all H variants). As the display power becomes dominant, the advantage of PowerGS over reducing only the rendering power increases, and vice versa. For instance, when the display vs. rendering power ratio is 2.66 : 1, PowerGS reduces 20%, 23%, and 38% power over FR-Render, FR-Display, and the dense model (not shown), respectively.

6.5 PowerGS Outperforms Two-Stage Baseline

We further compare against a two-stage baseline that combines RTGS [Lin et al. 2024] with a post-processing, image-space filter [Duinkharjav et al. 2022] (hereafter D22) for display power reduction, denoted as RTGS+D22. This baseline first train a model using FR-Render-H to reduce the rendering power, and then apply D22 to further reduce the display power. We use the `bicycle` scene and evaluate on Jetson Xavier GPU; the results are shown in Tbl. 1.

Table 1. Comparison against RTGS+D22 on the `bicycle` scene.

Method	FPS ↑	Power ↓	HVSQ ($\times 10^{-5}$) ↓			
			R1	R2	R3	R4
RTGS + D22	78	0.34 W	2.1	2.6	3.4	5.0
PowerGS	**84**	**0.32 W**	**2.1**	**2.1**	**2.1**	**2.1**

We first compare the speed and power. PowerGS runs 8% faster as RTGS+D22 adds ~10% runtime latency from the D22 post-processing. In contrast, our method only incurs a training-time overhead (60% more training time), which is paid once offline without any rendering-time overhead. RTGS+D22 also consumes 6% more power. This is not only due to the extra filter but also because PowerGS jointly optimizes the rendering and display power whereas RTGS+D22 optimizes the two aspects greedily in a sequence.

We evaluate quality using the eccentricity-dependent HVSQ [Walton et al. 2022] across all regions (R1–R4; see Supplementary Material C). Our method preserves consistently high visual quality across the visual field. In contrast, RTGS+D22 shows up to 2.4× worse quality (R4). This degradation arises because RTGS already prunes the model to a given quality constraint, and applying D22 on top further reduces fidelity, violating the constraint. The quality degradation in RTGS+D22 is also visually observable.

6.6 Non-Foveated Scenarios

We also evaluate PowerGS in non-foveated scenarios (Tbl. 2), where the entire image is treated as the fovea and rendered using the highest-quality model. We compare PowerGS against two dense baselines (3DGS and Mini-Splatting-D) as well as variants optimized only for rendering power (Render-only, which uses the FR-Render-H model but without FR) or only for display power

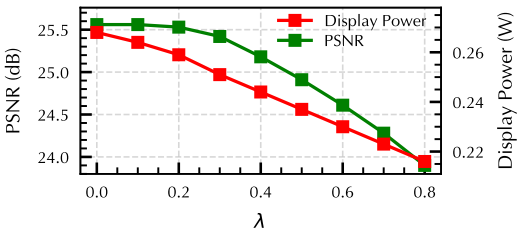


Fig. 10. λ vs. PSNR and Display Power on the `bicycle` scene. As λ increases, both PSNR and display power decrease but at different rates. At $\lambda \approx 0.2$, we observe a clear reduction in display power with only a minimal drop in PSNR.

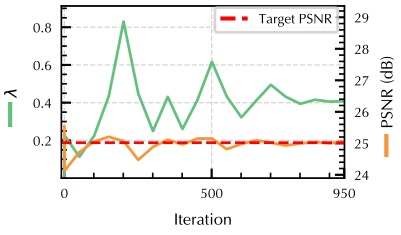


Fig. 11. The time course of how λ (Eqn. 5) and the model PSNR changes (`bicycle` scene; $\rho = 15\%$). By adaptively changing λ, our pruning method can converge to a given quality target.

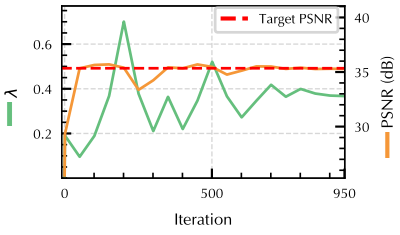


Fig. 12. Similar analysis for another scene (`lego`; $\rho = 65\%$). Our adaptive method generalizes across scenes, pruning ratios, and quality targets.

Table 2. Comparison of quality and power in non-foveated scenarios. The results are averaged over 9 scenes from the Mip-NeRF360 dataset.

Methods	SSIM ↑	PSNR (dB) ↑	LPIPS ↓	Total Power (W) ↓
Mini-Splat.-D	0.831	27.50	0.176	1.11 (2.06×)
3DGS	0.816	27.50	0.216	0.80 (1.48×)
Display-only	0.828	27.24	0.175	1.06 (1.96×)
Render-only	0.826	27.44	0.199	0.56 (1.04×)
PowerGS-H	0.825	27.35	0.201	**0.54 (1.00×)**

(Display-only, which uses FR-Display-H but without FR). Results are averaged across 9 scenes from the Mip-NeRF360 dataset.

All variants achieve similar quality. PowerGS achieves the lowest power consumption (up to 2.06× savings) due to rendering–display co-optimization. In non-foveated scenarios, however, the advantage of PowerGS over Render-only optimization is marginal (4%). This is because human vision is highly sensitive to foveal color, leaving little room for display power reduction.

6.7 λ vs. Quality and Display Power

As discussed in Eqn. 5, increasing λ reduces display power, but also affects rendering quality. To demonstrate this, we fine-tune Mini-Splatting-D using Eqn. 5 as the loss. We use different values of λ to obtain different variants. We do not apply pruning, so the rendering power remains approximately the same across variants.

The resulting trade-off is shown in Fig. 10. The x-axis shows λ and the y-axis reports both PSNR (left) and display power (right). We observe that as λ increases, both display power and PSNR decrease, confirming the intuition in Sec. 4.1 (below Eqn. 5). Interestingly, at $\lambda \approx 0.2$, display power drops significantly while PSNR remains nearly unchanged, yielding a favorable trade-off.

6.8 Effectiveness of Adaptively Weighting Display Power

To sample iso-quality models, PowerGS dynamically adjusts λ at a given ρ (Eqn. 5). This effectively reduces a 2D sampling of the $[\rho, \lambda]$ grid to sampling only along the ρ dimension. Fig. 11 shows the effectiveness of this approach (using the `bicycle` scene at $\rho = 15\%$) — in two aspects. First, we empirically confirm that as λ is being dynamically adjusted, the resulting model quality does gradually converge to the target PSNR. Second, it takes fewer than 1,000 iterations to converge, which is roughly the same as pruning under a given $[\rho, \lambda]$ pair; this suggests that our approaches roughly achieves a $N\times$ speedup over sampling N $[\rho, \lambda]$ pairs. Fig. 12 shows a similar time courses of another scene/ρ and conclusion generally holds.

7 Discussions and Future Work

In our particular display model, the blue channel consumes the highest power [Duinkharjav et al. 2022], so to save display power one reduces blue intensity, leading to the yellow-green-ish tint. This is evident from Fig. 13 (and additional examples in Supplementary Material D). This artifact is especially pronounced in the periphery where the point density is already low, so our joint optimization favors reducing display power rather than rendering power (which would require pruning even more points and degrade quality).

This artifact is generally unnoticeable, but can emerge in certain scenes (e.g., `bicycle` as shown in Sec. 6.2). The reason is twofold. First, the HVSQ metric we use for training different models in FR [Walton et al. 2021], while accounting for eccentricity-dependent visual perception, is still an approximation of the ventral metamerism [Freeman and Simoncelli 2011]. Second, like in any neural network training, the HVSQ loss does not become 0 during FR model training, which means perceptual differences always exist.

Our rendering power optimization focuses solely on pruning. Other techniques, such as compression [Fan et al. 2024; Niedermayr et al. 2024; Takikawa et al. 2022], variable rate shading [Nvidia 2018; Rolff et al. 2023], and temporal reuse [Feng et al. 2024a,b; Nvidia 2016] represent additional knobs, which we leave for future work.

8 Conclusion

PowerGS, for the first time, jointly optimizes the rendering and display power of a 3DGS model under a quality constraint. This generally intractable problem can be turned into a simple convex optimization by first sampling pruned models and then reconstructing iso-quality curves. PowerGS is readily extended to support foveated rendering to further reduce the total power. As XR devices are becoming increasingly computationally intensive but, at the same time, power-limited, our work is the first step, not the final ward, toward a holistic power optimization of XR rendering systems.

Acknowledgments

The work is partially supported by NSF Award #2225860 and a Meta research grant.

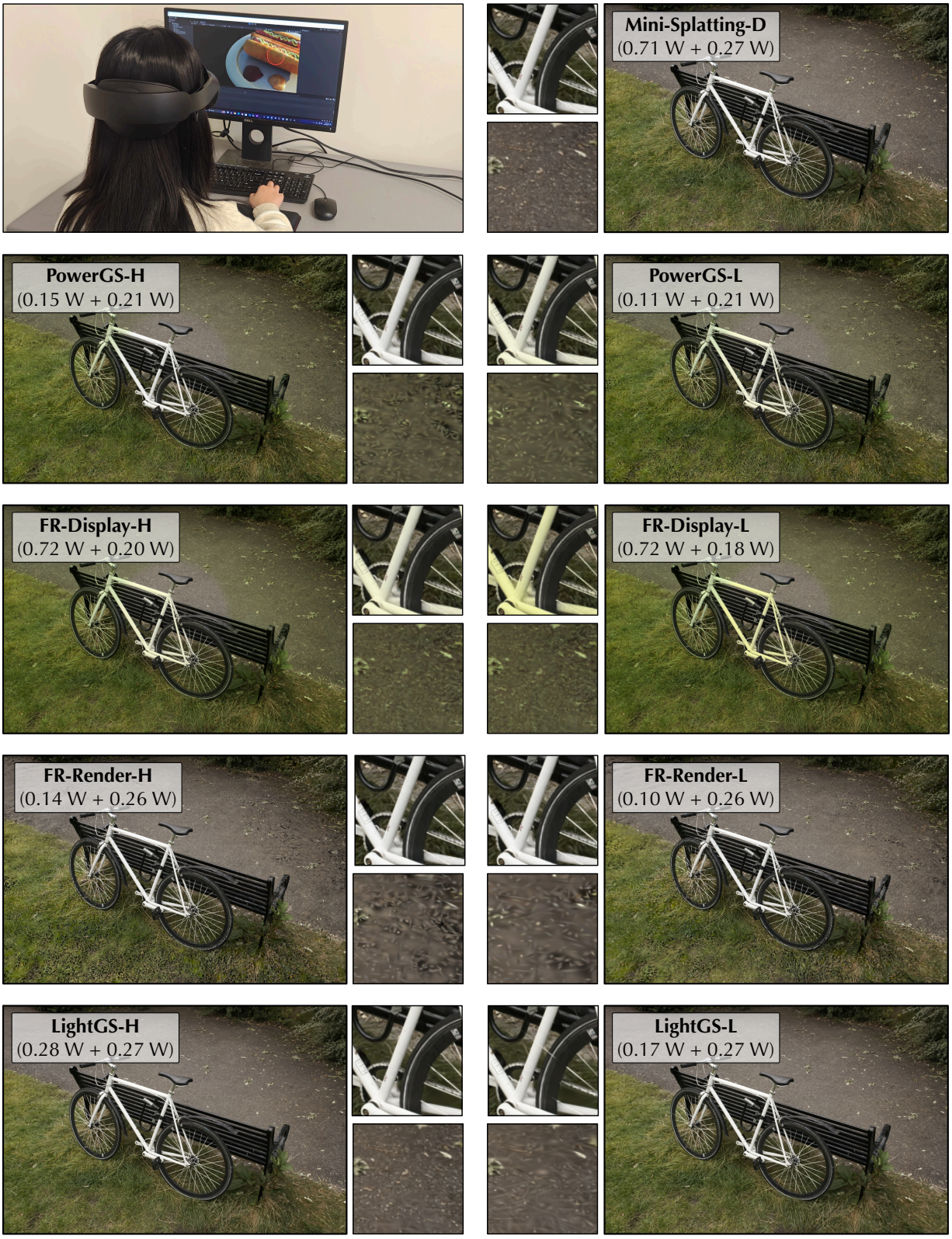


Fig. 13. A user study session (top left) and visual comparisons of our method and the baselines in the `bicycle` scene. The power consumptions reported are the rendering power + display power. The insets are the foveal (top) and peripheral (bottom) region.

References

Hyun-A Ahn, Yoo-Chang Sung, Yong-Hun Kim, Janghoo Kim, Kihan Kim, Dong-Hun Lee, Young-Gil Go, Jae-Woo Lee, Jae-Woo Jung, Yong-Hyun Kim, et al. 2024. A 1.01-V 8.5-Gb/s/pin 16-Gb LPDDR5x SDRAM With Advanced I/O Circuitry for High-Speed and Low-Power Applications. *IEEE Journal of Solid-State Circuits* (2024).

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. 2022. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5470–5479.

DA Baylor, AL Hodgkin, and TD Lamb. 1974. The electrical response of turtle cones to flashes and steps of light. *The Journal of Physiology* 242, 3 (1974), 685–727.

Philip Berne. 2024. *Exclusive: Up Close with Qualcomm Snapdragon AR2 Smart Glasses – and Why It's Taking So Long to Perfect Them.* https://www.techradar.com/computing/virtual-reality-augmented-reality/exclusive-up-close-with-qualcomm-snapdragon-ar2-smart-glasses-and-why-its-taking-so-long-to-perfect-them

Kenneth Chen, Thomas Wan, Nathan Matsuda, Ajit Ninan, Alexandre Chapiro, and Qi Sun. 2024. Pea-pods: Perceptual evaluation of algorithms for power optimization in xr displays. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–17.

Nianchen Deng, Zhenyi He, Jiannan Ye, Budmonde Duinkharjav, Praneeth Chakravarthula, Xubo Yang, and Qi Sun. 2022. Fov-nerf: Foveated neural radiance fields for virtual reality. *IEEE Transactions on Visualization and Computer Graphics* 28, 11 (2022), 3854–3864.

Zheng Dong, Ke Xu, Yaoan Gao, Hujun Bao, Weiwei Xu, and Rynson WH Lau. 2024. Gaussian Surfel Splatting for Live Human Performance Capture. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–17.

Yuanxing Duan, Fangyin Wei, Qiyu Dai, Yuhang He, Wenzheng Chen, and Baoquan Chen. 2024. 4d-rotor gaussian splatting: towards efficient novel view synthesis for dynamic scenes. In *ACM SIGGRAPH 2024 Conference Papers*. 1–11.

Daniel Duckworth, Peter Hedman, Christian Reiser, Peter Zhizhin, Jean-François Thibert, Mario Lučić, Richard Szeliski, and Jonathan T Barron. 2024. Smerf: Streamable memory efficient radiance fields for real-time large-scene exploration. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–13.

RCJ Dugdale. 1967. Nutrient limitation in the sea: Dynamics, identification, and significance 1. *Limnology and Oceanography* 12, 4 (1967), 685–695.

Budmonde Duinkharjav, Kenneth Chen, Abhishek Tyagi, Jiayi He, Yuhao Zhu, and Qi Sun. 2022. Color-perception-guided display power reduction for virtual reality. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–16.

Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. 2024. LightGaussian: Unbounded 3D Gaussian Compression with 15x Reduction and 200+ FPS. In *Advances in Neural Information Processing Systems*.

Guangchi Fang and Bing Wang. 2024. Mini-Splatting: Representing Scenes with a Constrained Number of Gaussians. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Yu Feng, Weikai Lin, Zihan Liu, Jingwen Leng, Minyi Guo, Han Zhao, Xiaofeng Hou, Jieru Zhao, and Yuhao Zhu. 2024a. Potamoi: Accelerating neural rendering via a unified streaming architecture. *ACM Transactions on Architecture and Code Optimization* 21, 4 (2024), 1–25.

Yu Feng, Zihan Liu, Jingwen Leng, Minyi Guo, and Yuhao Zhu. 2024b. Cicero: Addressing algorithmic and architectural bottlenecks in neural rendering by radiance warping and memory optimizations. In *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 1293–1308.

Linus Franke, Laura Fink, and Marc Stamminger. 2024. VR-Splatting: Foveated Radiance Field Rendering via 3D Gaussian Splatting and Neural Points. *arXiv preprint arXiv:2410.17932* (2024).

Jeremy Freeman and Eero P Simoncelli. 2011. Metamers of the ventral stream. *Nature neuroscience* 14, 9 (2011), 1195–1201.

Yonggan Fu, Zhifan Ye, Jiayi Yuan, Shunyao Zhang, Sixu Li, Haoran You, and Yingyan Lin. 2023. Gen-nerf: Efficient and generalizable neural radiance fields via algorithm-hardware co-design. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*. 1–12.

Guy Gafni, Justus Thies, Michael Zollhofer, and Matthias Nießner. 2021. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8649–8658.

Lin Gao, Jie Yang, Bo-Tao Zhang, Jia-Mu Sun, Yu-Jie Yuan, Hongbo Fu, and Yu-Kun Lai. 2024. Real-time large-scale deformation of Gaussian Splatting. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–17.

Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. 2024. Eagles: Efficient accelerated 3d gaussians with lightweight encodings. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

GooDisplay. [n. d.]. Product Specification of 0.39 inch OLED Display Micro-OLED Series. https://v4.cecdn.yun300.cn/100001_1909185148/GDOJ039FHP.pdf.

Markus Gross and Hanspeter Pfister. 2011. *Point-based graphics*. Elsevier.

Brian Guenter, Mark Finch, Steven Drucker, Desney Tan, and John Snyder. 2012. Foveated 3D graphics. *ACM transactions on Graphics (tOG)* 31, 6 (2012), 1–10.

Matthew Halpern, Yuhao Zhu, and Vijay Janapa Reddi. 2016. Mobile CPU's rise to power: Quantifying the impact of generational mobile CPU design trends on performance, energy, and user satisfaction. In *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 64–76.

Rehan Hameed, Wajahat Qadeer, Megan Wachs, Omid Azizi, Alex Solomatnikov, Benjamin C Lee, Stephen Richardson, Christos Kozyrakis, and Mark Horowitz. 2010. Understanding sources of inefficiency in general-purpose chips. In *Proceedings of the 37th annual international symposium on Computer architecture*. 37–47.

Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. 2018. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)* 37, 6 (2018), 1–15.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2024. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*. 1–11.

Yuheng Jiang, Zhehao Shen, Yu Hong, Chengcheng Guo, Yize Wu, Yingliang Zhang, Jingyi Yu, and Lan Xu. 2024. Robust Dual Gaussian Splatting for Immersive Human-centric Volumetric Videos. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–15.

Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. 2017. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*. 1–12.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.* 42, 4 (2023), 139–1.

Bernhard Kerbl, Andreas Meuleman, Georgios Kopanas, Michael Wimmer, Alexandre Lanvin, and George Drettakis. 2024. A hierarchical 3d gaussian representation for real-time rendering of very large datasets. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–15.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. 2017. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)* 36, 4 (2017), 1–13.

George Alex Koulieris, Kaan Akşit, Michael Stengel, Rafał K Mantiuk, Katerina Mania, and Christian Richardt. 2019. Near-eye display and tracking technologies for virtual and augmented reality. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 493–519.

Junseo Lee, Seokwon Lee, Jungi Lee, Junyong Park, and Jaewoong Sim. 2024a. GSCore: Efficient Radiance Field Rendering via Architectural Support for 3D Gaussian Splatting. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*. 497–511.

Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. 2024b. Compact 3d gaussian representation for radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21719–21728.

Yue Leng, Chi-Chun Chen, Qiuyue Sun, Jian Huang, and Yuhao Zhu. 2019. Energy-efficient video processing for virtual reality. In *Proceedings of the 46th International Symposium on Computer Architecture*. 91–103.

Deqi Li, Shi-Sheng Huang, Zhiyuan Lu, Xinran Duan, and Hua Huang. 2024. St-4dgs: Spatial-temporally consistent 4d gaussian splatting for efficient dynamic scene rendering. In *ACM SIGGRAPH 2024 Conference Papers*. 1–11.

Shijun Liang, Haofei Wang, and Feng Lu. 2024. EyeIR: Single Eye Image Inverse Rendering In the Wild. In *ACM SIGGRAPH 2024 Conference Papers*. 1–11.

Robert LiKamWa, Zhen Wang, Aaron Carroll, Felix Xiaozhu Lin, and Lin Zhong. 2014. Draining our glass: An energy and heat characterization of google glass. In *Proceedings of 5th Asia-Pacific Workshop on Systems*. 1–7.

Weikai Lin, Yu Feng, and Yuhao Zhu. 2024. RTGS: Enabling Real-Time Gaussian Splatting on Mobile Devices Using Efficiency-Guided Pruning and Foveated Rendering. *arXiv preprint arXiv:2407.00435* (2024).

Weikai Lin, Yu Feng, and Yuhao Zhu. 2025. MetaSapiens: Real-Time Neural Rendering with Efficiency-Aware Pruning and Accelerated Foveated Rendering. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*. 669–682.

Haimin Luo, Min Ouyang, Zijun Zhao, Suyi Jiang, Longwen Zhang, Qixuan Zhang, Wei Yang, Lan Xu, and Jingyi Yu. 2024. Gaussianhair: Hair modeling and rendering with light-aware gaussians. *arXiv preprint arXiv:2402.10483* (2024).

Xiaoyang Lyu, Yang-Tian Sun, Yi-Hua Huang, Xiuzhe Wu, Ziyi Yang, Yilun Chen, Jiangmiao Pang, and Xiaojuan Qi. 2024. 3dgsr: Implicit surface reconstruction with 3d gaussian splatting. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–12.

David L MacAdam. 1942. Visual sensitivities to color differences in daylight. *Josa* 32, 5 (1942), 247–274.

Paul McLellan. 2019. *HOT CHIPS: Microsoft Hololens 2*. https://community.cadence.com/cadence_blogs_8/b/breakfast-bytes/posts/holo2

Meta. 2022. Meta Quest Pro Specifications. https://vr-compare.com/headset/metaquestpro.

Meta. 2024. Fixed foveated rendering (FFR). https://developers.meta.com/horizon/documentation/unity/os-fixed-foveated-rendering/.

Meta. 2024. Orion. https://about.meta.com/realitylabs/orion/.

Leonor Michaelis and Maud L Menten. 1913. Die kinetik der invertinwirkung. *Biochem. z* 49, 333-369 (1913), 352.

Microsoft. 2019. Hololens. https://learn.microsoft.com/en-us/hololens/.

B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. 2020. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*.

Nicolas Moenne-Loccoz, Ashkan Mirzaei, Or Perel, Riccardo de Lutio, Janick Martinez Esturo, Gavriel State, Sanja Fidler, Nicholas Sharp, and Zan Gojcic. 2024. 3D Gaussian Ray Tracing: Fast Tracing of Particle Scenes. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–19.

Felix Mujkanovic, Ntumba Elie Nsampi, Christian Theobalt, Hans-Peter Seidel, and Thomas Leimkühler. 2024. Neural Gaussian Scale-Space Fields. *ACM Transactions on Graphics (TOG)* 43, 4 (2024).

Sean Murray, William Floyd-Jones, Ying Qi, George Konidaris, and Daniel J Sorin. 2016. The microarchitecture of a real-time robot motion planning accelerator. In *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 1–12.

Simon Niedermayr, Josef Stumpfegger, and Rüdiger Westermann. 2024. Compressed 3d gaussian splatting for accelerated novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10349–10358.

Nvidia. 2016. VRWorks – Context Priority. https://developer.nvidia.com/vrworks/headset/contextpriority. Accessed: 2025-05-12.

NVIDIA. 2018. NVIDIA Reveals Xavier SOC Details. https://www.forbes.com/sites/moorinsights/2018/08/24/nvidia-reveals-xavier-soc-details/amp/

Nvidia. 2018. Variable Rate Shading (VRS). https://developer.nvidia.com/vrworks/graphics/variablerateshading.

Anjul Patney, Marco Salvi, Joohwan Kim, Anton Kaplanyan, Chris Wyman, Nir Benty, David Luebke, and Aaron Lefohn. 2016. Towards foveated rendering for gaze-tracked virtual reality. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 1–12.

Zhexi Peng, Tianjia Shao, Yong Liu, Jingke Zhou, Yin Yang, Jingdong Wang, and Kun Zhou. 2024. Rtg-slam: Real-time 3d reconstruction at scale using gaussian splatting. In *ACM SIGGRAPH 2024 Conference Papers*. 1–11.

Maria Perez-Ortiz, Aliaksei Mikhailiuk, Emin Zerman, Vedad Hulusic, Giuseppe Valenzise, and Rafał K Mantiuk. 2019. From pairwise comparisons and rating to a unified quality scale. *IEEE Transactions on Image Processing* 29 (2019), 1139–1151.

Philip Berne. 2023. Up close with Qualcomm Snapdragon AR2 smart glasses and why it's taking so long to perfect them. https://www.techradar.com/computing/virtual-reality-augmented-reality/exclusive-up-close-with-qualcomm-snapdragon-ar2-smart-glasses-and-why-its-taking-so-long-to-perfect-them.

Wajahat Qadeer, Rehan Hameed, Ofer Shacham, Preethi Venkatesan, Christos Kozyrakis, and Mark A Horowitz. 2013. Convolution engine: balancing efficiency & flexibility in specialized computing. In *Proceedings of the 40th Annual International Symposium on Computer Architecture*. 24–35.

Qualcomm. 2022. Snapdragon XR2+ Gen 1 Platform. https://www.qualcomm.com/products/mobile/snapdragon/xr-vr-ar/snapdragon-xr2-plus-gen-1-platform.

Lukas Radl, Michael Steiner, Mathias Parger, Alexander Weinrauch, Bernhard Kerbl, and Markus Steinberger. 2024. Stopthepop: Sorted gaussian splatting for view-consistent real-time rendering. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–17.

Parthasarathy Ranganathan, Daniel Stodolsky, Jeff Calow, Jeremy Dorfman, Marisabel Guevara, Clinton Wills Smullen IV, Aki Kuusela, Raghu Balasubramanian, Sandeep Bhatia, Prakash Chauhan, et al. 2021. Warehouse-scale video acceleration: co-design and deployment in the wild. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*. 600–615.

Pramod Rao, Gereon Fox, Abhimitra Meka, Mallikarjun BR, Fangneng Zhan, Tim Weyrich, Bernd Bickel, Hanspeter Pfister, Wojciech Matusik, Mohamed Elgharib, et al. 2024. Lite2Relight: 3D-aware Single Image Portrait Relighting. In *ACM SIGGRAPH 2024 Conference Papers*. 1–12.

Tim Rolff, Susanne Schmidt, Ke Li, Frank Steinicke, and Simone Frintrop. 2023. VRS-NeRF: Accelerating Neural Radiance Field Rendering with Variable Rate Shading. In *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 243–252.

Shunsuke Saito, Gabriel Schwartz, Tomas Simon, Junxuan Li, and Giljoo Nam. 2024. Relightable gaussian codec avatars. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 130–141.

Fred Schlachter. 2013. No Moore's Law for batteries. *Proceedings of the National Academy of Sciences* 110, 14 (2013), 5273–5273.

David M Schneeweis and Julie L Schnapf. 1995. Photovoltage of rods and cones in the macaque retina. *Science* 268, 5213 (1995), 1053–1056.

Xuehuai Shi, Lili Wang, Xinda Liu, Jian Wu, and Zhiwen Shao. 2024. Scene-aware Foveated Neural Radiance Fields. *IEEE Transactions on Visualization and Computer Graphics* (2024).

Alex Shye, Benjamin Scholbrock, and Gokhan Memik. 2009. Into the wild: studying real user activity patterns to guide power optimizations for mobile architectures. In *Proceedings of the 42nd annual IEEE/ACM international symposium on microarchitecture*. 168–178.

Q. Sun, F.-C. Huang, J. Kim, L.-Y. Wei, D. Luebke, and A. Kaufman. 2017. Perceptually-Guided Foveation for Light Field Displays. *ACM Transactions on Graphics* 36, 6 (Nov. 2017).

Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. 2020. *Efficient processing of deep neural networks*. Springer.

Towaki Takikawa, Alex Evans, Jonathan Tremblay, Thomas Müller, Morgan McGuire, Alec Jacobson, and Sanja Fidler. 2022. Variable bitrate neural fields. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.

Xiaochun Tong and Toshiya Hachisuka. 2024. Efficient Image-Space Shape Splatting for Monte Carlo Rendering. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–11.

Yvan Tortorella, Luca Bertaccini, Davide Rossi, Luca Benini, and Francesco Conti. 2022. RedMulE: A compact FP16 matrix-multiplication accelerator for adaptive deep learning on RISC-V-based ultra-low-power SoCs. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 1099–1102.

Nisarg Ujjainkar, Ethan Shahan, Kenneth Chen, Budmonde Duinkharjav, Qi Sun, and Yuhao Zhu. 2024. Exploiting human color discrimination for memory-and energy-efficient image encoding in virtual reality. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*. 166–180.

David R. Walton, Rafael Kuffner Dos Anjos, Sebastian Friston, David Swapp, Kaan Akşit, Anthony Steed, and Tobias Ritschel. 2021. Beyond Blur: Ventral Metamers for Foveated Rendering. *ACM Trans. Graph. (Proc. SIGGRAPH 2021)* 40, 4 (2021).

David R Walton, Koray Kavaklı, Rafael Kuffner Dos Anjos, David Swapp, Tim Weyrich, Hakan Urey, Anthony Steed, Tobias Ritschel, and Kaan Akşit. 2022. Metameric varifocal holograms. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 746–755.

Zishen Wan, Bo Yu, Thomas Yuang Li, Jie Tang, Yuhao Zhu, Yu Wang, Arijit Raychowdhury, and Shaoshan Liu. 2021. A survey of fpga-based robotic computing. *IEEE Circuits and Systems Magazine* 21, 2 (2021), 48–74.

Brian A. Wandell. 1995. *Foundations of vision*. Sinauer Associates.

Penghao Wang, Zhirui Zhang, Liao Wang, Kaixin Yao, Siyuan Xie, Jingyi Yu, Minye Wu, and Lan Xu. 2024a. V^ 3: Viewing Volumetric Videos on Mobiles via Streamable 2D Dynamic Gaussians. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–13.

Yue Wang, Yan Zhang, Xuanhui Yang, Hui Wang, Dongxu Liu, and Xubo Yang. 2024b. Foveated fluid animation in virtual reality. In *2024 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*. IEEE, 535–545.

XREAL. 2023. XREAL Air 2. https://us.shop.xreal.com/products/xreal-air-2.

Linning Xu, Vasu Agrawal, William Laney, Tony Garcia, Aayush Bansal, Changil Kim, Samuel Rota Bulò, Lorenzo Porzi, Peter Kontschieder, Aljaž Božič, et al. 2023. VR-NeRF: High-fidelity virtualized walkable spaces. In *SIGGRAPH Asia 2023 Conference Papers*. 1–12.

Zhisheng Yan, Chen Song, Feng Lin, and Wenyao Xu. 2018. Exploring eye adaptation in head-mounted display for energy efficient smartphone virtual reality. In *Proceedings of the 19th International Workshop on Mobile Computing Systems & Applications*. 13–18.

Chen Yang, Sikuang Li, Jiemin Fang, Ruofan Liang, Lingxi Xie, Xiaopeng Zhang, Wei Shen, and Qi Tian. 2024a. Gaussianobject: High-quality 3d object reconstruction from four views with gaussian splatting. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–13.

Haichuan Yang, Yuhao Zhu, and Ji Liu. 2018. Energy-constrained compression for deep neural networks via weighted sparse projection and layer input masking. *arXiv preprint arXiv:1806.04321* (2018).

Tien-Ju Yang, Yu-Hsin Chen, and Vivienne Sze. 2017. Designing energy-efficient convolutional neural networks using energy-aware pruning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5687–5695.

Zongyuan Yang, Baolin Liu, Yingde Song, Lan Yi, Yongping Xiong, Zhaohe Zhang, and Xunbo Yu. 2024b. DirectL: Efficient Radiance Fields Rendering for 3D Light Field Displays. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–19.

Jiannan Ye, Anqi Xie, Susmija Jabbireddy, Yunchuan Li, Xubo Yang, and Xiaoxu Meng. 2022. Rectangular mapping-based foveated rendering. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 756–764.

Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. 2024a. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19447–19456.

Zehao Yu, Torsten Sattler, and Andreas Geiger. 2024b. Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–13.

Yan Zhang, Keyao You, Xiaodan Hu, Hangyu Zhou, Kiyoshi Kiyokawa, and Xubo Yang. 2024. Retinotopic foveated rendering. In *2024 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*. IEEE, 903–912.