LetsGo: Large-Scale Garage Modeling and Rendering via LiDAR-Assisted Gaussian Primitives

Jiadi Cui[1, 3, *], Junming Cao[4, *], Yuhui Zhong[2, *], Liao Wang[1, 4], Fuqiang Zhao[1, 4], Penghao Wang[1, 4], Yifan Chen[1, 4], Zhipeng He[1, 4], Lan Xu[1], Yujiao Shi[1], Yingliang Zhang[2, †], and Jingyi Yu[1, †]

[1]ShanghaiTech University, [2]DGene, [3]Stereye, [4]NeuDim

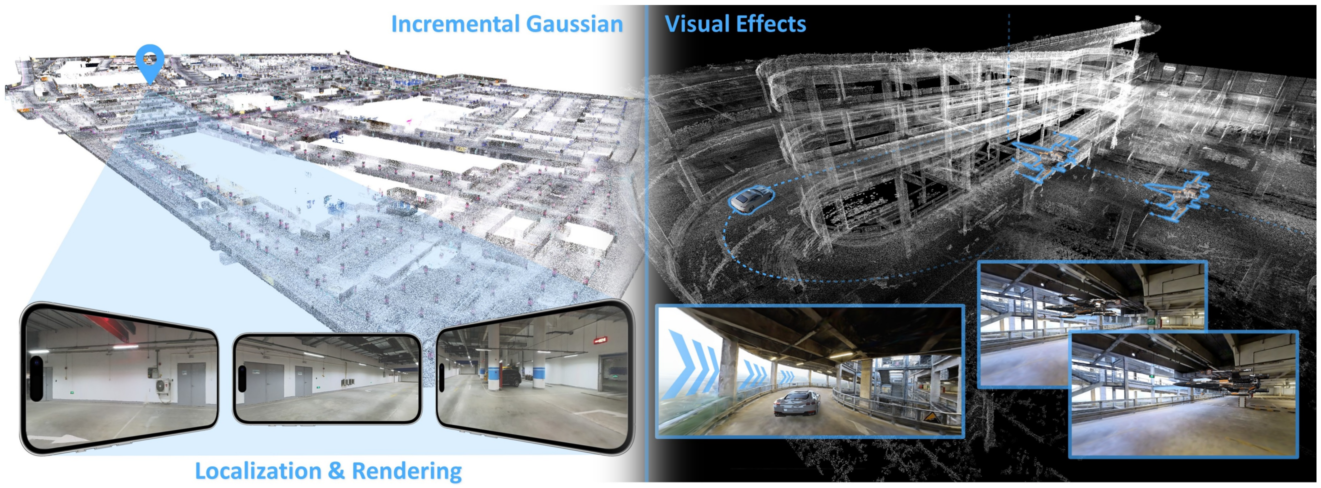


Figure 1: We present LetsGo - an explicit and efficient end-to-end framework for high-fidelity rendering of large-scale garages. We design a handheld Polar scanner to capture RGBD data of expansive parking environments and have scanned a garage dataset, named GarageWorld, comprising five garages with different structures. Our LiDAR-assisted Gaussian primitives approach along with GarageWorld dataset enables various applications, such as autonomous vehicle localization, navigation and parking, as well as VFX production.

Abstract

Large garages are ubiquitous yet intricate scenes in our daily lives, posing challenges characterized by monotonous colors, repetitive patterns, reflective surfaces, and transparent vehicle glass. Conventional Structure from Motion (SfM) methods for camera pose estimation and 3D reconstruction fail in these environments due to poor correspondence construction. To address these challenges, this paper introduces LetsGo, a LiDAR-assisted Gaussian splatting approach for large-scale garage modeling and rendering. We develop a handheld scanner, Polar, equipped with IMU, LiDAR, and a fisheye camera, to facilitate accurate LiDAR and image data scanning. With this Polar device, we present a GarageWorld dataset consisting of five expansive garage scenes with diverse geometric structures and will release the dataset to the community for further research. We demonstrate that the collected LiDAR point cloud by the Polar device enhances a suite of 3D Gaussian splatting algorithms for garage scene modeling and rendering. We also propose a novel depth regularizer for 3D Gaussian splatting algorithm training, effectively eliminating floating artifacts in rendered images, and a lightweight Level of Detail (LOD) Gaussian renderer for real-time viewing on web-based devices. Additionally, we explore a hybrid representation that combines the advantages of traditional mesh in depicting simple geometry and colors (e.g., walls and the ground) with modern 3D Gaussian representations capturing complex details and high-frequency textures. This

[*]These authors contributed equally to this work.
[†]These authors are corresponding authors.

strategy achieves an optimal balance between memory performance and rendering quality. Experimental results on our dataset, along with ScanNet++ and KITTI-360, demonstrate the superiority of our method in rendering quality and resource efficiency.

1. Introduction

For many of us, our daily life begins with a safe departure from a garage and ends with a safe arrival. The garage serves as the origin of our journey to innovation. For visual artists, garages represent a frontier in visual simulation that merges the aesthetic with the technical, offering a canvas where the intricacies of light, shadow, texture, and space coalesce. The complex interplay of artificial and natural lighting within the confines of a garage, with its reflective surfaces, varying materials, and intricate geometries, provides a rigorous testbed for modeling and rendering, pushing the boundaries of what is achievable in virtual environments.

A number of cutting-edge fields can also readily benefit from real-world garage modeling and rendering. For autonomous driving, high-fidelity 3D models of garages are crucial for simulating and testing the complex navigation and parking algorithms that self-driving vehicles must master. By rendering real-world garages with photorealistic details, developers can create varied scenarios, including different obstacles, ensuring that autonomous systems can operate safely and efficiently in real environments. In game design, 3D garages serve as interactive backdrops that can be rich in narrative and aesthetic detail. Digital replications of real-world garages provide a sense of immersion, crucial for players to explore and interact with. For the movie industry, the ability to render photorealistic garages allows for visually stunning special effects that can be seamlessly integrated into live-action footage. This capability is invaluable for creating believable scenes in a controlled environment without the logistical challenges and costs associated with on-location shooting.

Large-scale garages in the real world, however, are inherently difficult to model. They are generally poorly lit and are mainly composed of texture-less walls, sloped surfaces with distorted lines, cluttered and often reflective environments, and complex occlusions between walls, cars, and pillars (Fig. 11). Garages with internal circular or spiral paths further necessitate sophisticated handling of the continuous curvature and potential occlusions caused by the spiral design, which can lead to incomplete data and ambiguities in spatial relationships. Furthermore, the extensive spatial domain of these garages underscores the necessity for efficient 3D representations and lightweight rendering techniques to facilitate real-time interaction and visualization, particularly for scenarios demanding rapid situational assessment, e.g., in navigation and path adjustment.

Traditional computer vision techniques have long struggled to reconstruct accurate 3D models of garages. Passive perception schemes primarily employ color cameras to capture images, utilizing Structure from Motion (SfM) [56, 62, 60, 61] and Multi-view Stereo (MVS) [12, 22, 77, 75, 74] for reconstructing 3D geometry. However, due to the prevalence of texture-less regions and repetitive structural designs within garages, SfM and MVS methods often fail to extract sufficient feature points and establish accurate feature correspondences necessary for estimating camera poses. Active sensing technologies based on LiDAR can calculate camera poses and scene geometry using SLAM [29, 15, 32, 6] algorithms, but the reflective materials and transparent car windows common in garages lead to geometric inaccuracies. More importantly, the LiDAR data tend to be sparse, where the results contain many holes that corrupt high-frequency textures essential for rendering the color appearance of the scene.

Recent advances in neural representation, particularly the Neural Radiance Fields (NeRF) [40] approach, have shown promise in producing high-quality renderings given camera parameters, but they come with high computational costs and lengthy training times. A succession of NeRF-based [44, 9, 79, 26, 50, 82] enhancements have emerged to optimize training duration and visual rendering quality. However, integrating such implicit representations into conventional graphics rendering pipelines and tools for rapid 3D content applications remains challenging. The emerging 3D Gaussian Splatting [25] (3DGS) method has revisited explicit representations, utilizing 3D Gaussians to articulate the geometry and appearance of scenes. This explicit modality not only achieves high-quality scene modeling and rendering but also integrates seamlessly into existing 3D content production workflows. Nonetheless, 3DGS requires a relatively accurate point cloud from SfM for Gaussian initialization, while SfM often fails in the large-garage environment with large texture-less regions and high similarity between different parts of the garages. Furthermore, 3DGS requires a substantial number of 3D Gaussians to depict even the simplest geometries like walls and floors, leading to considerable memory and storage resource consumption that significantly hinders its application in large-scale scenes.

This paper introduces LetsGo - an explicit and efficient end-to-end modeling scheme for high-fidelity rendering of large-scale garages. Our critical insight is to aid the 3D Gaussian splatting variants with calibrated LiDAR points. We design a handheld Polar scanner for expansive garage data collection, which combines IMU, LiDAR and a fisheye camera for robust relative pose estimation. We have scanned five large-scale garages, named GarageWorld, with different geometric structures with this Polar scanner. To our knowledge, this dataset is the first aimed at large-scale garages and will be released to the community. We demonstrate that the LiDAR points collected by this Polar device successfully assist a suite of Gaussian splatting algorithms for garage scene representation. To improve the quality of the 3D Gaussian rendering, we also introduce a depth regularizer that uses depth priors as supervisory signals, significantly reducing floating artifacts and enabling high rendering quality.

Additionally, we develop a lightweight web renderer that supports Level of Detail (LOD) rendering for 3D Gaussians based on the camera's position, orientation, and frustum, enabling real-time high-quality rendering of large-scale scenes on various consumer-level devices. Our results, gathered from our dataset as well as from ScanNet++ [11] and the KITTI-360 [33] datasets, indicate that our approach not only exceeds other methods in rendering quality but also maintains a smaller memory footprint. Our GarageWorld dataset and the LiDAR-assisted Gaussian primitives for large-scale garage modeling and rendering enables various applications (Fig. 1), including autonomous driving, localization, navigation, and visual effects, *etc*.

Our primary contributions are as follows:

- We design a handheld Polar scanner with calibrated IMU, LiDAR, and a fisheye camera, allowing scanning and reconstruction of large-scale garage scenes with large texture-less regions and repetitive patterns where conventional SfM fails.
- We present the first garage dataset, GarageWorld, consisting of five large-scale parking scenes with diverse structures, which will be released to the community to tackle modeling and rendering challenges in expansive environments.
- We demonstrate our LiDAR points assist a suite of Gaussian splatting algorithms, making them feasible for expansive and challenging garage scene modeling. We also introduce a depth prior for Gaussian primitive representation training, and a lightweight renderer to enable real-time rendering on consumer-level devices.

2. Related Work

Conventional Explicit Visual Reconstruction. Conventional algorithms for reconstructing large-scale scenes include Structure from Motion (SfM) [63, 10, 71, 65, 42], Simultaneous Localization and Mapping (SLAM) [29, 4, 5, 7] and Multi-View Stereo (MVS) [58, 18]. They are dedicated to discerning the three-dimensional structure of a scene through the sequential or multi-view analysis of two-dimensional image frames. All these methods leverage feature tracking and multi-view consistency to recover the 3D scene structures. SfM- and SLAM-based methods [63, 10, 56, 1, 13, 21, 67, 55, 31] estimate poses of input images and recover the scene structure jointly. However, their main purpose is pose estimation, and the recovered scene point clouds are always sparse, making it difficult for high-quality free-view synthesis. While MVS-based methods [58, 18, 30, 12, 22, 77, 75, 74] require posed images as input. It computes a dense depth map for each input image, and thus, the constructed point clouds for 3D scene representation are dense. Despite successes, the constructed scenes often lack accuracy and robustness in texture-less and complex scenes.

Implicit Neural Scene Representations. In recent years, neural scene representation [44, 9, 79, 26, 50] has emerged as a promising avenue for addressing the complexities of novel view synthesis. Unlike traditional explicit representation methods, neural approaches leverage the power of deep learning to learn implicit representations directly from data. These methods have shown significant advancements in capturing complex 3D scenes and generating novel views with remarkable realism.

Neural Radiance Fields (NeRF), initially introduced by Mildenhall et al. [40], has revolutionized the field of 3D reconstruction and provided an innovative framework for capturing complex scene details. Since then, a multitude of research has flourished. Considering the original NeRF can only handle bounded and forward-facing scenes, NeRF++ [82] firstly extends NeRF to unbounded scenes by introducing an inverted sphere parameterization. MipNeRF [2] reduces objectionable aliasing artifacts when training and testing images are at different scene resolutions by introducing a new conical frustum rendering scheme instead of rendering along a camera ray. Based on MipNeRF, MipNeRF360 [3] further leverages a non-linear scene representation, online distillation, and a novel distortion-based regularizer to improve the rendering quality in unbounded scenes. NeRF has also been extended to address deformable scenes [46, 38, 37] and scenes with dynamic objects [81, 49, 47].

The original NeRF requires long training and evaluation time, prohibiting its practical use. To achieve real-time 3D modeling and rendering, kiloNeRF [52] proposes to leverage many tiny MLPs to replace the original big MLP, as tiny MLPs require a significantly shorter time for evaluation than the original large MLP. Instant-NGP [43] introduces a new hash encoding approach, which guarantees rendering quality with a smaller neural network and thus reduces the rendering speed. To handle large-scale scenes, Block-NeRF [66] decomposes the scene into individually trained NeRFs, which decouples the rendering time from scene size and enables per-block updates of the environment. Turki et al. [68] introduce Mega-NeRF, which handles varying lighting conditions of images spanning different buildings or city blocks and explores temporal consistency. Considering that previous NeRFs are mainly designed for forward-facing or the 360° object-centric trajectory, F2-NeRF [70] presents a novel space-warping method to handle arbitrary camera trajectories. To handle the pose drift issue of SLAM and SfM for large-scale scene reconstruction and rendering, Wu et al. [72] introduce bundle adjusting neural radiance field, which jointly optimizes camera pose & scene representation, achieving promising results in both indoor and outdoor scenes.

Hybrid Scene Representations. Based on the observation that a sparse point cloud of a scene is often easily obtained by multi-view stereo, PointNeRF [76], PointNeRF++ [64], and TetraNeRF [28] are proposed to combine the merits between explicit scene geometry and implicit neural representations. They have demonstrated impressive visual quality and high rendering speed compared to previous pure implicit neural representations. To handle asynchronously captured LiDAR data and exposure variation between captured images in large urban outdoor environments, Rematas et al. [53] present Urban Radiance Field (URF), which also leverages segmentation maps to supervise densities on rays pointing at the sky.

Utilizing neural networks for scene rendering often comes with a significant trade-off in rendering speed and quality. To address this, the Plenoxels [14] achieves a remarkable two orders of magnitude acceleration in photorealistic view synthesis speed compared to Neural Radiance Fields, maintaining visual quality through optimization of a sparse 3D grid with spherical harmonics from calibrated images. Extending this concept, Wang et al. [69] combine Fourier and hyperspherical harmonics, achieving dynamic reconstruction. Chen et al. introduce TensoRF [8], a groundbreaking approach to radiance field modeling that represents scenes as 4D tensors, leveraging CP and VM decompositions for improved rendering quality and reduced memory footprint compared to NeRF. Building upon this innovation, Jin et al. [23] propose TensoIR, an inverse rendering method that combines tensor factorization and neural fields for efficient estimation of scene geometry, surface reflectance, and environment illumination.

Back to Explicit. Recently, 3D Gaussian splitting (3DGS) [25] revolutionizes the view synthesis quality for unbounded and complete scenes with 1080p resolution rendering. It represents sparse points from SfM/SLAM/LiDAR sensors with 3D Gaussians and directly optimizes the position, anisotropic covariance, opacity, and spherical harmonic (SH) coefficients of each Gaussian. Benefitting from a tile-based rasterizer solution, 3DGS achieves super-fast and ultra-fine rendering quality. Our method is built upon the 3DGS technique and combines the merits of conventional explicit point cloud/mesh and Gaussian representation for large-scale scene reconstruction and rendering.

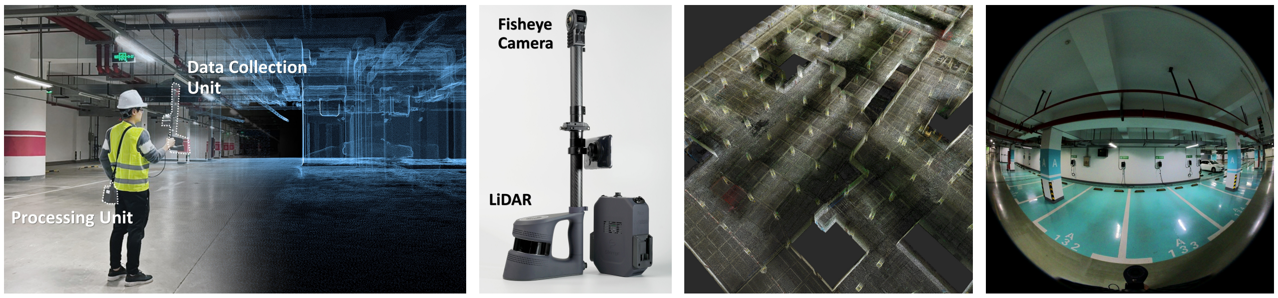
3. Garage Data Capture

3.1. Raw Data Acquisition

Scanning and modeling a large garage is a non-trivial task. Underground and indoor garages often face challenges in receiving GPS signals due to the physical barriers presented by the structures and materials surrounding them, making camera pose estimation for scanning and modeling difficult. Furthermore, there are always large-scale textureless regions inside a garage, *e.g.*, floors and walls. The parked vehicles often contain transparent glasses, and their surfaces are sometimes reflective. This complicated feature matching between images for camera pose estimation and 3D geometry reconstruction. Using a LiDAR sensor for scanning and modeling can provide detailed geometric information. However, the RGB color for each scanned point is not associated.

Capturing Device. To address these problems, we design a lightweight handheld scanning device named "Polar" to jointly collect color and geometric information about the garage. The data collection unit of the Polar device comprises a color fisheye camera, a LiDAR sensor, and an IMU sensor, as visualized in Fig. 2. The fisheye camera captures RGB color information. It has a resolution of 6K and a field of view (FOV) of 270×360 degrees, allowing for quick and comprehensive recording of vivid color data. The LiDAR sensor collects 3D point clouds, recording geometric information at a rate of 2.6 million points per second. With a measurement accuracy of 1 to 1.5 cm and a maximum detection distance of 50 m, it is ideal for 3D scanning of large garage scenes. The IMU sensor provides acceleration information on the device's motion, enabling more accurate pose estimation. In addition to the data acquisition unit, we also equip the Polar device with a data processing unit consisting of a mini PC for real-time SLAM calculations. The data processing unit is powered by two removable batteries that provide more than 30 minutes of single scan endurance. With this unit, one can use a smartphone to connect with the Polar device and preview 3D point cloud reconstruction results in real-time. The entire Polar device weighs no more than 1kg, making it suitable for handheld scanning applications.

Scanning Scheme. We use the Polar device to scan various garages for modeling and free-viewpoint rendering. For each garage, our scanning trajectory aligns with the common vehicle trajectory when driving through garages. We collect data along each trajectory four times: one for forward-facing data collection, one for backward-facing, one for side-left, and one for side-right, resulting in comprehensive capturing of the garage. We set the camera to auto exposure and auto ISO to accommodate complex lighting changes in the garage. For garages that are partially open-air and partially covered, the data is collected when the sun-



(a) Scene (b) Device (c) Point Cloud (d) Fisheye Image

Figure 2: Our compact Polar scanner (b) is engineered for capturing expansive garage environments (a). It is optimized for handheld operation or vehicular mounting, enabling versatile data capture in extensive spaces. At the core of Polar's data acquisition unit lies a high-fidelity LiDAR sensor, capturing precise 3D point clouds (c), complemented by a fisheye camera that procures wide-angle 2D RGB images (d) for a complete scene modeling.

Table 1: Detailed illustrations of our GarageWorld dataset.

Dataset	Category	Geometry	Aera(m^2)	Image Num	Point Num	Face Num	Lighting Condition
Campus 1	Underground	Flat & Sloped Surfaces	38447.86	8479	1.9B	121.9M	Great
Campus 2	Underground	Flat & Sloped Surfaces	28046.37	7772	1.4B	95.8M	Great
Shopping Mall 1	Indoor (Multi-level)	Spiral & Circular Paths	32646.68	5792	1.14B	40.78M	Fair
Shopping Mall 2	Outdoor	Spiral & Sloped paths	13495.92	2280	0.6B	77.1M	Good
Office Building	Indoor	With Mechanical Parking System	22159.25	9308	1.15B	72.3M	Good

light is weak, *i.e.*, during the early morning or evening. This ensures the images captured at the transition between open-air and covered areas share similar illumination. We also apply a short pause for data capturing at the transition area, allowing the sensor to adapt to different lighting conditions and thus ensuring the images are neither overexposed nor underexposed. For garages with motion-activated lights, we ensure the data is collected after the light turns on, maintaining consistent lighting conditions across all images. The travel speed for data collection is at around 1.0±0.2 m/s, with a turning speed of 15°±3 °/s. In areas with insufficient light, these parameters are reduced to 0.5 m/s and 10 °/s, respectively, to avoid motion blur. Some garages are very large, for example, over 30,000 square meters. Thus, we divide the large garage into small subsections and collect data for each subsection. The data is fused later after collection.

3.2. GarageWorld Dataset

We collect data for five garages, including two underground garages on Campus, one indoor garage with multi-levels at Shopping Mall One, one outdoor surface parking at Shopping Mall Two, and one indoor garage with a mechanical parking system in an Office Building. These garages comprise various challenging structures, such as sloped surfaces with distorted lines, internal circular or spiral paths, vehicle elevators, *etc*., as shown in Fig. 11. Almost all our scanned garages contain electric vehicle charging stations, making them eco-friendly. Tab. 1 provides an overall description of the garages.

Underground Garages. We first collect data for two underground garages on campus. This design for constructing garages beneath the ground is often employed in areas with limited above-ground space, providing a solution that preserves surface area for other uses. The coverage of the two garages is around 38.4k and 28k square meters, respectively, and they only have one floor for vehicle parking with flat and sloped surfaces. The two garages are illuminated by constant light conditions, with regular fluorescent light tubes on the ceiling. We collect 8,479 and 7,772 images for the two garages, respectively, with 1.9 and 1.4 billion point clouds.

Indoor Garage with Multi-floors. Staking garages to multi-floors is a common design in dense urban environments. We collect data for this type of parking garage in Shopping Mall One. The different floors of this garage are connected by spiral and circular paths, which have a semi-open structure and are partially illuminated by Sunlight and partially by indoor lights. Given that the lights in the shopping mall are not turned on in the early morning, we collect data for this garage during the early evening to maintain consistent illumination between different images. The total coverage for this garage data is over 32k square meters, with 5,792 images and a point cloud containing 1.14 billion points.

Outdoor Parking. The outdoor surface parking facility is often in areas with large spaces or on the top of a commercial building. We collect data for a garage with this type located on the top of Shopping Mall Two. The entrance to this parking space is from indoor to outdoor, containing spiral and sloped paths. During daylight time, the outdoor illumination is stronger than indoor. At night, the limited dim streetlights are insufficient for photography requirements. Therefore, we conduct our data capture during the early morning and evening when the sunlight is soft, and the illumination between indoors and outdoors is similar. We scanned an area surpassing 13,000 square meters for this garage, resulting in 2,280 images and a point cloud containing 0.6 billion points.

Indoor Garage with a Mechanical Parking System. We also scan several indoor garages with a mechanical parking system in an office building. These garages are typically small in size and contain colored surfaces. The total coverage of these garages is around 22k square meters, containing 9,308 images and 1.15 billion points in point clouds.

3.3. Relative Pose Estimation and Mesh Reconstruction

We start processing the collected garage data by Polar device calibration and dynamic object removal, followed by relative pose estimation for the images captured at different time steps and mesh reconstruction of the entire garage. The detailed descriptions for each step are presented below.

For the intrinsic parameter calibration of the fisheye camera and the IMU sensor, we leverage the program provided by OpenCV and the Allan Variance ROS toolbox [51, 17, 16, 39, 45], respectively. The extrinsic calibration between the fisheye camera and the IMU is done using the Kalibr calibration program [51]. Due to the sparsity, noisiness, and uncolored nature of the point cloud data collected by our LiDAR sensor, it is hard to establish correspondences between the LiDAR data and the images captured by the fisheye camera, making relative pose estimation between the sensors difficult. To address this issue, we introduce an additional sensor, a FARO laser scanner, which provides dense, accurate, and colored point clouds and thus functions as a bridge for the relative pose estimation between the LiDAR sensor and the fisheye camera. The detailed calibration steps between the sensors are presented in the supplementary material.

To remove the dynamic objects in the collected data, we apply Segment Anything [27] to the images. For the point cloud data, we estimate whether the points scanned at one LiDAR frame are also observed at other frames, similar to Schauer and Nüchter [54].

After this preprocessing step, we employ the LiDAR-Inertial-Visual(LIV) SLAM [59] to estimate the relative poses of the sensor between different time steps. The LIV-SLAM system integrates a tightly coupled LiDAR-Inertial Odometry and Visual-Inertial Odometry, along with a joint optimization approach between LiDAR and camera data, for relative pose estimation. Specifically, we employ LIO to fuse IMU data with point clouds collected at different time steps by the LiDAR sensor. This process yields accurate distance measurements and effectively mitigates drift in the IMU data. Subsequently, using the IMU data updated by LIO, we implement VIO among captured fisheye images to estimate their relative poses. Following this, we utilize joint factor graph optimization [35] to refine the estimated poses of the LiDAR and camera. The initial pose graph is constructed by incorporating the updated LiDAR, IMU, and camera poses. Then, bundle adjustment is applied to further optimize the graph. This pose estimation system capitalizes on the strengths of each sensor: IMU data offers short-term motion estimation, LiDAR provides accurate distance measurements, and visual sensors enhance pose estimation in feature-rich environments. Thus, it ensures robust and accurate pose estimation. After merging the point cloud data collected at different time steps using the estimated relative pose, we apply Poisson Reconstruction [24] to convert the point cloud data into an uncolored mesh. To ensure the accuracy and integrity of the resulting mesh, we also compare the reconstructed meth to the original point cloud data to remove incorrect faces.

This geometry-based mesh reconstruction method performs well for Lambertian surface reconstruction. However, it faces challenges for reflective and transparent surfaces, *e.g.*, vehicle glass windows, which is especially common in garage environments. Furthermore, the granularity and preciseness of the reconstructed meshes depend on the density of the 3D points.

4. LiDAR-Assisted Gaussian Primitives

3D Gaussian representation excels at modeling transparent and reflective surfaces compared to mesh. However, it requires a sparse 3D point cloud of the scene and accurate camera parameters obtained from SfM. As discussed previously, the large-scale garage scenes are challenging for SfM algorithms, with low lighting conditions, large textureless regions, repetitive patterns, *etc.*, making sufficient feature correspondences between different images hard to establish. As a result, the capacity of the original 3DGS for high-quality modeling and rendering is limited.

Our Polar scanner with calibrated IMU, LiDAR, and fisheye camera sensors addresses this challenge. The strengths of the different sensors are combined to achieve accurate camera poses and 3D point cloud representations. In the following sections, we describe technical details of our LiDAR-assisted Gaussian splatting with a depth regularizer and a hybrid representation that combines the merits of mesh and 3D Gaussians, followed by training details and experimental evaluation. Fig.3 provides an overview of our LiDAR-assisted Gaussian splatting framework.

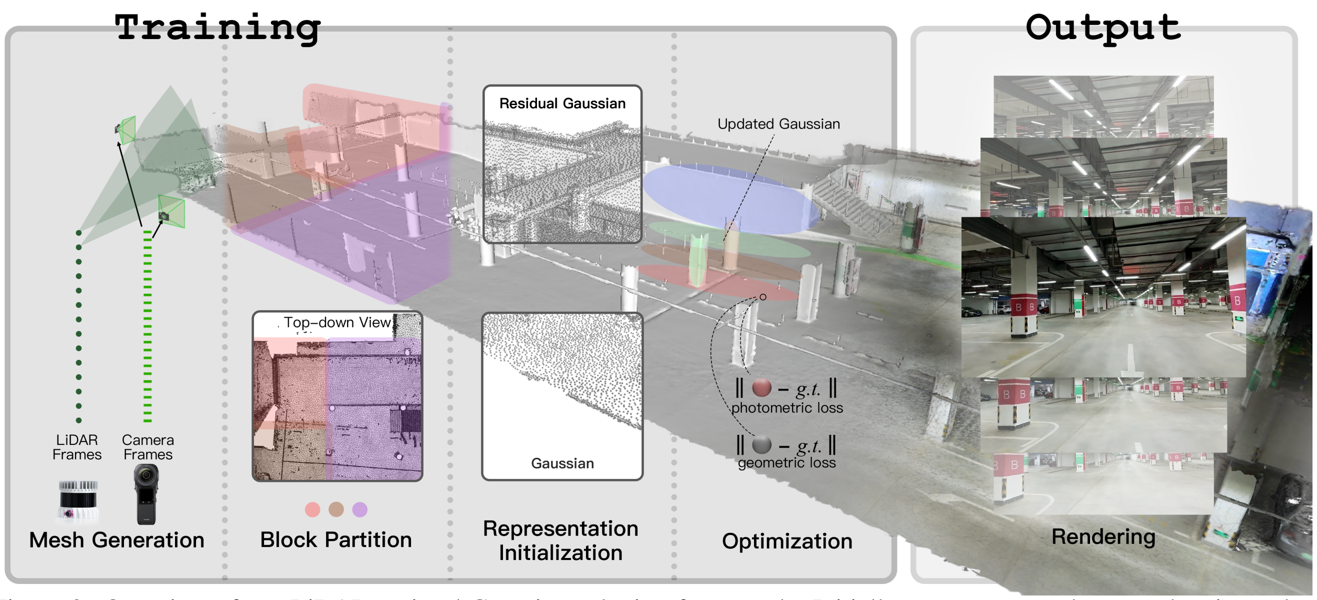


Figure 3: Overview of our LiDAR-assisted Gaussian splatting framework. Initially, we generate a base mesh using color and depth data collected by our self-designed Polar device. Then, the data is partitioned into blocks for parallel and rapid processing. Next, we utilize high-quality scanned point clouds as the input for our LiDAR-GS (Sec.4.1) algorithm. Besides photometric supervision, we also apply our novel unbiased Gaussian depth regularizer for geometric supervision. Finally, our system produces photorealistic rendering results.

4.1. LiDAR-Assisted Gaussian Splatting

3DGS [25] represents 3D points with 3D Guassians, parameterized by position μ, opacity α, anisotropic covariance Σ, and spherical harmonic (SH) coefficients representing view-dependent color c. The projection from 3D Gaussians to 2D images [84] is given by

$$\Sigma' = JW\Sigma W^T J^T, \tag{1}$$

where J represents the Jacobian of the affine approximation of the projective transformation, and W corresponds to the viewing transformation.

The purpose is to optimize the Gaussian parameters so that the rendered images from the 3D Gaussians are as close to their ground truth (GT) images as possible. Normally, the optimization is achieved by stochastic gradient descent (SGD). Since it cannot constrain the covariance matrix to be semi-definite, only in which scenario the covariance matrix has its physical meaning, Kerbl *et al*. proposes to optimize a scaling matrix S and rotation matrix R, and compute the covariance matrix as:

$$\Sigma = RSS^T R^T, \tag{2}$$

where S is parameterized as a 3D vector and R is parameterized as quaternion, a 4D vector with an unit norm.

In this paper, we use our Polar device with a LiDAR sensor to scan point clouds of the garages. Considering the originally scanned point clouds contain noises, we resample a set of new points from the reconstructed mesh with a uniform sampling strategy. These resampled points, in conjunction with the camera parameters, are used to train the 3DGS representations.

In addition to the original Gaussian splatting, we further introduce a depth-regularizer for the 3DGS training, which leverages the high-fidelity LiDAR data to incorporate depth priors during training. We denote this method as LiDAR-GS, and the original 3DGS method with LiDAR-assisted point cloud for Gaussian initialization as 3DGS*.

Depth Regularizer. Inspired by the depth calculation from NeRF [40], we utilize the rasterization pipeline of Gaussians to compute the depth of each Gaussian primitive:

$$D_{\mathrm{G}} = \sum_{i \in N} d_i \alpha_i T_i, \qquad T_i = \prod_{j=1}^{i-1} (1 - \alpha_j), \tag{3}$$

where D_{G} is the rendered depth of the 3D Gaussian primitive and d_i is the depth of each Gaussian splat in camera perspective.

It should be noted that the center of the Gaussian is not directly employed for depth computation. Due to variations in the shape and orientation of the Gaussian, the depth at the precise point where the ray intersects the Gaussian deviates from the depth at its center. Our approach computes the expected depth at the specific point of intersection with the Gaussian as follows:

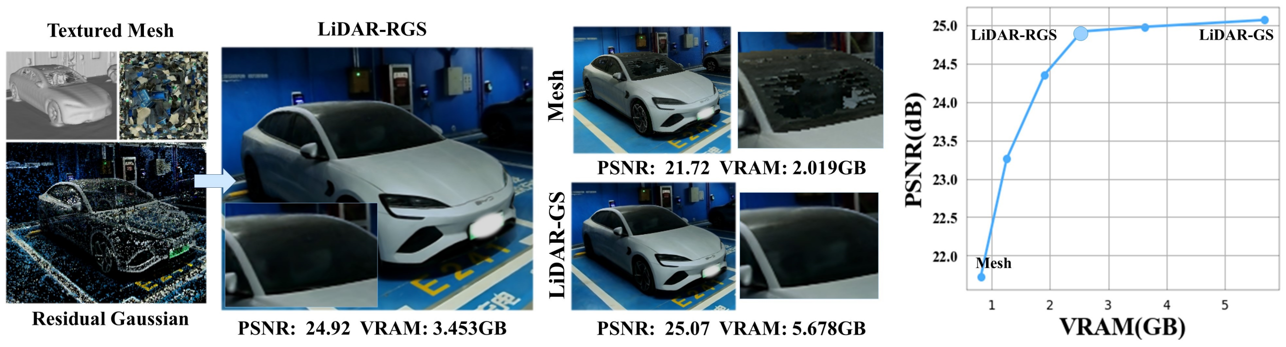


Figure 4: Our LiDAR-RGS system comprises a base mesh, a series of residual Gaussians, and a blending weight map. These components enable our system to maintain a photorealistic appearance comparable to LiDAR-GS while significantly reducing runtime memory. This approach strikes an optimal balance between memory consumption and visual fidelity.

$$d_i = \frac{1}{l}\left(p_2 - \frac{\left(\mathbf{\Sigma}^{-1}\right)_{0,2}}{\left(\mathbf{\Sigma}^{-1}\right)_{2,2}}(x_0 - p_0) - \frac{\left(\mathbf{\Sigma}^{-1}\right)_{1,2}}{\left(\mathbf{\Sigma}^{-1}\right)_{2,2}}(x_1 - p_1)\right) \tag{4}$$

where $\mathbf{p} = [p_0, p_1, p_2]$ represents the position of the Gaussian center in the ray space, and $\mathbf{\Sigma}$ is the 3×3 covariance matrix. $(\cdot)_{m,n}$ represents the corresponding element in the matrix. For detailed derivation and understanding of this process, please refer to the supplementary materials provided with the paper.

Given the K captured views, we compute the depth loss using the following equation:

$$\mathcal{L}_{\text{depth}} = \sum_k \left\| D_G^k - D^k \right\|_1, \tag{5}$$

where D^k represents the inherent depth prior from LiDAR data.

The total loss function is as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{rgb}} + \lambda_{\text{depth}}\mathcal{L}_{\text{depth}}, \tag{6}$$

where $\mathcal{L}_{\text{rgb}}$ is the RGB image reconstruction loss, following the original 3DGS, and λ_{depth} is the weight for our depth term. By incorporating the depth constraint, our LiDAR-GS effectively minimizes the occurrence of floating artifacts and aligns the Gaussian kernel more closely with the depth information inherent in the LiDAR data.

Furthermore, our method also supports residual Gaussian, a hybrid representation composed of meshes and Gaussian. In scenarios of large-scale underground garages, most parts consist of continuous planes like the ground, which can be effectively utilized by a rough mesh. Meanwhile, details such as texture and glasses on the car can be effectively represented by the Gaussian. As shown in Fig. 4, by applying this hybrid representation, the memory consumption for training and model storage can be significantly reduced.

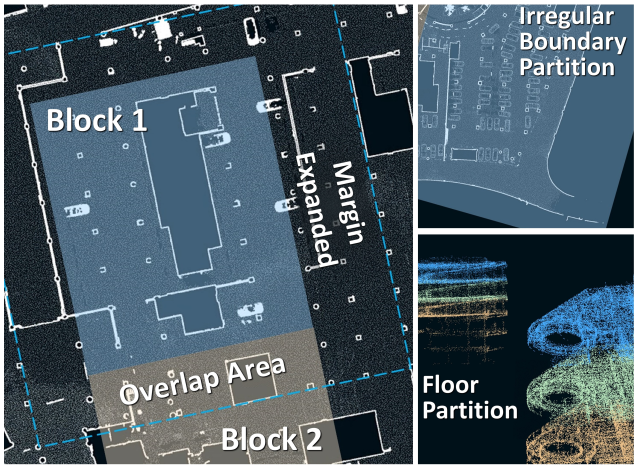


Figure 5: Illustration of our partition scheme. The left image shows the margin expansion method employed to guarantee overlap amongst the partitioned blocks. The top right image demonstrates the extension of partition boundaries to incorporate irregular areas. The bottom right image details the approach for multi-level parking structures, wherein each level is segregated into different partitions.

4.2. Training Details

We train our model using the PyTorch Framework on NVIDIA RTX A6000 GPUs. Considering the large scale of our scene and the use of LiDAR point clouds for initialization, we adjust the scaling learning rate to 0.0015 and set the initial position learning rate for residual Gaussians primitives to 0.000016. Additionally, we increase the opacity reset interval to 2,000,000 and delay the Gaussian densification step start to 75,000. The total iteration count is empirically set to twenty times the number of captured images. We set the spherical harmonics (SH) degree to 2 and the weights for regularization terms λ_{depth} and λ^*_{depth} to

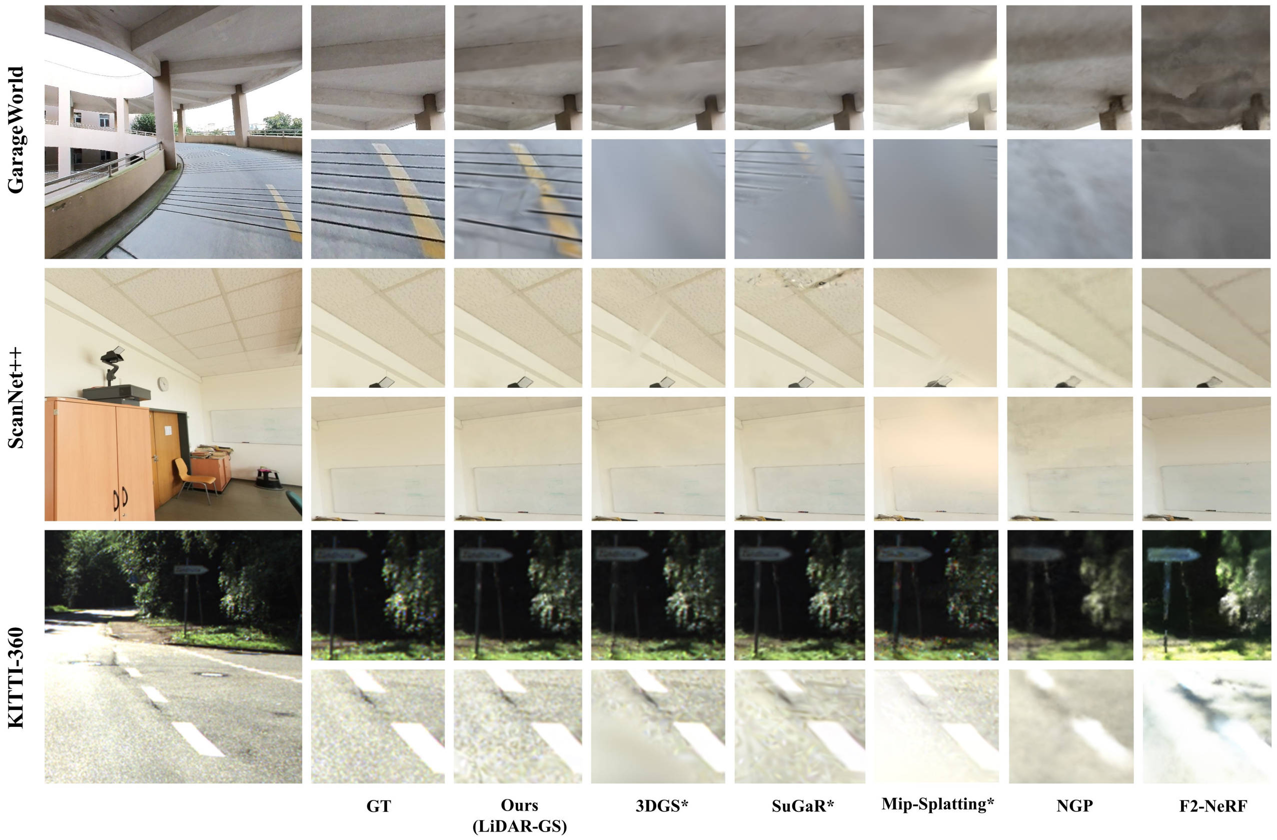


Figure 6: Qualitative comparison between our LiDAR-GS, 3DGS*, SuGaR*, Mip-Splatting*, NGP and F2-NeRF on the various datasets. Here, we use a superscription * to denote their Gaussian primitives are initialized according to our LiDAR-assisted 3D point cloud.

Table 2: Quantitative comparison between our LiDAR-GS, 3DGS*, SuGaR*, Mip-Splatting*, NGP and F2-NeRF on the various datasets.

Method	GarageWorld			ScanNet++			KITTI-360		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
3DGS*	23.52	0.822	0.412	27.41	0.902	**0.149**	20.39	0.698	0.289
Mip-Splatting*	22.08	0.791	0.448	25.74	0.898	0.179	20.19	0.682	0.318
SuGaR*	24.14	0.828	**0.400**	27.28	0.897	0.167	21.18	0.706	0.313
F2-NeRF	18.88	0.739	0.552	23.59	0.888	0.237	18.60	0.653	0.414
NGP	20.68	0.734	0.507	**28.72**	0.896	0.230	21.21	0.655	0.406
Ours(LiDAR-GS)	**24.63**	**0.835**	**0.400**	27.54	**0.903**	**0.149**	**21.26**	**0.722**	**0.269**

0.8, respectively. We also apply partition-based training to train our data in parallel for acceleration.

Block Partitioning. As illustrated in Fig. 5, we partition each level of garages into sub-blocks with overlapped quadrilateral, typically a rectangle or a trapezoid with small internal angles. Each block covers approximately 100 square meters. This enables scene representation training at a small scale.

Edge Expansion and Point Cloud Partitioning. Due to poor reconstruction quality at edges by Gaussian splatting based methods, we expand each quadrilateral outward by 30% to enhance the reconstruction quality. Since the collected garage point clouds are vertically aligned with the Z-axis, we only need the quadrilateral vertices for point cloud segmentation and downsample the point cloud at 4cm intervals.

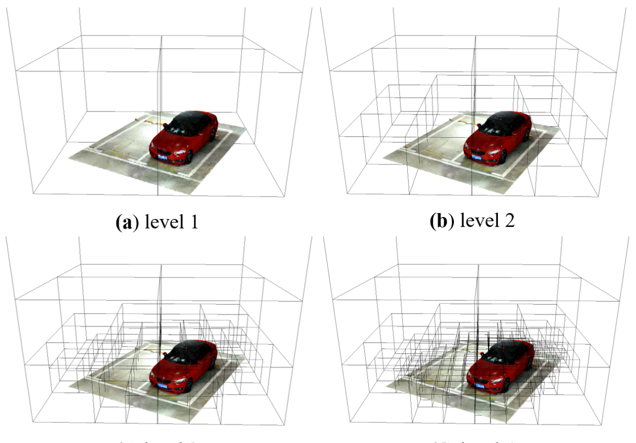

(a) level 1 (b) level 2 (c) level 3 (d) level 4

Figure 7: Illumination of different Gaussian splatting octree levels. The root level contains the lowest density subsample of the original Gaussian splats, and with each level, the resolution is doubled.

Image Selection. After processing the original fisheye images, we retain camera views with origins inside the partition. For cameras outside the partition, we project the complete and quadrilateral-partitioned meshes to the views, comparing the number of triangles visible from each view. Views with a ratio greater than a certain threshold, determined as 0.8 from experiments, are retained.

Results Merging. After partition training, we use the original quadrilateral coordinates to segment and merge the results, achieving a complete scene reconstruction.

4.3. Experimental Results

Datasets. In this section, we compare our approach with the recent state-of-the-art on various challenging large-scale datasets, including our **GarageWorld**, **KITTI-360** [33], a large outdoor street scene, and **ScanNet++** [78], an indoor scene characterized by complex geometry and variable lighting. We adopt 10% images of each scene for test sets, others are training sets.

Competing Methods We compare our method against a suite of 3DGS approaches, including **3DGS***[25], **Mip-Splatting***[80], and **SuGaR***[20]. They all utilized our LiDAR-derived point clouds as initialization, enabling their application to the challenging garage scenes. Additionally, our approach is benchmarked against recent implicit representation methods, namely **F2-NeRF**[70] and **Instant-NGP**[43],

Evaluation Metrics. For quantitative comparisons, we adopt Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS)[83] as metrics to evaluate the rendering quality.

Results. As shown in Fig. 6, 3DGS* and SuGaR* tend to generate floating Gaussian points around scene surfaces, leading to blurriness and poor rendering quality. F2-NeRF performs well on the relatively small-scale scene of ScanNet++ but suffers from artifacts in the large-scale KITTI-360 dataset and blurry in our GarageWorld dataset. As for Instant-NGP, which utilizes hash encoding, its rendering results in large-scale scenes appear notably blurry. In contrast, our method shows the highest rendering quality in various scenarios. The comparison between Ours (LiDAR-GS) and 3DGS* demonstrates that our depth regularizer successfully suppresses the float Gaussian primitives, resulting in improved rendering quality. The quantitative comparisons between the state-of-the-art methods on the three datasets are presented in Tab. 2. Our method consistently achieves the best result.

5. Web-based Lightweight Renderer

Lightweight rendering is crucial for enabling real-time visualization on commonly used devices like laptops and mobile phones. It reduces the dependency on high-performance computing devices and ensures a positive user experience across various application scenarios. In the context of large-scale scenes, exemplified by the garages discussed in this paper, the 3D Gaussian splatting necessitates a substantial number of Gaussians to depict the entire scene comprehensively. However, mainstream lightweight devices currently grapple with the challenge of simultaneously loading all Gaussian splats into VRAM for rendering. To mitigate this, this paper partitioned the large garage scene into small sub-blocks. Despite this optimization, the required number of Gaussians is still significant, making real-time rendering on lightweight devices challenging.

In response to the complexities posed by large-scale scenes, recent approaches [57, 36] have turned to a Level of Detail (LOD) structure for storing and loading point clouds or textured mesh data into memory. Drawing inspiration from this LOD structure, this paper introduces a new renderer tailored to our LiDAR-assisted Gaussian primitives. We present a LOD structure for 3D Gaussian storage and rendering at different resolutions. This innovation enables real-time, high-quality rendering on lightweight devices, addressing the unique demands of large-scale scenes.

5.1. Level of Detail Structure for 3D Gaussians

Similar to Potree [57], we store the unstructured 3D Gaussians of our representation in the partitioned sub-block of the large-scale garage in a layered manner in an octree, as shown in Fig. 7. Each level of the Octree contains a

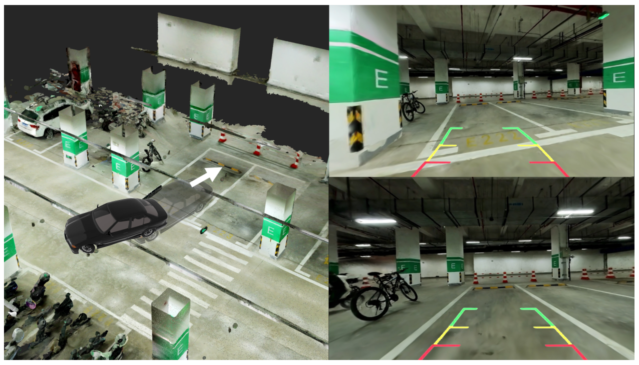


Figure 8: Autonomous vehicle parking. Our diverse garage scenes facilitate training algorithms for generating parking trajectories under different scenarios. When guiding the vehicle to the parking space, our garage model allows real-time and wide-FOV rendering of the environment, capturing drivable area and obstables, thereby enhancing safe parking capabilities.

subsample of the whole data at a specific resolution, with the density/resolution of the subsample decreasing with the level number. The max level of the Octree contains the original residual 3D Gaussian data at a maximum level of detail, while the lowest level denotes the root node and stores the coarsest subsample of the whole data at the lowest level of detail. Following previous works, we apply chunking, merging, and downsampling operations to create the octrees at each level.

The chunking step splits the original 3D Gaussians into cubic chunks with an appropriate chunk size, which is small enough that parallel processing for multiple chunks can be guaranteed while being large enough to contain adequate numbers of 3D Gassians to avoid massive tiny chunks. In our scenario, we divide the space of the partitioned sub-block into a chunk grid of 128^3 and count at most 10k 3D Gaussians within each grid. Unlike previous works for point cloud LOD rendering that only need to consider the position of the 3D point, we take into account both the position and radius of the 3D Gaussians during the chunking step. A 3D Gaussian is assigned to a chunk when its center position is within this chunk and also at least 50% of its radius extends within the boundaries of the chunk, ensuring that most of the Gaussian volume is contained within the designated spatial segment. We merge adjacent chunks containing fewer than 10k 3D Gaussians to avoid sparsity within any chunk. To create the LOD structure, we downsample the leaf nodes of a higher-level octree to generate the subsample of 3D Gaussians at the lower level. This process is repeated recursively until the coarsest level.

We store the chunking information at each level, including the whole region of the garage scene, the number of 3D Gaussians, and the starting and ending address of each chunk in binary files. During the following rendering processes, these metadata are loaded into memory, assisting in deciding which parts of the 3D Gaussians to access.

5.2. Coarse-to-Fine Rendering

We implement a coarse-to-fine loading and rendering scheme for the residual 3D Gaussians to enable real-time rendering on lightweight devices. We divide the LOD structure for 3D Gaussians into two parts. Levels under four are preloaded at the beginning and kept in memory. These are primarily used for rendering the overall scene at a lower resolution. Since they store a relatively smaller number of 3D Gaussians and are frequently accessed data, keeping them continuously in memory does not significantly impact resource usage while helping avoid I/O issues associated with frequent loading. Levels higher than four are dynamically loaded into memory to compensate for scene details of the coarse levels based on the given camera parameters. To ensure the rapid rendering of the dynamically loaded data, we employ a multi-threaded asynchronous reading approach, loading each level of 3D Gaussians in parallel. Additionally, to guarantee the loading time of the new 3D Gaussian nodes at the updated viewing positions, we set an upper limit on the size of data loaded at once. This limit is set to 2 million 3D Gaussians per load in our implementation.

The above-illustrated loading and rendering parameters could be adjusted according to the memory size and computational power of different devices, thereby accommodating the rendering tasks of garage scenes on various devices. We test the performance of our renderer on different devices. On a high-performance desktop equipped with an i9-10900X CPU and a Samsung SSD T5 Disk, it takes 1.36 seconds to load approximately 2 million 3D Gaussians. On a MacBook laptop equipped with an Apple M2 CPU and APPLE SSD AP0512Z Disk, it takes 1.29 seconds to load the same 2 million 3D Gaussians. Combining these extra 3D Gaussians with the preloaded 3 million Gaussians of the scene, both devices maintain a rendering frame rate of 60 FPS.

For our hybrid mesh and residual Gaussian representation, we leverage an open-source library, Nexus [48], to store and render mesh from different resolutions, and the residual Gaussian primitives are rendered using the above illustrated LOD renderer.

6. GarageWorld and Applications

Fig. 11 visualizes the GarageWorld dataset, consisting of five large-scale garages that exhibit a diverse array of types: underground garages, outdoor parking, an indoor garage with multi-floors, and an indoor garage featured with mechanical parking systems. These environments are characterized by intricate geometric structures, such as sloped

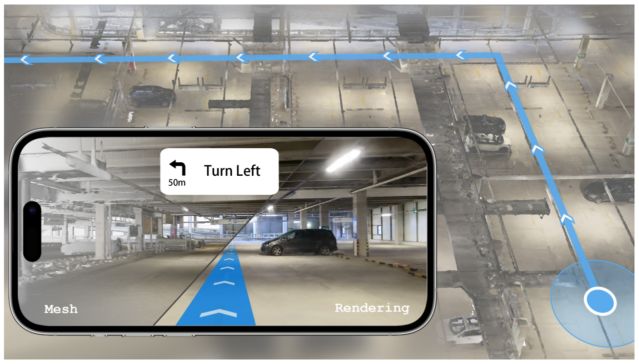


Figure 9: Real-time localization & navigation in challenging garage environments. Our colored 3D model facilitates precise vehicle camera localization and optimal path navigation, particularly in low-light garage conditions, ensuring safe driving. The lightweight web-based rendering ensures deployability in vehicles with limited computing resources.

surfaces with distorted lines, internal circular paths, spiral ramps, and parked vehicles. The point clouds illustrated in Fig. 11 capture the comprehensive architectures of the garages. Our neural rendering pipeline is employed to produce images from multiple perspectives, exemplifying the photorealistic quality of our rendering technique. Our neural rendering pipeline and the GarageWorld dataset are designed to support an extensive range of applications, details of which are delineated in the forthcoming sections.

6.1. Data Generation and Testbed for Autonomous Driving

Our reconstructed 3D structure for large-scale garage scenes and high-quality real-time rendering assists various autonomous driving algorithms. For example, autonomous vehicles are equipped with multiple sensors, including LiDAR, IMU and cameras, to perceive the environment. The reconstructed 3D structure aids in sensor fusion algorithms by providing a diverse and accurate dataset for training. Our GarageWorld dataset includes a wide range of driving scenarios, such as tight parking spaces, multi-level structures, complex spiral and circular paths, diverse geometric layouts. This provides various solution generation for different autonomous driving applications, such as the trajectory generation for autonomous parking, path planning for navigation, *etc*. With the generated data, autonomous driving algorithms can be trained to handle these complex scenarios, improving their ability to operate effectively in various garage environments. Furthermore, benefiting from our reconstructed garage world by LiDAR-assisted Gaussian primitives, algorithms can be rigorously tested and validated in a controlled virtual environment before being deployed in real-world scenarios. This accelerates the development and deployment process while ensuring a higher level of safety. Additionally, the challenging garage environments, as discussed previously, with large textureless regions, repetitive patterns, transparent and reflective surfaces, *etc*., offer a realistic testbed for SfM and visual SLAM techniques.

6.2. Real-time Localization and Navigation

Apart from generating various training data in complex scenarios, our LiDAR-assisted garage modeling and rendering also benefit autonomous vehicle localization and navigation by providing accurate 3D references and complementing real-time sensor readings.

In the challenging indoor or underground garage environment, there is often limited or no access to GPS signals, making traditional GPS-based localization challenging. The confined spaces and structured obstacles can lead to interference for LiDAR and radar sensors, and multipath reflections in the confined spaces can distort sensor readings, making localization by LiDAR and radar less reliable. Furthermore, the low lighting in underground and indoor garages reduces camera visibility, posing challenges for purely vision-based localization. Benefiting from our Polar scanner and LiDAR-assisted Gaussian primitives, we provide accurate 3D references with colors that assist devices in accurately recognizing and matching environmental features. This integration significantly enhances localization and navigation accuracy and reliability.

Fig. 9 presents an example of real-time localization and navigation. When the task is triggered, our localization system correlates an image captured at the current location with our reconstructed colored 3D models and computes a relative pose. Then, our navigation system generates an optimal path connecting the current location and destination in the complex garage environment. During the driving process, our 3D model and real-time rendering complement the vehicle camera readings and broaden the field of view of the vehicle, providing a comprehensive understanding of the environment, thus enabling safe driving. Our web-based rendering engine empowers our system to represent expansive 3D garage maps and navigation trajectories on lightweight devices with limited computational capacity, thus delivering smooth real-time interactions and superior rendering efficacy.

6.3. Autonomous Vehicle Parking

Our diverse and complex garage environment provides various challenging parking scenarios with different parking trajectories for vehicle parking algorithm training. During deployment, based on the vivid rendering output of the garage, autonomous driving systems analyze available parking spaces, generate a trajectory, and guide the vehicle into these spots along the trajectory.

Apart from this trajectory planning, our LetsGo pipeline for garage modeling and rendering can also assist au-

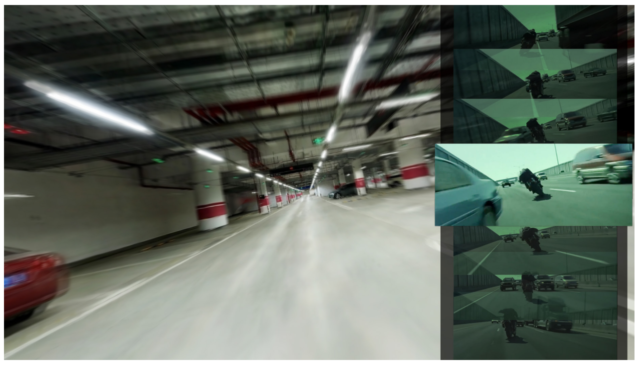


Figure 10: VFX demonstration. Through an analysis of the animation in the reference video, we manually extract the poses of several keyframes, which enable our system's renderer to generate corresponding video segments. Our 3D garage modeling and rendering also enables motion blur rendering, producing realistic visual effects.

tonomous vehicles in safe parking by complementing sensor readings. As discussed previously, due to the low light conditions in garage environments, real-time sensor observations by vehicle-mounted cameras may not observe the surrounding scene accurately. The vehicle-equipped LiDAR or Radar sensors may suffer inaccuracies due to reflections in the confined space. Our detailed 3D model helps identify and classify obstacles within the garage, ensuring safe and efficient parking. As shown in Fig. 8, our method generates rear view images with a wide field of view, encapsulating drivable area and obstacles (*e.g*., walls). This allows accurate distance measurement between the vehicle and the obstacle, avoiding collisions. The proposed lightweight real-time rendering allows the vehicle to dynamically assess its surroundings and make quick decisions, increasing safety and navigating the vehicle through tight spaces.

6.4. VFX Production

Our reconstructed garage datasets and real-time rendering can significantly contribute to VFX productions. For example, iconic scenes like the highway chase in "The Matrix Reloaded" are captivating and impressive. Filming such scenes can involve shooting or simulating extreme locations using visual effects. Our LetsGo pipeline can achieve high-quality modeling and rendering of complex garage scenes. Moreover, real-time rendering enables the manipulation of various factors, such as exposure time and the significance of motion blur, which can be challenging to adjust in actual filming scenarios. Our GarageWorld serves as a valuable foundation for creating lifelike backgrounds or integrating CGI (Computer-Generated Imagery) elements seamlessly into live-action footage.

To demonstrate the capabilities of our approach, we create a VFX video production using our reconstructed garage models (Fig. 10) and the supplementary video. In this demonstration, we download a video depicting highway chase scenes with dynamic viewing angles of a rapidly moving motorcycle. By manually extracting the camera trajectory, we align it with one of our garage scenes and render corresponding images. We also synthesize motion blur during the rendering process, creating compelling visual effects of a chase inside a garage.

Moreover, our 3D garage model serves as a versatile background that can be overlaid with special effects, such as shots of cars, for movie scene creation. With the advantage of real-time rendering, VFX artists can explore various camera angles and compositions before finalizing the shot. This eliminates the need for extensive reshooting, saving valuable time and resources in the production pipeline. Additionally, the appearance consistency between the real and virtual components can be guaranteed, enhancing the overall believability of the scenes.

7. Limitations and Discussions

Given that the large texture-less regions and repetitive patterns in garage scenes hinder the 3D reconstruction of expansive garages from images by SfM and MVS algorithms, this paper designs a Polar device using which the point cloud data of the large-scale garages can be easily captured and calibrated, and demonstrated that the collected LiDAR data successfully assist a suit of 3D Gaussian splatting algorithms, enabling high-quality rendering. Moreover, our lightweight renderer, combined with LOD techniques, facilitates real-time rendering of expansive garage environments on lightweight platforms. Although our rendering results exhibit commendable realism, our workflow still possesses certain limitations. Here we present a detailed analysis and explore further potential applications.

Firstly, our method focuses on large-scale garage scenes and relies on our lightweight 3D scanner to provide high-quality LiDAR point clouds, color images, and corresponding camera information. Although we have also validated the effectiveness of our method on other open-source datasets, it is important to explore the use of even lighter devices for scanning and rendering large-scale scenes, such as smartphones equipped with depth sensors. Additionally, we have currently collected data from five large-scale garages, but it is necessary to gather more garage data to contribute to the community and facilitate further research on garage modeling and rendering.

As a method based on image rendering, our approach achieves highly realistic rendering effects, almost indistinguishable from real scenes. However, the existing pipeline does not support modifying lighting conditions. This requires us to carefully design the shooting process accord-

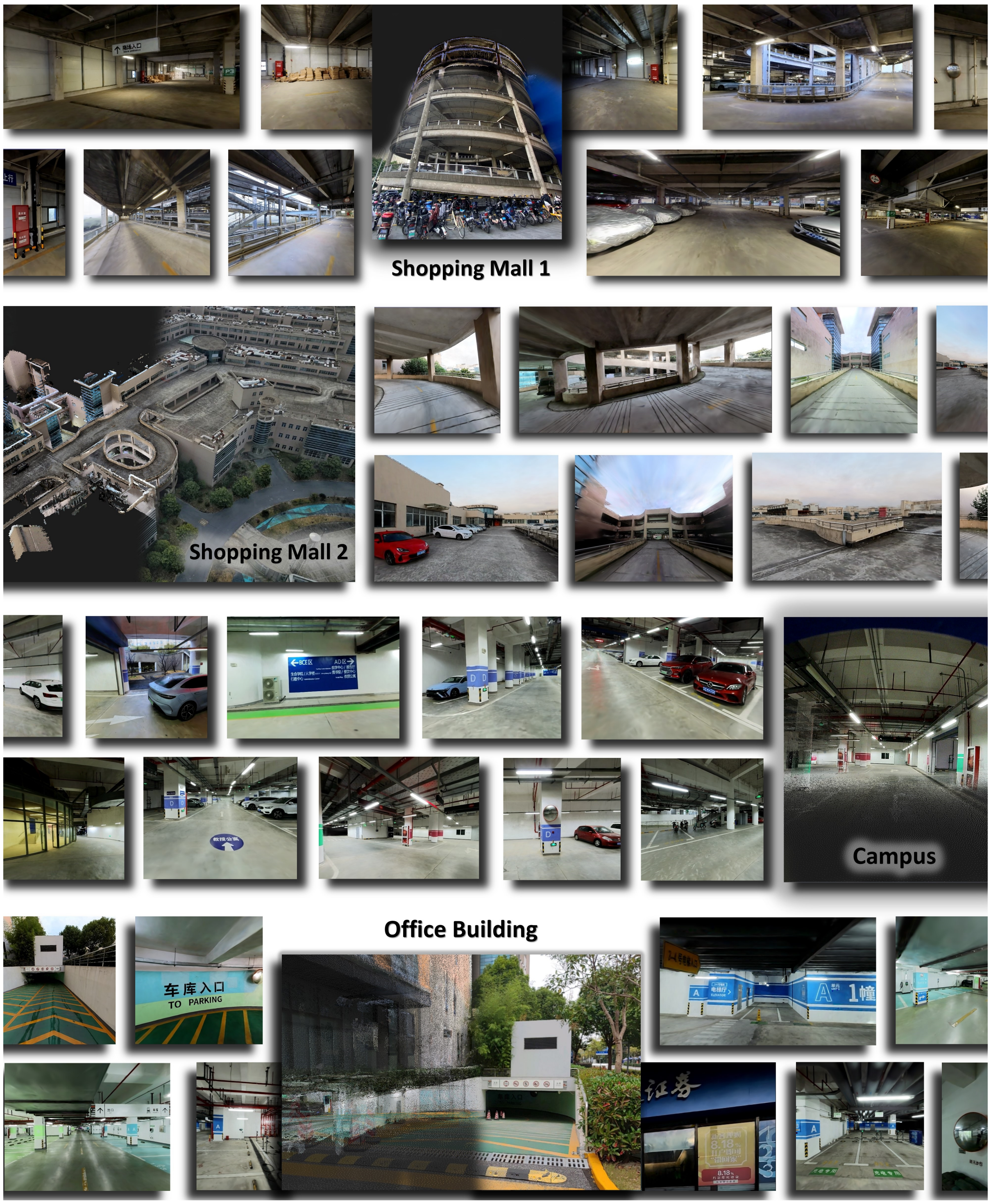


Figure 11: A gallery showcasing the rendering results of our pipeline across various scenes in GarageWorld. Our LiDAR-assisted Gaussian primitives enable photorealistic rendering of expansive garages. The enlarged images illustrate the fidelity of our high-resolution point clouds and captured imagery.

ing to the lighting conditions, limiting the applicability of our method in various scenarios. Recently, Gaussian-based work SuGaR [20] has introduced editing operations such as adjusting the size, position, and orientation of objects, but it still does not support modifying lighting information. Enabling rendering and editing of large-scale scenes under different lighting conditions is a meaningful direction that deserves further investigation. Our open-source dataset provides a foundation for the community to conduct research in these directions, allowing for advancements in this field.

We have demonstrated the applications of our method in automatic driving data generation, localization and navigation on lightweight devices, and the production of visual effects in films. These applications illustrate the excellent performance of our method in rendering quality. Furthermore, there are additional directions and applications worth exploring. Inspired by SMEREF [41], one potential research direction is to investigate streaming transmission methods for Gaussian kernels, enabling the distribution and on-the-fly rendering of large-scale scene data. Additionally, there are city generation methods such as InfiniCity [34] and CityDreamer [73], which leverage Generative Adversarial Networks (GANs) [19] to achieve rapid modeling of large-scale scenes. We intend to study large-scale scene generation based on 3D Gaussian representations. This fully explicit representation can be easily integrated into existing computer graphics workflows and achieve superior rendering effects.

8. Conclusion

This paper has contributed a handheld Polar device for data collection, a GarageWorld dataset, LiDAR-assisted Gaussian primitives for scene representation, and a lightweight rendering technique that allows web-based rendering on consumer-level devices. Benefiting from these innovations, we successfully reconstruct various garages with diverse and challenging environments, allowing real-time lightweight rendering from any viewpoint. Experimental results on the collected and two public datasets have demonstrated the effectiveness of our approach. Our GarageWorld, along with the reconstructed 3D model and real-time rendering, enables a set of applications, including training data generation and testbed for autonomous driving algorithms, real-time assistance for autonomous vehicle localization, navigation, and parking, as well as VFX production. Our current contributions mainly focus on the perception of the world, and it enables downstream recognition tasks. In the future, we will also explore garage generation, keeping pushing the boundary of garage modeling and accomplishing a closure from perception, recognition and generation.

References

[1] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011.

[2] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021.

[3] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022.

[4] H. Bavle, J. L. Sanchez-Lopez, C. Cimarelli, A. Tourani, and H. Voos. From slam to situational awareness: Challenges and survey. *Sensors*, 23(10):4849, 2023.

[5] M. Bujanca, X. Shi, M. Spear, P. Zhao, B. Lennox, and M. Luján. Robust slam systems: Are we there yet? In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5320–5327. IEEE, 2021.

[6] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[7] S. Ceriani, C. Sánchez, P. Taddei, E. Wolfart, and V. Sequeira. Pose interpolation slam for large maps using moving 3d sensors. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 750–757. IEEE, 2015.

[8] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022.

[9] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021.

[10] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *CVPR 2011*, pages 3001–3008. IEEE, 2011.

[11] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[12] Y. Dai, Z. Zhu, Z. Rao, and B. Li. Mvs2: Deep unsupervised multi-view stereo with multi-view symmetry. In *2019 International Conference on 3D Vision (3DV)*, pages 1–8. Ieee, 2019.

[13] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, et al. Building rome on a cloudless day. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 368–381. Springer, 2010.

[14] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022.

[15] Q. Fu, H. Yu, X. Wang, Z. Yang, Y. He, H. Zhang, and A. Mian. Fast orb-slam without keypoint descriptors. *IEEE Transactions on Image Processing*, 31:1433–1446, 2021.

[16] P. Furgale, T. D. Barfoot, and G. Sibley. Continuous-time batch estimation using temporal basis functions. In *2012 IEEE International Conference on Robotics and Automation*, pages 2088–2095. IEEE, 2012.

[17] P. Furgale, J. Rehder, and R. Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1280–1286. IEEE, 2013.

[18] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz. Multi-view stereo for community photo collections. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[20] A. Guédon and V. Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *arXiv preprint arXiv:2311.12775*, 2023.

[21] J. Heinly, J. L. Schonberger, E. Dunn, and J.-M. Frahm. Reconstructing the world* in six days*(as captured by the yahoo 100 million image dataset). In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3287–3295, 2015.

[22] B. Huang, H. Yi, C. Huang, Y. He, J. Liu, and X. Liu. M3vsnet: Unsupervised multi-metric multi-view stereo network. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 3163–3167. IEEE, 2021.

[23] H. Jin, I. Liu, P. Xu, X. Zhang, S. Han, S. Bi, X. Zhou, Z. Xu, and H. Su. Tensoir: Tensorial inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2023.

[24] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, page 0, 2006.

[25] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023.

[26] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19729–19739, 2023.

[27] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollar, and R. Girshick. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, October 2023.

[28] J. Kulhanek and T. Sattler. Tetra-nerf: Representing neural radiance fields using tetrahedra. *arXiv preprint arXiv:2304.09987*, 2023.

[29] J. J. Leonard and H. F. Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *IROS*, volume 3, pages 1442–1447, 1991.

[30] J. Li, Z. Lu, Y. Wang, Y. Wang, and J. Xiao. Ds-mvsnet: Unsupervised multi-view stereo via depth synthesis. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 5593–5601, 2022.

[31] J. Li, L. Pei, D. Zou, S. Xia, Q. Wu, T. Li, Z. Sun, and W. Yu. Attention-slam: A visual monocular slam learning from human gaze. *IEEE Sensors Journal*, 21(5):6408–6420, 2020.

[32] Y. Li, N. Brasch, Y. Wang, N. Navab, and F. Tombari. Structure-slam: Low-drift monocular slam in indoor environments. *IEEE Robotics and Automation Letters*, 5(4):6583–6590, 2020.

[33] Y. Liao, J. Xie, and A. Geiger. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3292–3310, 2022.

[34] C. H. Lin, H.-Y. Lee, W. Menapace, M. Chai, A. Siarohin, M.-H. Yang, and S. Tulyakov. InfiniCity: Infinite-scale city synthesis. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2023.

[35] J. Lin and F. Zhang. R 3 live: A robust, real-time, rgb-colored, lidar-inertial-visual tightly-coupled state estimation and mapping package. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10672–10678. IEEE, 2022.

[36] J. Liu, V. Hull, H. C. J. Godfray, D. Tilman, P. Gleick, H. Hoff, C. Pahl-Wostl, Z. Xu, M. G. Chung, J. Sun, et al. Nexus approaches to global sustainable development. *Nature Sustainability*, 1(9):466–476, 2018.

[37] J.-W. Liu, Y.-P. Cao, W. Mao, W. Zhang, D. J. Zhang, J. Keppo, Y. Shan, X. Qie, and M. Z. Shou. Devrf: Fast deformable voxel radiance fields for dynamic scenes. *Advances in Neural Information Processing Systems*, 35:36762–36775, 2022.

[38] L. Ma, X. Li, J. Liao, Q. Zhang, X. Wang, J. Wang, and P. V. Sander. Deblur-nerf: Neural radiance fields from blurry images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12861–12870, 2022.

[39] J. Maye, P. Furgale, and R. Siegwart. Self-supervised calibration for robotic systems. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 473–480. IEEE, 2013.

[40] 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. *Communications of the ACM*, 65(1):99–106, 2021.

[41] I. Mitchell, T. Cockerton, S. Hara, and C. Evans. Smerf: social media, ethics and risk framework. *Cyber Criminology*, pages 203–225, 2018.

[42] P. Moulon, P. Monasse, and R. Marlet. Adaptive structure from motion with a contrario model estimation. In *Computer Vision–ACCV 2012: 11th Asian Conference on Computer Vision, Daejeon, Korea, November 5-9, 2012, Revised Selected Papers, Part IV 11*, pages 257–270. Springer, 2013.

[43] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022.

[44] A. Noguchi, X. Sun, S. Lin, and T. Harada. Neural articulated radiance field. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5762–5772, 2021.

[45] L. Oth, P. Furgale, L. Kneip, and R. Siegwart. Rolling shutter camera calibration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1360–1367, 2013.

[46] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021.

[47] S. Peng, J. Dong, Q. Wang, S. Zhang, Q. Shuai, X. Zhou, and H. Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14314–14323, 2021.

[48] F. Ponchio and M. Dellepiane. Multiresolution and fast decompression for optimal web-based rendering. *Graphical Models*, 88:1–11, 2016.

[49] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[50] D. Rebain, W. Jiang, S. Yazdani, K. Li, K. M. Yi, and A. Tagliasacchi. Derf: Decomposed radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14153–14161, 2021.

[51] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart. Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4304–4311. IEEE, 2016.

[52] C. Reiser, S. Peng, Y. Liao, and A. Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *International Conference on Computer Vision (ICCV)*, 2021.

[53] K. Rematas, A. Liu, P. P. Srinivasan, J. T. Barron, A. Tagliasacchi, T. Funkhouser, and V. Ferrari. Urban radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12932–12942, 2022.

[54] J. Schauer and A. Nüchter. The peopleremover—removing dynamic objects from 3-d point cloud data by traversing a voxel occupancy grid. *IEEE robotics and automation letters*, 3(3):1679–1686, 2018.

[55] P. Schmuck, T. Ziegler, M. Karrer, J. Perraudin, and M. Chli. Covins: Visual-inertial slam for centralized collaboration. In *2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pages 171–176. IEEE, 2021.

[56] J. L. Schonberger and J.-M. Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[57] M. Schütz, S. Ohrhallinger, and M. Wimmer. Fast out-of-core octree generation for massive point clouds. In *Computer Graphics Forum*, volume 39, pages 155–167. Wiley Online Library, 2020.

[58] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 1, pages 519–528. IEEE, 2006.

[59] T. Shan, B. Englot, C. Ratti, and D. Rus. Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 5692–5698. IEEE, 2021.

[60] N. Snavely. Scene reconstruction and visualization from internet photo collections: A survey. *IPSJ Transactions on Computer Vision and Applications*, 3:44–66, 2011.

[61] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. 2006.

[62] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *International journal of computer vision*, 80:189–210, 2008.

[63] N. Snavely, S. M. Seitz, and R. Szeliski. Skeletal graphs for efficient structure from motion. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[64] W. Sun, E. Trulls, Y.-C. Tseng, S. Sambandam, G. Sharma, A. Tagliasacchi, and K. M. Yi. Pointnerf++: A multi-scale, point-based neural radiance field. *arXiv preprint arXiv:2312.02362*, 2023.

[65] C. Sweeney, V. Fragoso, T. Höllerer, and M. Turk. Large scale sfm with the distributed camera model. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 230–238. IEEE, 2016.

[66] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8248–8258, 2022.

[67] Z. Teed and J. Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021.

[68] H. Turki, D. Ramanan, and M. Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12922–12931, 2022.

[69] L. Wang, J. Zhang, X. Liu, F. Zhao, Y. Zhang, Y. Zhang, M. Wu, J. Yu, and L. Xu. Fourier plenoctrees for dynamic radiance field rendering in real-time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13524–13534, 2022.

[70] P. Wang, Y. Liu, Z. Chen, L. Liu, Z. Liu, T. Komura, C. Theobalt, and W. Wang. F2-nerf: Fast neural radiance field training with free camera trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4150–4159, 2023.

[71] C. Wu. Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 127–134. IEEE, 2013.

[72] X. Wu, J. Xu, X. Zhang, H. Bao, Q. Huang, Y. Shen, J. Tompkin, and W. Xu. Scanerf: Scalable bundle-adjusting neural radiance fields for large-scale scene rendering. *ACM Transactions on Graphics (TOG)*, 42(6):1–18, 2023.

[73] H. Xie, Z. Chen, F. Hong, and Z. Liu. Citydreamer: Compositional generative model of unbounded 3d cities. *arXiv preprint arXiv:2309.00610*, 2023.

[74] H. Xu, Z. Zhou, Y. Qiao, W. Kang, and Q. Wu. Self-supervised multi-view stereo via effective co-segmentation and data-augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3030–3038, 2021.

[75] H. Xu, Z. Zhou, Y. Wang, W. Kang, B. Sun, H. Li, and Y. Qiao. Digging into uncertainty in self-supervised multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6078–6087, 2021.

[76] Q. Xu, Z. Xu, J. Philip, S. Bi, Z. Shu, K. Sunkavalli, and U. Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022.

[77] J. Yang, J. M. Alvarez, and M. Liu. Self-supervised learning of depth inference for multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7526–7534, 2021.

[78] C. Yeshwanth, Y.-C. Liu, M. Nießner, and A. Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023.

[79] A. Yu, V. Ye, M. Tancik, and A. Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021.

[80] Z. Yu, A. Chen, B. Huang, T. Sattler, and A. Geiger. Mip-splatting: Alias-free 3d gaussian splatting. *arXiv:2311.16493*, 2023.

[81] J. Zhang, X. Liu, X. Ye, F. Zhao, Y. Zhang, M. Wu, Y. Zhang, L. Xu, and J. Yu. Editable free-viewpoint video using a layered neural representation. *ACM Transactions on Graphics (TOG)*, 40(4):1–18, 2021.

[82] K. Zhang, G. Riegler, N. Snavely, and V. Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020.

[83] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[84] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 371–378, 2001.