SAGE: Semantic-Driven Adaptive Gaussian Splatting in Extended Reality

Chiara Schiavo, Elena Camuffo, Leonardo Badia and Simone Milani
Dept. of Information Engineering, University of Padova, Padua, Italy
Emails: {chiara.schiavo, elena.camuffo, leonardo.badia, simone.milani}@unipd.it

***Abstract*—3D Gaussian Splatting (3DGS) has significantly improved the efficiency and realism of three-dimensional scene visualization in several applications, ranging from robotics to eXtended Reality (XR). This work presents SAGE (Semantic-Driven Adaptive Gaussian Splatting in Extended Reality), a novel framework designed to enhance the user experience by dynamically adapting the Level of Detail (LOD) of different 3DGS objects identified via semantic segmentation. Experimental results demonstrate how SAGE effectively reduces memory and computational overhead while keeping a desired target visual quality, thus providing a powerful optimization for interactive XR applications.**

***Index Terms*—Extended Reality, Quality Adaptation, Gaussian Splatting**

I. Introduction

The rapid evolution of eXtended Reality technologies on mobile and wearable platforms has increased the demand for efficient 3D rendering techniques that provide highly immersive and fluid user experiences. However, balancing computational efficiency and visual quality introduces several challenges, particularly in resource-constrained environments [1], [2]. Traditional approaches often rely on geometric simplification or predefined Level of Detail strategies, which are chosen according to user proximity and interaction [3], [4]. Previous research has focused on adapting cognitive load [5], predicting user actions to optimize training experiences [6], or minimizing transmitted information [7], while only a few recent attempts have been made using Deep Neural Networks [8]. The advent of neural representation techniques, such as Neural Radiance Fields (NeRF) [9] and 3D Gaussian Splatting [10], has transformed 3D scene rendering. These methods enable implicit scene representations that maintain high fidelity while offering flexibility for on-demand rendering. While NeRF emphasizes detailed and photorealistic scenes using a single neural network, 3DGS leverages Gaussian-shaped primitives for lightweight yet differentiable scene management and rendering.

In this paper, we introduce SAGE (Semantically Adaptive Gaussian Splatting in Extended Reality), a novel approach that integrates semantic information in the optimization process of 3DGS. By performing semantic segmentation, SAGE dynamically adjusts the 3DGS representation quality of individual scene components based on their spatial and visual importance, using a quality prediction model to estimate the optimal number of training iterations per region. This method leads to a reduction in memory and computational overhead while maintaining high visual quality. Through extensive evaluations on different scenes of the Mip-NeRF360 dataset [11], we demonstrate SAGE's ability to significantly enhance efficiency and scalability in 3DGS rendering. This ability proves to be extremely suitable in XR applications, where rendering complex scenes in real-time is essential, and selecting quality based on scene semantics can be beneficial to ensure a better user experience. SAGE is designed to optimize resource allocation rather than maximize perceptual quality alone, maintaining comparable visual quality with significantly lower memory usage compared to other standard approaches.

II. Related Work

The problem of quality maximization under bandwidth constraints first appeared in image, audio and video compression [13]. Later, several techniques in networking were proposed for HTTP Adaptive Streaming (HAS) to modulate the data stream while minimizing the quality decrement [14], [15]. Recently, deep learning solutions were investigated to address this problem in the video domain [16] and allowed the extension to XR applications. Whenever referring to XR applications, visual quality does not rely only on the modeling accuracy for the synthetic 3D objects but also on the rendering complexity and smoothness as viewpoint changes.

With the advent of 3DGS [10] as a powerful technique for real-time rendering, it becomes possible to represent 3D scenes via Gaussian primitives in a more flexible manner, even if the representation of high-quality scenes remains computationally demanding due to the large number of Gaussians required. To address these issues, several techniques have been proposed so far. LightGaussians [17] and Compact3D [18] use pruning and quantization to reduce memory, while Multi-Scale 3DGS [19] introduces multi-scale representations to enhance fidelity and mitigate aliasing. OctreeGS [20] employ hierarchical structures to optimize memory and computation, while FlOD [21] applies a Level of Detail strategy for real-time rendering based on device constraints.

Despite these improvements, experimental results have shown that tayloring the computational effort to the specific objects allows a better complexity saving while preserving the visual quality. To this purpose, SAGE identifies the

This work was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) Mission 4, Component 2, Investment 1.3, CUP C93C22005250001, partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART") and by the Italian PRIN project 2022PNRR "DIGIT4CIRCLE" (project code P2022788KK).

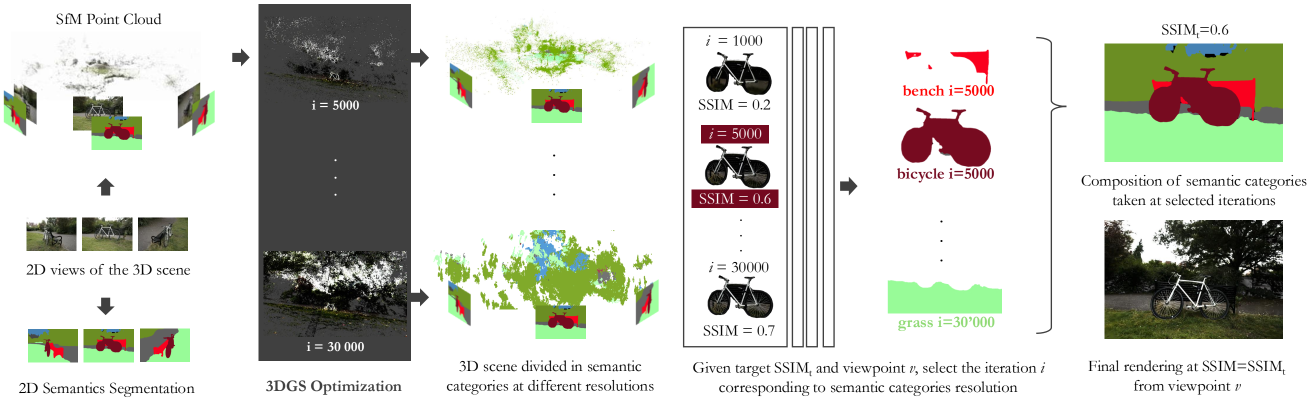
SfM Point Cloud
2D views of the 3D scene
2D Semantics Segmentation
i = 5000
i = 30 000
3DGS Optimization
3D scene divided in semantic categories at different resolutions
i = 1000
SSIM = 0.2
i = 5000
SSIM = 0.6
i = 30000
SSIM = 0.7
Given target SSIM_t and viewpoint v, select the iteration i corresponding to semantic categories resolution
bench i=5000
bicycle i=5000
grass i=30'000
SSIM_t=0.6
Composition of semantic categories taken at selected iterations
Final rendering at SSIM=SSIM_t from viewpoint v

Fig. 1. SAGE pipeline. Starting from a set of 2D views V, SAGE retrieves the 2D semantics using DeepLabV2 [12]. In parallel, it constructs the Structure from Motion point cloud, like in standard 3DGS. Then it processes the SfM point cloud at increasing resolution with proceeding iteration i. Differently from standard 3DGS, SAGE follows the semantic masks provided on 2D views to partition the 3D point cloud and perform selective optimization of different semantic categories. By setting a target quality (SSIM_t) the optimization of each semantic category stops the optimization process when such target value is achieved. The final render from selected viewpoint v is obtained as a composition of the scene categories optimized separately for target quality.

different objects by means of sematic segmentation, which has also been investigated in 3D models reconstruction [22] or transmission [23]. Recently, a few attempts have also been made to integrate semantic understanding tasks within 3DGS [24] to jointly improve understanding and reconstruction.

III. Methodology

Preliminaries: 3D Gaussian Splatting [10] is an explicit radiance field technique for efficient, high-quality rendering. It represents a scene using differentiable 3D Gaussian primitives optimized to fit the scene's geometry, capturing shapes and features in a compact, expressive form. Given a set of images, 3DGS is initialized from a point cloud generated by a Structure from Motion (SfM) algorithm, and then iteratively refined through adaptive point densification and pruning, enabling high-quality rendering.

Proposed Approach: SAGE utilizes semantic segmentation to increase the 3DGS optimization performance on individual objects and develop an adaptive quality system. Semantic segmentation is applied to the input 2D views [12] assigning a label $l \in L$ to each pixel in every image $v \in V$. These labels are then related to 3D points in the SfM reconstruction via the estimated camera parameters, using a majority voting scheme that resolves conflicts by assigning each 3D point the most frequent label across its projections.

To perform the optimization, we iteratively compute predefined quality metrics, such as PSNR and SSIM, for each fixed viewpoint (rendered image) v and level of reconstruction i, restricted to every semantic label l separately. These quality metrics are computed with respect to $d_{min,l,v}$, which represents the distance of the closest point assigned to semantic label l with respect to the camera position, to be conservative in the optimization process. Formally, given a fixed viewpoint $v = \omega$, the quality-constrained optimization problem is designed as follows:

$$\min_i \sum_l N_l(i) \quad \text{s.t.} \quad \text{SSIM}_{l,i}(d_{min,l}) \geq \text{SSIM}_t, \ \forall l \in L \tag{1}$$

where i represents the minimum iteration of the 3DGS algorithm to ensure the desired quality (with SSIM_t denoting the target SSIM value) is met for the semantic category l, while minimizing the number of Gaussian primitives used. $N_l(i)$ denotes the number of Gaussians representing category l at iteration i. The quality at iteration i is represented by $\text{SSIM}_{l,i}$ and depends on the distance between the position of the camera and the closest 3D point to the camera, with label l, *i.e.*, $d_{min,l} = d_{min,l,v}|_{v=\omega}$ with a fixed viewpoint. As a result, a parametric model can be fit for a target SSIM_t:

$$\text{SSIM}_{l,i}(d_{min,l}) = \begin{cases} K_1 \cdot e^{-\gamma_1 |d_{min,l} - \mu_1|^{\alpha_1}} & \text{if } d_{min,l} < \beta, \\ K_2 \cdot e^{-\gamma_2 |d_{min,l} - \mu_2|^{\alpha_2}} & \text{if } d_{min,l} \geq \beta; \end{cases} \tag{2}$$

where the coefficients K_n, γ_n, μ_n, and α_n (with $n = 1, 2$) and the threshold β that separates the two distance regimes are obtained by fitting the equation to the desired label l at the level of reconstruction detail i (see Sec. IV and Fig. 3 for clarity). This way, the model is able to predict for a target class l at which iteration i the 3DGS algorithm should be halted to meet the desired quality SSIM_t. The operation is iterated on all viewpoints $v \in V$, obtaining distance-dependent fitting. An overall representation of SAGE is shown in Fig. 1.

Note that this technique is efficient in terms of computational efficiency as it reallocates computational resources based on semantic importance, while keeping visual quality sufficiently high. This makes the semantic-based optimization of SAGE applicable to many rendering systems, in addition to 3DGS.

Evaluation methodologies: In order to measure both the adaptability and robustness of the proposed approach, two types of evaluation tests were performed: (i) cross-view and (ii) cross-scene. The (i) cross-view test focuses on synthesizing a novel view starting from existing views of a single 3D scene using the model for fitting different labels. The 3D scene is partitioned into class-labeled 3D points by leveraging the segmentation of 2D views and projecting onto the 3D point cloud. Semantic masks are extracted by projecting back the semantics from the 3D points to 2D onto the novel view, and

TABLE I
AVERAGE RESULTS OF SAGE FOR SCENE "BICYCLE". NUMBERS ON THE HEADER DENOTE 3DGS ITERATION COUNTS.

	5 000		10 000		15 000		30 000	
	SSIM	# Gauss.	SSIM	# Gauss.	SSIM	# Gauss.	SSIM	# Gauss.
Bench	0.598	141k	0.635	234k	0.659	334k	0.682	336k
Bicycle	0.602	50k	0.658	111k	0.674	144k	0.688	146k
Grass-merged	0.438	287k	0.491	703k	0.521	972k	0.537	985k
Pavement-merged	0.585	163k	0.625	337k	0.642	422k	0.651	424k
Sky-other-merged	0.917	278k	0.920	456k	0.923	587k	0.923	578k
Tree-merged	0.540	1.4M	0.574	2.5M	0.594	3.2M	0.613	3.3M
Total	0.546	2.3M	0.598	4.3M	0.627	5.7M	0.647	5.8M

TABLE II
AVERAGE RESULTS OF SAGE FOR SCENE "GARDEN". NUMBERS ON THE HEADER DENOTE 3DGS ITERATION COUNTS.

	5 000		10 000		15 000		30 000	
	SSIM	# Gauss.	SSIM	# Gauss.	SSIM	# Gauss.	SSIM	# Gauss.
Dining table	0.828	211k	0.869	309k	0.879	379k	0.880	379k
Grass-merged	0.691	742k	0.776	1.1M	0.798	1.3M	0.804	1.3M
Pavement-merged	0.753	447k	0.805	682k	0.816	736k	0.818	736k
Potted plant	0.765	56k	0.808	82k	0.827	94k	0.829	94k
Tree-merged	0.688	1.2M	0.739	2.0M	0.749	2.3M	0.757	2.3M
Vase	0.847	14k	0.892	17k	0.902	18k	0.906	18k
Total	0.778	3.1M	0.826	4.9M	0.841	5.6M	0.848	5.6M

the model is optimized on the novel view according to the desired visual quality. Once the new view has been synthesized, the SSIM is evaluated on two levels: first, for the entire view to check overall image quality, and second, for each semantic mask to verify that the prediction is correct. The (ii) cross-scene test examines whether a model trained on a semantic category (*e.g.*, *grass-merged* from "bicycle" scene) generalizes to the same category in another scene (*e.g.*, *grass-merged* from "garden" scene). If the SSIM curve remains consistent, the learned parameters and iteration values are transferable, enabling a general optimization routine applicable to unseen data.

IV. EXPERIMENTAL RESULTS

Experiments were conducted on the Mip-NeRF360 dataset [11] considering PSNR, SSIM, Learned Perceptual Image Patch Similarity (LPIPS) [25] as quality metrics, and the number of Gaussians (# Gauss.) to characterize the quality and the complexity of the scene. SAGE was implemented on top of the original 3DGS framework [10], with point clouds of Gaussians saved and analyzed at intervals of 1000 iterations throughout the optimization process. Table I presents the cross-view results for scene "bicycle" averaged across all training images. The reported per-label evaluations allow us to verify that there exist significant class-dependent variations: *sky-other-merged*[1] class consistently exhibits high SSIM values (above 0.8) even at early iterations, while for others, SSIM values improve progressively. The overall behavior of SSIM versus iterations can be seen in Fig. 2 for scene "bicycle". The category-related quality appears to be related to the variation of the content behavior within the same semantic category (*e.g.*, *sky-other-merged* is quite uniform with low-frequency content while *grass-merged* presents different textures depending on the distance from the viewpoint). Visual quality is also affected by the distance of each class from the camera. Using this model, we created an iteration predictor capable of estimating the level of detail necessary for each class to achieve the target SSIM$_t$. Table II reports the SSIM values for different iterations for scene "garden". Considerations similar to "bicycle" can be made. The number of Gaussians and SSIM values evolve differently for different semantic categories over time, but interestingly, as for classes *grass-merged* and *pavement-merged* that appear both in "bicycle" and "garden", SSIM metric increases proportionally with the number of Gaussians over time. A cross-scene evaluation for classes *grass-merged*, *pavement-merged* and *tree-merged* (see highlighted rows in Tables I and II) showed that the model, trained on "bicycle", could generalize to "garden" with reasonable success. The SSIM values remained consistent across both scenes, indicating that SAGE can predict the appropriate iteration to achieve the target SSIM$_t$ in the new scene. However, some variations were observed in the prediction of the optimal iteration, suggesting that scene-dependent factors affect the precision of the model.

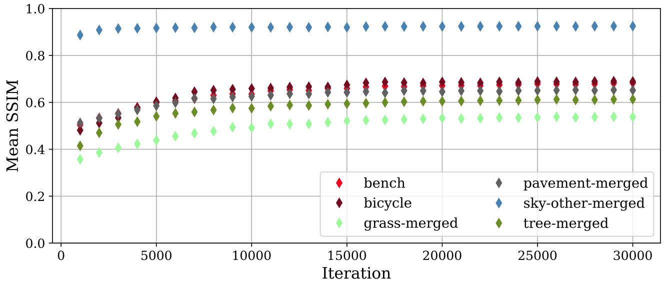

Fig. 2. Mean SSIM over training iterations for individual scene components of scene "bicycle". Variations in optimization performance across semantic categories are visible. Highly textured content (*e.g.*, *grass-merged*) show lower overall quality compared to smooth areas (*e.g.*, *sky-other-merged*).

Fig. 3 proves the model's predictive power, enabling adaptive reconstruction of scenes by integrating elements with varying LOD. Three targets SSIM are selected for SAGE evaluation (*i.e.*, 0.5, 0.6, 0.7 for scene "bicycle") and tested on both the original training views and the novel synthesized perspectives, robustly assessing our method's capabilities. Our results show that the SSIM follows a distance-dependent trend, where closer objects undergo sharper SSIM variations. The proposed two-phase decay model (Eq. 2) captures this effect, prioritizing foreground detail while preventing excessive refinement of background content. We use intermediate steps of 3DGS optimization rather than the final optimized state to avoid fully optimizing 3DGS and account for memory and processing constraints in real-world XR applications. Halting the optimization at different points per object reduces redundancy while maintaining high perceptual quality. Table III highlights SAGE's performance for a specific view (*i.e.*, "`DSC8719`"), showcasing the trade-offs between visual fidelity and resource efficiency. Selected

[1] *-merged* notation refers to labels which comprehend multiple related COCO [26] classes mapped to a single class, following the standards of [27].

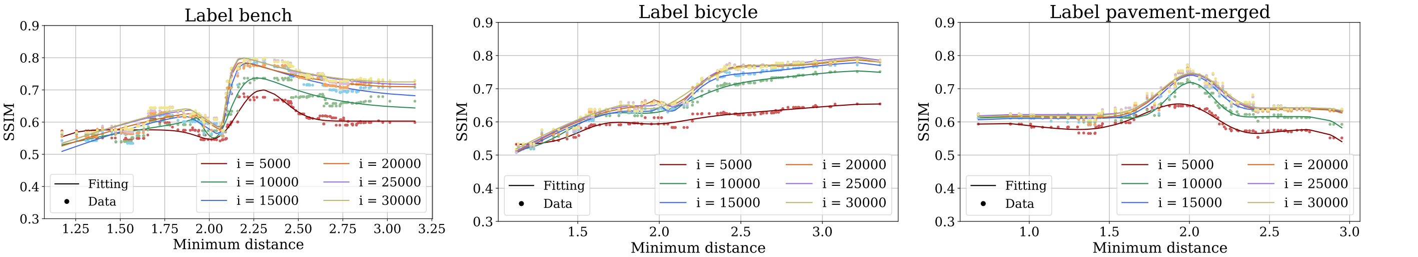

Fig. 3. SSIM as a function of the minimum distance for the semantic labels *bench*, *bicycle* and *pavement-merged*. Each curve represents data collected at a different iteration i of 3DGS, with experimental data (dots) and fitted trends (lines). The SSIM generally increases with distance, reaching a peak before stabilizing or declining, with higher iterations showing improved reconstruction quality and smoother trends.

TABLE III
QUANTITATIVE RESULTS ON SINGLE VIEW "DSC8719" OF SCENE "BICYCLE". HEADERS IN SSIM SECTION DENOTE 3DGS ITERATION COUNTS.

	Distance		SSIM_i				# Gaussians				Occupancy ↓			
	Min	Avg	5 000	10 000	15 000	30 000 (3DGS)	3DGS	$\text{SAGE}_{t=0.5}$	$\text{SAGE}_{t=0.6}$	$\text{SAGE}_{t=0.7}$	3DGS	$\text{SAGE}_{t=0.5}$	$\text{SAGE}_{t=0.6}$	$\text{SAGE}_{t=0.7}$
Bench	2.615	4.621	0.587	0.697	0.742	0.750	336 632	141 804	141 804	334 433	83.5 MB	35.2 MB	35.2 MB	82.9 MB
Bicycle	3.046	4.358	0.632	0.713	0.759	0.758	146 103	50 965	50 965	111 799	36.2 MB	12.6 MB	12.6 MB	27.7 MB
Grass-merged	1.609	6.809	0.341	0.393	0.434	0.419	985 863	985 863	985 863	985 863	224.5 MB	224.5 MB	224.5 MB	224.5 MB
Pavement-merged	2.671	6.335	0.586	0.662	0.688	0.693	424 146	163 298	337 966	424 124	105.2 MB	40.5 MB	83.8 MB	105.2 MB
Sky-other-merged	8.980	27.192	0.805	0.794	0.795	0.803	578 378	278 139	278 139	278 139	143.4 MB	69.0 MB	69.0 MB	69.0 MB
Tree-merged	0.753	16.424	0.595	0.612	0.642	0.634	3 343 641	1 428 984	3 343 635	3 343 641	829.2 MB	354.4 MB	829.2 MB	829.2 MB
Total	—	—	0.531	0.579	0.615	0.617	5 832 994	3 049 053	5 138 872	5 477 999	1.45 GB	756.2 MB	1.27 GB	1.36 GB

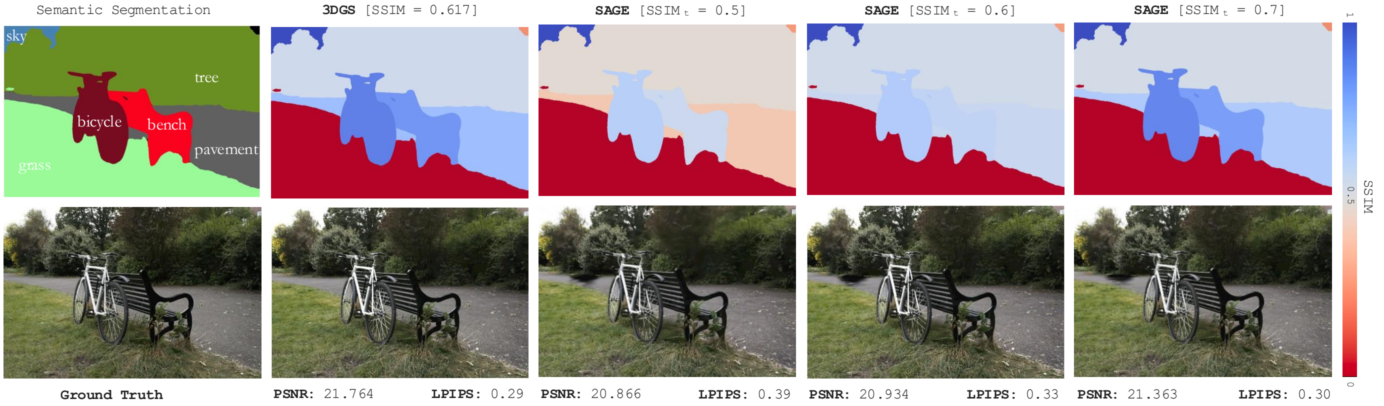

Fig. 4. Qualitative results on view "DSC8719" of scene "bicycle".

TABLE IV
3DGS AND SAGE COMPARISON AT FIXED OCCUPANCY IN "BICYCLE".

	3DGS			SAGE		
Occupancy	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓
$\sim$ 700 MB	0.533	22.057	0.38	0.557	22.497	0.27
$\sim$ 1, 25 GB	0.599	22.594	0.24	0.581	22.669	0.24
$\sim$ 1, 3 GB	0.612	22.629	0.23	0.618	22.816	0.19

iterations for each semantic category at each target SSIM_t are highlighted for $t = 0.5, t = 0.6, t = 0.7$ in green, yellow, and orange, respectively (if the same iteration is selected for more than one SSIM_t, the values are highlighted with the color of the highest iteration). Compared to the baseline 3DGS (where all categories are optimized for 30 000 iterations), SAGE reduces Gaussian count from 5.83M to 3.05M at $\text{SSIM}_t = 0.5$, lowering memory usage from 1.45GB to 756.2MB. By accounting for minimum distance in optimization, SAGE prevents spread classes like *sky-other-merged* or *tree-merged* from suffering reduced quality in close-up regions. At a fixed occupancy/memory, SAGE achieves higher visual quality in rendering, as clearly highlighted by the LPIPS metric (Table IV). Finally, figs. 4 and 5 compare SAGE's qualitative results to groundtruth views and standard 3DGS, for three targets SSIM_t. Quality spreads differently across diverse semantic categories; however, SAGE obtains similar quality for every category with $\text{SSIM}_t = 0.7$ while consistently reducing the occupancy. PSNR and LPIPS results are also reported under each scene, showing they are aligned with the SSIM behavior. Note that being SAGE designed to optimize resource allocation rather than maximize perceptual quality alone, it may not always outperform 3DGS in SSIM and LPIPS. However, it always maintains comparable visual quality with significantly lower memory usage, extremely important for real-time XR applications.

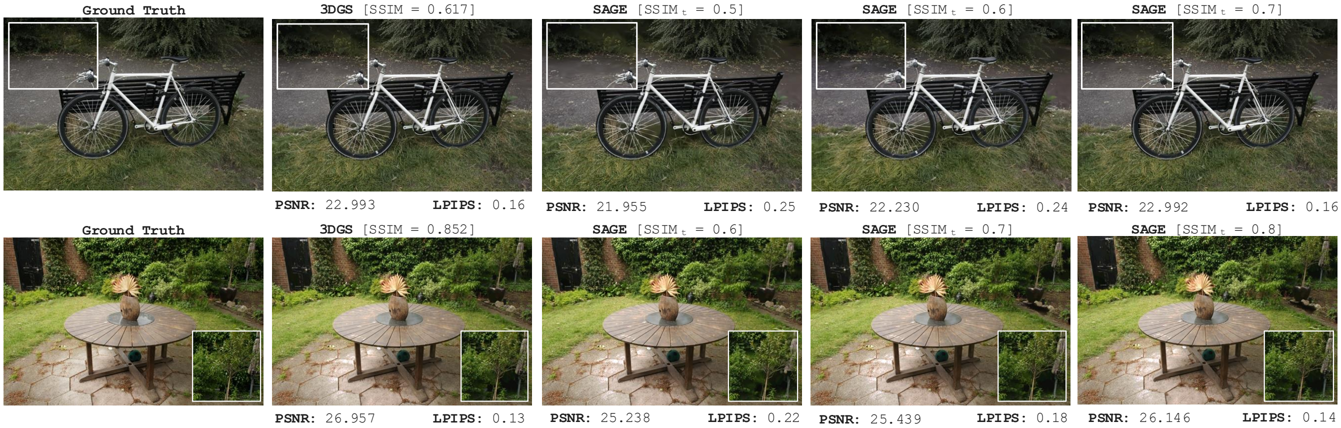

Fig. 5. Qualitative cross-view results on “bicycle” and cross-scene results on ”garden”.

V. Conclusions

In this work, we presented SAGE, a novel strategy for optimizing the rendering of 3D scenes driven by semantics. By using a parametric model to predict the required iterations per semantic region, our approach effectively balances performance and resource usage, minimizing the number of Gaussians while maintaining a target visual quality, which is crucial for XR applications and other resource-constrained environments. Through experiments on the Mip-NeRF360 dataset, SAGE demonstrated not only to achieve significant memory savings compared to the 3DGS baseline but also to be able to effectively adapt the LOD of each semantic category within the 3D scene, based on their visual characteristics. By isolating each category and reconstructing the scene for a given SSIM value, SAGE ensures smooth rendering at reduced computational overhead. The flexibility and resource efficiency of the proposed solution makes it a promising approach for large-scale 3D scene rendering, especially for real-time XR applications. Future work will include subjective evaluations to assess perceptual benefits of SAGE’s semantic adaptation.

References

[1] F. Capraro and S. Milani, “Rendering-aware point cloud coding for mixed reality devices,” in *Proc. IEEE ICIP*, 2019.

[2] A. Javaheri, C. Brites, F. Pereira, and J. Ascenso, “Point cloud rendering after coding: Impacts on subjective and objective quality,” *IEEE Trans. Multim.*, 2021.

[3] F. Biljecki, H. Ledoux, and J. Stoter, “Redefining the level of detail for 3D models,” *GIM International*, 2014.

[4] J. Hasselgren, J. Munkberg, J. Lehtinen, M. Aittala, and S. Laine, “Appearance-driven automatic 3D model simplification,” in *Proc. EGSR*, 2021.

[5] J. T. Doswell and A. Skinner, “Augmenting human cognition with adaptive augmented reality,” in *International Conference on Augmented Cognition*. Springer, 2014.

[6] N. Vaughan, B. Gabrys, and V. N. Dubey, “An overview of self-adaptive technologies within virtual reality training,” *Comput. Sci. Rev.*, 2016.

[7] H. Kim, Y. Yoon, and H. Park, “Adaptation method for level of detail (LOD) of 3D contents,” in *Proc. IFIP NPC*, 2007.

[8] E. Camuffo, F. Battisti, F. Pham, and S. Milani, “Deep 3D model optimization for immersive and interactive applications,” in *Proc. IEEE EUVIP*, 2022.

[9] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “NeRF: representing scenes as neural radiance fields for view synthesis,” *Commun. ACM*, 2021.

[10] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3D Gaussian Splatting for real-time radiance field rendering,” *ACM Trans. Graphics*, 2023.

[11] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, “Mip-NeRF 360: Unbounded anti-aliased neural radiance fields,” *Proc. IEEE CVPR*, 2022.

[12] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017.

[13] O. Oyman and S. Singh, “Quality of experience for HTTP adaptive streaming services,” *IEEE Commun. Mag.*, 2012.

[14] C. Wang, D. Bhat, A. Rizk, and M. Zink, “Design and analysis of QoE-aware quality adaptation for DASH: A spectrum-based approach,” *ACM Trans. Multimedia Comput. Commun. Appl.*, 2017.

[15] D. Lorenzi, M. Nguyen, F. Tashtarian, S. Milani, H. Hellwagner, and C. Timmerer, “Days of future past: an optimization-based adaptive bitrate algorithm over HTTP/3,” in *Proc. EPIC*, 2021.

[16] T. N. Duc, C. T. Minh, T. P. Xuan, and E. Kamioka, “Convolutional neural networks for continuous QoE prediction in video streaming services,” *IEEE Access*, 2020.

[17] Z. Fan, K. Wang, K. Wen, Z. Zhu, D. Xu, and Z. Wang, “Lightgaussian: Unbounded 3D gaussian compression with 15x reduction and 200+ FPS,” 2023.

[18] J. C. Lee, D. Rho, X. Sun, J. H. Ko, and E. Park, “Compact 3D gaussian representation for radiance field,” in *Proc. IEEE CVPR*, 2024.

[19] Z. Yan, W. F. Low, Y. Chen, and G. H. Lee, “Multi-scale 3D Gaussian Splatting for anti-aliased rendering,” in *Proc. IEEE CVPR*, 2024.

[20] K. Ren, L. Jiang, T. Lu, M. Yu, L. Xu, Z. Ni, and B. Dai, “Octree-GS: Towards consistent real-time rendering with LOD-structured 3D gaussians,” *arXiv preprint arXiv:2403.17898*, 2024.

[21] Y. Seo, Y. S. Choi, H. S. Son, and Y. Uh, “FLoD: Integrating flexible level of detail into 3D Gaussian Splatting for customizable rendering,” *arXiv preprint arXiv:2408.12894*, 2024.

[22] D. Campagnolo, E. Camuffo, U. Michieli, P. Borin, S. Milani, and A. Giordano, “Fully automated scan-to-bim via point cloud instance segmentation,” in *Proc. IEEE ICIP*, 2023.

[23] D. Mari, E. Camuffo, and S. Milani, “CACTUS: Content-aware compression and transmission using semantics for automotive LiDAR data,” *Sensors*, 2023.

[24] H. Schieber, J. Young, T. Langlotz, S. Zollmann, and D. Roth, “Semantics-controlled Gaussian Splatting for outdoor scene reconstruction and rendering in virtual reality,” in *Proc. IEEE VR*, 2025.

[25] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proc. IEEE CVPR*, 2018.

[26] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *Proc. ECCV*, 2014.

[27] M. Weber, H. Wang, S. Qiao, J. Xie, M. D. Collins, Y. Zhu, L. Yuan, D. Kim, Q. Yu, D. Cremers, L. Leal-Taixe, A. L. Yuille, F. Schroff, H. Adam, and L. Chen, “DeepLab2: A tensorflow library for deep labeling,” *arXiv preprint arXiv:2106.09748*, 2021.