DRAGON: Drone and Ground Gaussian Splatting for 3D Building Reconstruction

Paper ID 24

Abstract—3D building reconstruction from imaging data is an important task for many applications ranging from urban planning to reconnaissance. Modern Novel View synthesis (NVS) methods like NeRF and Gaussian Splatting offer powerful techniques for developing 3D models from natural 2D imagery in an unsupervised fashion. These algorithms generally require input training views surrounding the scene of interest, which, in the case of large buildings, is typically not available across all camera elevations. In particular, the most readily available camera viewpoints at scale across most buildings are at near-ground (e.g., with mobile phones) and aerial (drones) elevations. However, due to the significant difference in viewpoint between drone and ground image sets, camera registration – a necessary step for NVS algorithms – fails. In this work we propose a method, DRAGON, that can take drone and ground building imagery as input and produce a 3D NVS model. The key insight of DRAGON is that intermediate elevation imagery may be extrapolated by an NVS algorithm itself in an iterative procedure with perceptual regularization, thereby bridging the visual feature gap between the two elevations and enabling registration. We compiled a semi-synthetic dataset of 9 large building scenes using Google Earth Studio, and quantitatively and qualitatively demonstrate that DRAGON can generate compelling renderings on this dataset compared to baseline strategies.

Index Terms—3D Building Reconstruction, Novel View Synthesis, 3D Gaussian Splatting, Multi-Elevation Reconstruction

1 Introduction

3D building reconstruction is an important task that is useful in a variety of applications from urban planning and monitoring to disaster relief and reconnaissance. Traditional building reconstruction methods assume a data modality such as aerial LiDAR [1], [2], [3] as input, which is expensive and offers limited viewpoint coverage. The proliferation of standard imaging sensors, along with powerful recent advances in Novel View Synthesis (NVS) algorithms like Neural Radiance Fields (NeRF) [4] and 3D Gaussian Splatting (3DGS) [5], offer the potential to develop 3D building models in a cheap and scalable manner. Natural images of buildings are most readily acquired at two elevations: near-ground (such as those taken by mobile phones) and aerial (such as those taken by drones), representing highly contrasting structural viewpoints (see Fig. 1). 3D reconstruction that can work with such sparse footage without needing intermediate elevation imagery would enable large-scale structural modeling for a variety of otherwise inaccessible scenarios. For this reason, in this study, we aim to develop a NVS-based method that can combine *only* drone and ground building imagery into a coherent 3D model of a building's outer structure.

Standard variants of NeRF and 3DGS algorithms assume a set of input 2D images covering a continuous "shell" of viewpoints around a scene. These algorithms first register the views together into a common reference system by finding matching features, and then combine their information into a 3D model via a rendering-based optimization. Unfortunately, the crucial registration step poses significant challenges when given only aerial and ground footage as input. As shown in the examples in Fig. 1, the same structure on a building can look completely different in scale and viewpoint across elevations. Indeed, we find that registration packages such as COLMAP [7] completely fail when given such data because of a lack of robust feature point matches. Hence, while aerial and ground building footage are widely accessible, they are apparently not immediately usable for 3D building reconstruction without further camera pose input metadata.

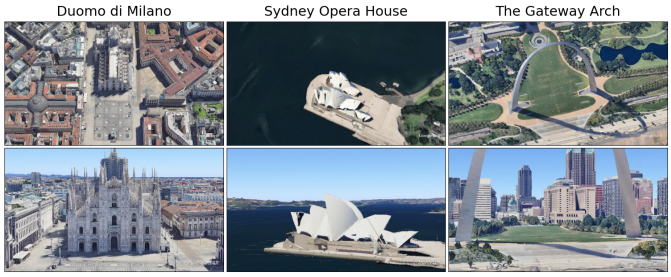


Fig. 1. **Sample drone and near-ground images from our collected Google Earth Studio dataset.** We show views of three large buildings from drone-level (top) and ground-level (bottom) elevations. These opposing elevations offer highly contrasting viewpoints of the physical structures, which can provide complementary information towards 3D modeling. However, the lack of easily matching visual features across elevations inhibits registration, a key step in novel view synthesis algorithms like NeRF [6] and 3D Gaussian Splatting [5].

In this work, we address the challenge of large-scale building reconstruction where only drone and near-ground image sets are available, *without* known camera poses. To do this, we draw on three insights. First, while registration methods fail given only drone and ground images, they are successful when intermediate elevation images are avail-

- *This paper is under review for ICCP 2024 and the PAMI special issue on computational photography. Do not distribute.*

able. Second, view rendering methods such as NeRF and 3DGS are capable of limited *extrapolation* beyond their training view range, likely due to inherent implicit regularization in their representations. Third, as shown in previous studies this extrapolation ability may be enhanced using perceptual regularization. We use these ideas to develop a framework, DRAGON (for DRone And GrOuNd Rendering). DRAGON begins by training a 3D model on aerial footage only, providing an initial rendering model with full context of the coarse layout of the structure. The algorithm then iteratively alternates between generating progressively lower-elevation imagery and updating its representation with previously generated views. DRAGON eventually reaches the lowest elevation (ground), at which point all of the real and generated views are used to perform registration and generate a final model. DRAGON is conceptually simple, and requires no further annotation or information per view to operate.

We demonstrate the effectiveness of DRAGON using the 3DGS rendering algorithm on a new semi-synthetic dataset we build using Google Earth Studio consisting of 9 building sites (see Fig. 2). Results first show that popular registration packages like COLMAP fail to register the drone and ground footage on any of these scenes. However, DRAGON aligns 97.47% of the input views (with average 0.01m position and 0.12° rotation errors). With camera poses obtained using DRAGON, we demonstrate that DRAGON produces high-quality renderings across all viewing angle/elevation scenarios. We show that adding perceptual regularization encoded by deep feature spaces like DreamSim [8] and OpenCLIP [9] further improves quantitative and qualitative performance. We conclude by discussing limitations and considerations when using DRAGON.

2 Related Work

2.1 Building Reconstruction from LIDAR and Aerial Footage

Several prior works estimate building structures from LIDAR footage [1], [2], [3], [10], [11], [12]. LIDAR directly provides 3D scene information, but is also expensive to acquire and therefore limited in availability. Several methods also try to infer 3D information of buildings from monocular (aerial) imagery alone [13], [14], [15]. Limited by the information from a single elevation, these methods offer coarse 3D details, such as building height and general shape. In contrast, we attempt to build full 3D rendering of outer building structures based on both aerial and ground images, without additional modalities. In addition, our technique builds on differentiable view rendering techniques like 3DGS, which is fundamentally different from these past approaches.

2.2 Neural/Differentiable Rendering

Neural Radiance Fields (NeRF) [6], [16], [17] are a family of deep neural network techniques that have brought a new wave of ideas and state-of-the-art results to view synthesis. NeRF reconstructs scene parameters including geometry, scattered radiance, and camera parameters [18] in an end-to-end fashion based on a simple reconstruction loss over the input views. NeRF uses neural networks to fit the scene parameters, which are capable of regressing on highly nonlinear, complex functions typical of real-world scenes. This leads to a continuous scene representation, which can be rendered immediately for any desired novel view simply by evaluating the representation for each pixel.

Recently, 3D Gaussian Splatting (3DGS) [5] has emerged as a compelling alternative to NeRF. 3DGS explicitly represents a scene with (millions of) 3D Gaussian objects with optimizable 3D location and appearance parameters. Crucially, these parameters are optimized in an end-to-end fashion using a reconstruction loss by "splatting" the Gaussians onto 2D planes. 3DGS has proven far faster to train for large scenes than NeRF. Due to this reason, we use 3DGS in our experiments.

2.3 Neural Rendering for Large Scenes

NeRF and 3DGS also have some demonstrated successes on large scenes. The first work in this space was Block-NeRF [19], which modeled city blocks from ground-level footage. BungeeNeRF [20] is the closest related study to ours, which reconstructs buildings using dense multi-elevation imagery from drone to ground elevations. That study pointed out that scale differences resulting from cameras positioned at varying altitudes can pose significant challenges to rendering. For example, the level of detail and spatial coverage varies significantly across different altitudes. These scaling problems are somewhat mitigated with follow-up NeRF variants [19], [21], [22], [23]. Recently, VastGaussian [23] extended 3DGS to handle large building scenes, mostly from aerial imagery. Unlike our problem setup, all of these algorithms assume a densely sampled set of input 2D views at orientations and elevations, and can not handle the case of combining only drone and ground footage into one coherent 3D model.

2.4 Registration and Camera Pose Estimation

A crucial first step for NVS algorithms is image registration, i.e., obtaining the camera poses for each input image. This is commonly achieved by running the Structure-from-Motion (SfM) [24] library COLMAP, designed for 3D reconstruction from 2D images. The process begins by extracting features from each image, followed by matching these features across multiple images to establish corresponding points [7]. The output of registration includes intrinsic camera parameters, camera poses (extrinsic parameters), and 3D point clouds. Other image registration and camera pose estimation algorithms also exist, such as SLAM-based methods [25], and deep learning models such as SuperGlue [26]. Some NeRF and 3DGS variants relieve the reliance on SfM prepocessing by incorporating camera pose estimation directly into the optimization framework [18], [27], [28]. These methods work reasonably for normal-sized objects and with sparse viewpoints that cover the full range of the scene, but do not work with large ranges of missing viewpoints, as in our problem setup.

3 Buildings-NVS Dataset

We introduce *Buildings-NVS*, a dataset comprising multi-elevation imagery from 9 building sites using Google Earth

TABLE 1
Camera altitudes/trajectory radii/target altitudes (all measured in meters) per elevation for each building in our collected dataset Buildings-NVS. The targeted altitude refers to the point where the perpendicular line from the center of the building intersects the plane at the targeted height where the camera is aimed. Each orbit from 5 different elevations consists of 61 images positioned with evenly spaced camera locations. We selected these values manually by focusing on relevant perceptual characteristics while keeping most of the structure in view.

Elevation \ Building	Arc de Triomphe	Colosseum	Duomo di Milano	Eiffel Tower	Himeji Castle	Piazza del Duomo	Space Needle	Sydney Opera House	The Gateway Arch
Ground	101/250/89	77/313/52	177/250/153	101/438/89	92/250/67	58/250/34	83/188/71	60/438/48	166/313/166
Mid 1	201/313/89	127/375/52	250/438/152	201/500/89	154/313/67	146/250/34	195/250/71	148/438/48	240/375/166
Mid 2	301/375/89	252/438/52	402/500/177	351/625/114	279/375/67	246/313/34	345/250/95	298/563/48	403/438/203
Mid 3	451/500/89	452/500/52	600/438/152	601/750/151	404/375/67	346/313/34	495/375/120	498/563/48	603/563/203
Drone	701/313/89	702/500/52	802/375/152	1000/375/177	604/313/67	596/313/34	695/313/145	748/500/48	803/375/203

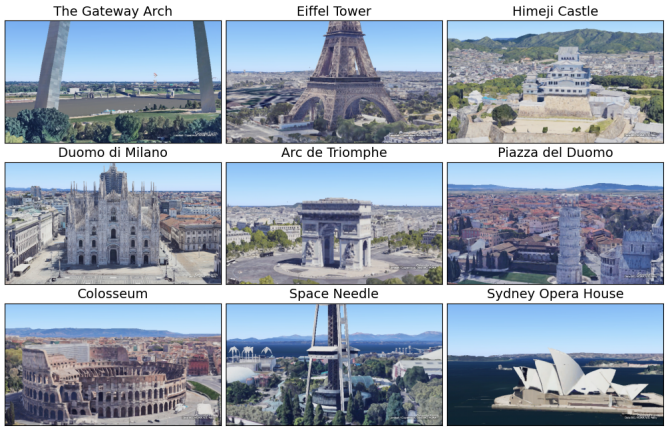


Fig. 2. **Visual depictions of the 9 buildings in our Buildings-NVS dataset.** These buildings have unique characteristics from height to architecture style and backgrounds.

Studio [1] (see Table 1 and Fig. 2). The 9 buildings are diverse in many aspects, including location (spanning four continents), height, architectural styles, backgrounds (e.g., water bodies, trees, other buildings), symmetry, and occlusions. For each building, we captured images over five camera elevations ranging from near-ground to high aerial elevations. For each elevation, we sampled 61 images evenly along a circle, with the camera pointed at a particular target 3D location which affects the vertical rotation of the camera. We manually selected the camera elevations, target elevations, and trajectory radii per building to focus on relevant perceptual characteristics while keeping most of the structure in view. We present these parameters in Table 1. Note that Google Earth Studio permits a different lower limit on camera elevation per location, leading to different ground elevations across buildings in our dataset. We partitioned these images into training and testing subsets by elevation (see Fig. 3. The training dataset consists of elevations X^0 (ground) and X^4 (drone), and the testing dataset comprises all elevations ($X^0 - X^4$). Each building has 122 training images and 305 testing images.

4 METHODS

4.1 Background on 3D Gaussian Splatting (3DGS)

In 3D Gaussian splatting, a set of 3D Gaussian objects is used to represent a scene. Initialized with point clouds from structure-from-motion [24], each Gaussian is defined spatially by:

$$G(x) = \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right), \qquad (1)$$

where μ and Σ represent the spatial average and the covariance matrix. Additionally, each Gaussian has an opacity factor o and a color c conditioned on viewing perspective. During rasterization, each 3D Gaussian is projected onto the 2D image space from a particular viewing angle via a projection matrix. The color of a pixel in a rendered image is obtained by alpha-blending N 2D Gaussians arranged in a depth-aware sequence from the nearest to the farthest relative to the camera viewpoint's perspective:

$$C = \sum_{i \in N} \tau_i \alpha_i c_i \quad \text{with} \quad \tau_i = \prod_{j=1}^{i-1}(1-\alpha_j), \qquad (2)$$

where α is a product of the opacity o and the likelihood of the pixel coordinate in image space. The Gaussian parameters are then optimized using a rendering loss function over the input 2D views:

$$L_{3DGS}(I, \hat{I}) = L_1(I, \hat{I}) + \lambda_{ssim} L_{\text{ssim}}(I, \hat{I}), \qquad (3)$$

where $L_1(\cdot,\cdot)$ measures pixelwise L1 distance, $L_{\text{ssim}}(\cdot,\cdot)$ measures SSIM [29] distance, and λ_{ssim} is a tradeoff hyperparameter.

4.2 DRAGON

We assume as input two sets of images X_{ground} and X_{drone}, depicting the building from ground and drone elevations (see Fig. 3). We assume these images are reasonably distributed at different viewing angles in order to give a full 360-degree coverage of the building. Our goal is to develop one 3D NVS algorithm (3DGS in our experiments) based on this imagery.

An obvious first approach to solve this problem is to simply run the NVS model on the provided views, described in Alg. 1 (BASIC). However, this requires camera poses as input, produced by a registration package such as COLMAP [7]. Such registration algorithms typically use Structure-from-Motion techniques [30], using a key assumption that the input set consists of overlapping images of the same object, taken from different viewpoints. However, in the case where only drone and near-ground image sets are available, visual overlap across sets is limited. Furthermore, even if there is visual overlap between image sets, high

1. https://www.google.com/earth/studio/

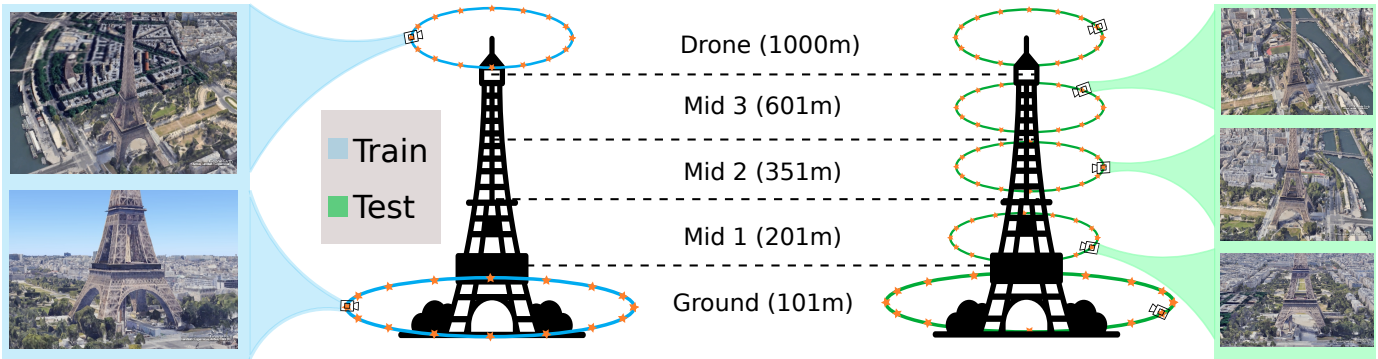
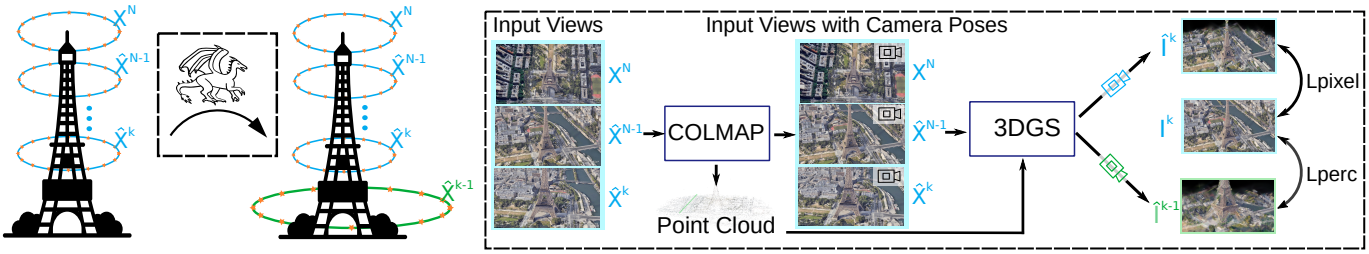


Fig. 3. **Train/test split of images for a given scene (Eiffel Tower) from our proposed Buildings-NVS dataset.** (Left) Training data covers only drone and ground elevations. (Right) Testing data covers all five elevations including drone and ground (due to space constrains, we provide sample images from middle three elevations in this figure).



a) Dragon iteration

b) Dragon internals

Fig. 4. **Overview of DRAGON's iterative pipeline.** (Left) We iteratively render viewpoints starting from aerial elevations towards ground. Given footage from elevations $X^N, \cdots X^k$, we generate views for elevation X^{k-1}. (Right) We feed the accumulated views into COLMAP to obtain the corresponding camera poses and point cloud, which are then passed to the 3DGS model. The model is trained using pixel-wise and perception-wise losses (Equation 4). Blue color denotes training images, while green color represents images rendered for the next elevation.

Algorithm 1 BASIC(model, $X_{\text{train}}, C_{\text{train}}, C_{\text{test}}$)

Given NVS model, train images X_{train},
train camera poses C_{train}, test camera poses C_{test}
1: model.train($X_{\text{train}}, C_{\text{train}}$)
2: X_{test} = model.test(C_{test})
3: **return** X_{test}

TABLE 2
Overview of notation used in DRAGON.

Symbol	Description
X_{train}	Train images
X_{test}	Test images
X^i	Images from elevation level i
X^0	Images from ground level
X^N	Images from drone level
X^{cum}	Images from accumulated elevation levels
I^i	Image sampled from elevation level i
$\hat{I}^i$	Novel image rendered from elevation level i
C_{train}	Camera poses for train images
C_{test}	Camera poses for test images
C^i	Camera poses for images from elevation level i
C^{cum}	Camera poses for images from accumulated elevation levels
model-1H	A "1-headed" model trained on ground truth images from a single elevation
model-2H	A "2-headed" model trained on ground truth images from both drone and ground elevations

disparity in feature quality makes it challenging to find correspondences.

We do find, however, that registration succeeds when intermediate level images are provided, serving as a "bridge" between the ground and aerial elevations (see Fig. 1). This insight motivates DRAGON, described in Alg. 2, which extrapolates intermediate elevation imagery from the drone altitude views in an iterative manner (see Sec. 4.2.1). After generating images across all elevations, DRAGON registers all views (real and generated) together (lines 7-8 of Alg. 2). Finally, using the original training views only along with their (now inferred) camera poses, we construct a 3D model using an NVS algorithm like 3DGS (line 9). We describe details of the iterative process and loss function in the following sections.

4.2.1 *Intermediate Elevation Image Generation*

We generate intermediate elevation images in an iterative manner, consisting of repeated steps of registration and extrapolated view rendering. We begin the process using only drone input images because they cover the entire context of the building and are therefore also easier to register to one another with few errors. As detailed in lines 3-5 of Alg. 2, we iteratively expand the viewing range of the NVS model by registering the current cumulative input image set, retraining the NVS model using these images and registered poses, generating new (extrapolated) views at camera poses in the next lowest elevation level, and adding these new views to the cumulative image set. Eventually, this process reaches the ground elevation, and stops.

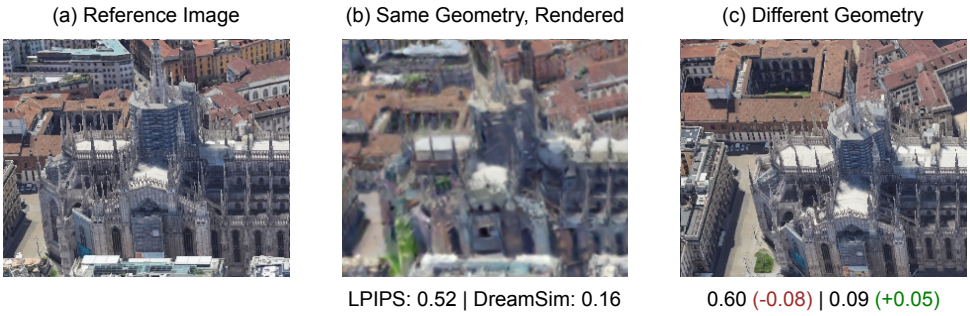


Fig. 5. **DreamSim vs. LPIPS** While LPIPS score in image (c) increased compared to same geometry in iamge (c), DreamSim score decreased, meaning Dreamsim perceive image (c) being more similar to the reference image compared to the image (b).This suggests that DreamSim places greater emphasis on perceptual image quality rather than geometric differences (spatial displacement), as long as the image being assessed shares content similarities with the reference image. Perceptual distance measured by DreamSim is more robust to geometry changes than LPIPS metric thus being more suitable for an auxiliary loss function as a regularizer.

Algorithm 2 DRAGON(model, X_{train}, N)

Given NVS model, $X_{\text{train}} = X^0 \cup X^N$,
number of elevation levels N

$X^{\text{cum}} = X^N$
for $i = N-1$ to 0 **do** % iterate over elevations
  $C^{\text{cum}} = \text{COLMAP}(X^{\text{cum}})$
  $\hat{X}^{i-1} = \text{BASIC}(\text{model}, X^{\text{cum}}, C^{\text{cum}}, C^{i-1})$
  $X^{\text{cum}} = X^{\text{cum}} \cup \hat{X}^{i-1}$
end for
$C_{\text{train}} = \text{COLMAP}(X^{\text{cum}})$
return BASIC(model, $X_{\text{train}}, C_{\text{train}}, C_{\text{test}}$)

4.2.2 Perceptual Regularization for Improved Extrapolation

Novel view synthesis methods like 3DGS work best when *interpolating* novel views that are nearby a subset of input training views. Conversely, 3DGS performance degrades rapidly when *extrapolating* target views that are significantly far from training viewpoints, as illustrated in Fig. 6. Rendering intermediate elevation imagery from drone and ground imagery is closer to extrapolation since the input views are so far apart, posing a significant challenge. Rendering errors affect both registration (propagating errors in landmarks), and view synthesis steps of DRAGON (see Fig. 7) (b) Basic.

To address this challenge, we propose adding perceptual regularization to the basic 3DGS loss function in Eq. 3 to encourage extrapolated rendered views to be perceptually similar to nearby 3DGS training views. Perceptual similarity metrics have been well-studied in computer vision, and popular current metrics are based on measuring distances in feature spaces encoded by deep neural networks. We consider two such models, DreamSim [8] and OpenCLIP. DreamSim encodes features particularly designed to correlate with humans on semantic visual perception tasks. We find in our own analysis (see Fig. 5) that DreamSim is superior to the widely used LPIPS metric for our task. LPIPS tends to perceive significant differences between image patches that have some geometrical distortions, even when they share the same semantic content. This is a negative property for our task, since we intend to compare images with slight changes to camera pose. In contrast, DreamSim prioritizes perceptual image similarity over geometric disparities, rendering it more suitable as an auxiliary loss function for DRAGON. We additionally consider OpenCLIP because it captures a different type of semantically-guided representation to DreamSim based on language. Such vision-language models have been successfully shown to provide regularization for NVS methods, including NeRF-based approaches [31], [32], [33], [34] as well as diffusion-based ones [35], [36], [37].

During iteration i of Algorithm 2, we sample an image I^k from a randomly chosen elevation k, where $N < k \leq i$, along with its corresponding predicted image $\hat{I}^k$, and a predicted image $\hat{I}^{k+1}$ from the previous lower elevation. Our loss function is:

$$\begin{aligned} L_{DRAGON}(I^k, \hat{I}^k, \hat{I}^{k+1}) = L_{3DGS}(I^k, \hat{I}^k) \\ + \lambda_{DS} L_{\text{DS}}(I^k, \hat{I}^{k+1}) \\ + \lambda_{CLIP} L_{\text{CLIP}}(I^k, \hat{I}^{k+1}) \end{aligned} \quad (4)$$

where $L_{DS}(\cdot,\cdot)$ and $L_{\text{CLIP}}(\cdot,\cdot)$ are distances computed using DreamSim and CLIP, and λ_{DS} and λ_{CLIP} are tradeoff hyperparameters.

5 Experiments

5.1 Implementation Details

5.1.1 Registration

We use COLMAP for registration because it is the most widely used registration package, and because its GPU-accelerated SIFT [38] feature extraction offers efficient performance. We used an exhaustive matcher between each image pair.

5.1.2 DRAGON and Baseline Variants

Since the basic 3DGS model cannot register both drone and ground images together, we explore two classes of methods: 1-Headed, *i.e.* those trained on images from a single elevation, and 2-Headed, *i.e.* those trained on both drone and ground images.

Basic-Drone is trained using vanilla 3DGS (Eq. 3) on drone images registered with COLMAP.

Basic-Ground is trained using vanilla 3DGS (Eq. 3) on ground images registered with COLMAP.

Oracle D&G is trained using vanilla 3DGS (Eq. 3) with ground and drone elevation images, utilizing ground truth camera poses.

TABLE 3
Quantitative evaluation on drone and ground registration. The percentage of registered images averaged over buildings when estimating camera poses separately from the drone and ground level image sets, as well as when estimating camera poses by combining them. It shows that the percentage of registered images higher when using our iterative scheme, and using perceptual regularizer help in registration. Drone and ground imagery sets may not be registered together at once since they have significant scale variations in levels of detail and field of view.

		1-HEADED		2-HEADED		
		COLMAP	COLMAP	COLMAP	DRAGON vanilla	DRAGON DreamSim
		drone-only	ground-only	drone&ground		
Matched (%) ↑		100.00	79.58	50.00	88.00	**97.47**
Errors for matched (Avg/Std)	rotation (°) ↓	0.76/0.68	19.13/2.39	0.69/0.60	0.15/0.12	**0.12/0.10**
	position (m) ↓	0.02/0.02	0.50/0.05	0.02/0.02	0.02/0.02	**0.01/0.01**

Oracle All is trained using vanilla 3DGS (Eq. 3) with images from all elevations, including ground, drone, and intermediate levels. Ground truth camera poses are used for these images.

Dragon Vanilla is trained using vanilla 3DGS (Eq. 3) with ground and drone elevation images, along with camera poses obtained from our proposed iterative registration scheme.

Dragon-2H DreamSim is trained with images from ground and drone elevations and using 3DGS equipped with the L_{DS} auxillary loss (Eq. 4 with λ_{CLIP} set to 0). We empirically determined the value of λ_{DS} to be 0.01 based on the DreamSim distance between images from adjacent elevations. We train Dragon-2H Dreamsim for 30k iterations during each registration iteration and for 30k iterations the final rendering. We used en ensemble model of DreamSim which uses DINO ViT-B/16 [39], CLIP ViT-B/16 B/32 [40], and OpenCLIP ViT-B/32 [9]. Additionally, it utilizes camera poses obtained from our proposed iterative registration scheme.

Dragon-2H DreamSim+CLIP. We train Dragon-2H DreamSim for 25k iterations for the final rendering and finetune the model by setting λ_{CLIP} to 0.01 in Eq. 4 for an additional 5k iterations. We used ViT-G/14 model of OpenCLIP which was pretrained on the LAION-2B dataset.

5.1.3 NVS model

We adopt 3DGS [5] as a novel view synthesis method for extrapolating images in iterative registration due to its significantly faster training speed, inference time and reconstruction quality compared to NeRF models with comparable rendering quality [17], [41], [42].

5.1.4 Hyperparameters

Unless noted otherwise, we follow the experimental setup in the 3DGS study [5]: we reset opacity every 3k iterations and use densification every 100 iterations. The spherical harmonics coefficients, which encode the appearance of each Gaussian splat, are set to 0.0025. Additionally, opacity, scaling, and rotation parameters are set to 0.05, 0.005, and 0.001. For optimizing the 3D position of the Gaussian splats, we employ a learning rate schedule ranging from 1.6×10^{-4} to 1.6×10^{-6} and use the Adam optimizer [43]. λ_{ssim} from the loss function (Eq. 2) was set to 0.2. Our iterative registration scheme incorporates three intermediate levels between drone and ground levels resulting in four registration iterations.

5.1.5 Training Time

We train all methods on NVIDIA A100 GPUs. Using a single GPU, training any DRAGON method on a building requires roughly 30 minutes, with an inference time of 8 rendered views per second.

5.2 Evaluation

5.2.1 Registration Evaluation

To evaluate obtained drone and ground camera poses, we need ground truth camera poses. We establish a pseudo-ground truth registration, we densely cover the building across elevations, supplementing additional images at intermediate levels and run COLMAP to get a dense camera poses. We calculated the percentage of registered matched images and the position and rotation errors for the matched images. Image registration entails a variable coordinate system, influenced by the position of keypoints. Thus, aligning the coordinate system is a prerequisite for accurate error computation against pseudo ground truth [44], [45].

5.2.2 Reconstruction Evaluation Metrics

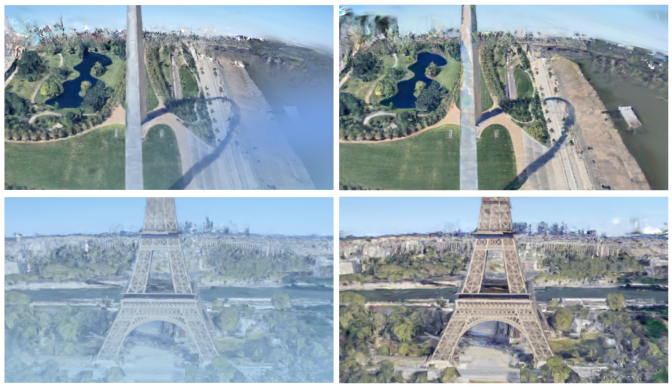
To quantitatively evaluate reconstruction quality, we use the following metrics: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM) [29], and Learned Perceptual Image Patch Similarity (LPIPS) [46]. Given ground truth image I and a rendered view $\hat{I}$, PSNR is defined as $\text{PSNR}(I, \hat{I}) = 10 \cdot \log_{10}\left(\text{MAX(I)}^2/\text{MSE}(I, \hat{I})\right)$, where $\text{MAX}(I)$ is the largest pixel value of image I and MSE denotes pixel-wise mean squared error. Higher PSNR values indicate greater similarity between images. SSIM assesses structural similarity considering luminance, contrast, and structure, making it less sensitive to color or brightness changes and perception aligned. Higher SSIM values indicate greater similarity between images. LPIPS measures the distance between overlapping patches in the input and reconstructed images in a deep feature space. Lower LPIPS values indicate higher perceptual similarity.

5.3 Registration Results

We first demonstrate that DRAGON enables accurate registration across drone and ground views. Table 3 presents a quantitative comparison between our registration pipeline and COLMAP applied directly on drone and ground data. While COLMAP manages to register drone images, it fails entirely to register ground images due to significant scale variations in levels of detail and field of view. In contrast,

TABLE 4
Quantitative evaluation of view synthesis methods. We report PSNR, SSIM, and LPIPS metrics averaged across all nine scenes, for three distinct elevation groups: 'drone and ground' (training data), 'mid elevations' (unseen data), and 'all elevations' (combining all data). Comparison is made between 1-headed methods, trained solely on images from one elevation, and 2-headed methods, trained using both ground and drone images.

	ground and drone			mid elevations			all elevations		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
2-HEADED METHODS									
DRAGON-2H Vanilla	25.68	**0.85**	**0.17**	17.38	0.54	0.40	20.70	0.66	0.31
DRAGON-2H DreamSim	**25.77**	**0.85**	**0.17**	**18.11**	**0.57**	**0.37**	**21.17**	**0.68**	**0.29**
DRAGON-2H DreamSim+CLIP	25.76	**0.85**	**0.17**	18.03	**0.57**	**0.37**	21.12	**0.68**	**0.29**
1-HEADED METHODS (ground)									
Basic	15.97	0.47	0.45	9.35	0.24	0.65	12.00	0.33	0.57
DRAGON-1H Vanilla	17.55	0.54	0.41	9.02	0.26	0.66	12.43	0.37	0.56
DRAGON-1H	**18.60**	**0.55**	**0.38**	**11.72**	**0.31**	**0.57**	**14.47**	**0.40**	**0.49**
1-HEADED METHODS (drone)									
Basic	**17.07**	**0.55**	0.37	17.25	**0.52**	0.41	17.17	**0.53**	0.39
DRAGON-1H Vanilla	17.06	**0.55**	0.37	17.30	**0.52**	0.41	**17.21**	**0.53**	0.39
DRAGON-1H	**17.07**	**0.55**	**0.38**	**17.31**	0.51	**0.42**	**17.21**	**0.53**	**0.40**
1-HEADED VS 2-HEADED									
Basic ground	15.97	0.47	0.45	9.35	0.24	0.65	12.00	0.33	0.57
Basic drone	17.07	0.55	0.37	17.25	0.52	0.41	17.17	0.53	0.39
DRAGON-2H Vanilla	25.68	**0.85**	**0.17**	17.38	0.54	0.40	20.70	0.66	0.31
DRAGON-2H DreamSim	**25.77**	**0.85**	**0.17**	**18.11**	**0.57**	**0.37**	**21.17**	**0.68**	**0.29**
DRAGON-2H DreamSim+CLIP	25.76	**0.85**	**0.17**	18.03	**0.57**	**0.37**	21.12	**0.68**	**0.29**
ORACLE METHODS									
Oracle D&G	**26.71**	**0.88**	**0.15**	17.87	0.59	0.37	21.41	0.70	0.28
Oracle All	25.04	0.83	0.18	**25.31**	**0.84**	**0.18**	**25.20**	**0.84**	**0.18**



(a) without Perceptual Regularization (b) with Perceptual Regularization

Fig. 6. **Perceptual regularization reduces prominent extrapolation artifacts.** Shown here are rendering results for the The GateWay Arch and Eiffel Tower at an intermediate altitude. (a) When training 3DGS on drone and ground image sets without additional regularization, the rendering at intermediate altitude exhibits haziness and artifacts throughout the image. (b) The incorporation of perceptual regularization using DreamSim [8] results in a cleaner rendering.

our approach achieves near-perfect registration for both drone and ground images. Additionally, our proposed iterative scheme results in significantly lower rotation and position errors.

5.4 Quantitative Results

We categorize the test elevations into three groups: 'drone and ground', representing ground truth images; 'mid elevations', comprising images from unseen intermediate elevations; and 'all elevations', which combines images from every elevation. In Table 4, we present the average PSNR, SSIM, and LPIPS metrics across all nine scenes for these three groups. We note the following observations from the table:

5.4.1 2-Headed Methods

We first compare our three reconstruction approaches with camera poses obtained from our iterative registration scheme using both ground and drone images. Results demonstrate that incorporating auxiliary perceptual losses based on DreamSim and OpenClip yield additional improvements across all metrics for mid elevations, resulting in overall reconstruction enhancement.

5.4.2 1-Headed Methods

Given the limitations of the basic 3DGS in registering both ground and drone images simultaneously, we also report the performance of 1-headed methods. These methods are trained exclusively on either drone images or ground images, showcasing their individual limitations. We find that the Basic ground-only approach is particularly susceptible to occlusions caused by nearby buildings, resulting in suboptimal registration rates and significant registration errors, and, as a result, suboptimal reconstruction performance. Incorporating the DreamSim auxiliary loss yields substantial enhancements across all elevation groups.

5.4.3 1-Headed vs 2-Headed

Here, we highlight the full potential of our registration and reconstruction pipeline. Due to registration issues, 3D Gaussian Splatting method cannot readily take advantage of both drone and ground images. Conversely, our 2-headed approaches leverage both drone and ground images and produce much better reconstruction results.

5.4.4 Oracle Methods

Finally, we investigate an oracle scenario where camera poses are given as part of the train set. Remarkably, our

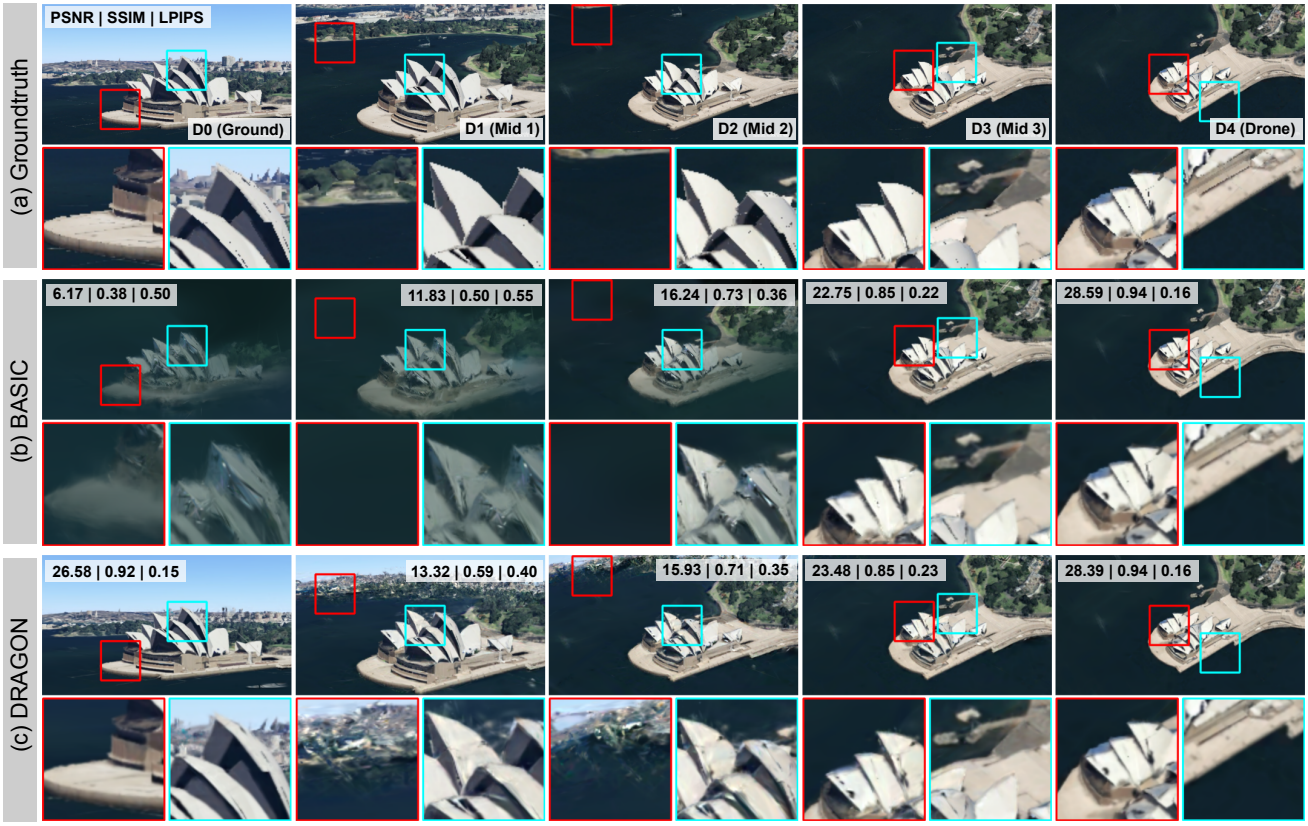


Fig. 7. **Rendering results for the Sydney Opera House.** We present ground truth images (a, top row), along with renderings produced by BasicDRAGON (b, middle), and name (c). Each column corresponds to a different elevation, from left to right: Ground, Mid 1, Mid 2, Mid 3, and Drone. We also provide metrics (PSNR, SSIM, and LPIPS) directly on the images. The green and cyan boxes below the full images are zoomed in patches. In general, DRAGON outperforms Basic qualitatively, particularly for lower elevations. For Mid 2, the image rendered by Basic exhibits higher PSNR and SSIM scores compared to the image rendered by DRAGON. But upon closer visual inspection, it is clear that DRAGON achieves a more accurate reconstruction of the opera house's true appearance and structure, despite some hallucinatory effects at the top of the image, which may have impacted the quantitative metrics negatively.

2-headed method, without known camera poses, achieves comparable performance by using estimated camera poses to the oracle Drone & Ground (D&G) approach. For reference, we also report the reconstruction results of an oracle-all approach, which has access to both images and camera poses from all elevations. Oracle All surpasses all methods by a significant margin, underscoring the need for continued research to bridge this performance gap.

5.5 Qualitative Results

Fig. 7 and Fig. 8 provide a qualitative comparison between DRAGON and Basic using two scenes: Sydney Opera House and Duomo di Milano. DRAGON consistently yields more visually appealing results. While Basic is trained solely on drone images, resulting in slightly better PSNR scores at that specific elevation, DRAGON demonstrates superior performance across other elevations, particularly near ground levels. For instance, in the case of the Sydney Opera House, Basic blends the Opera building with the surrounding water area, resulting in an underwater-like rendering for near-ground elevations. This color mixing does not occur with our DRAGON approach. Additionally, DRAGON produces sharper edges for the opera petals and better preserves the structure of the building.

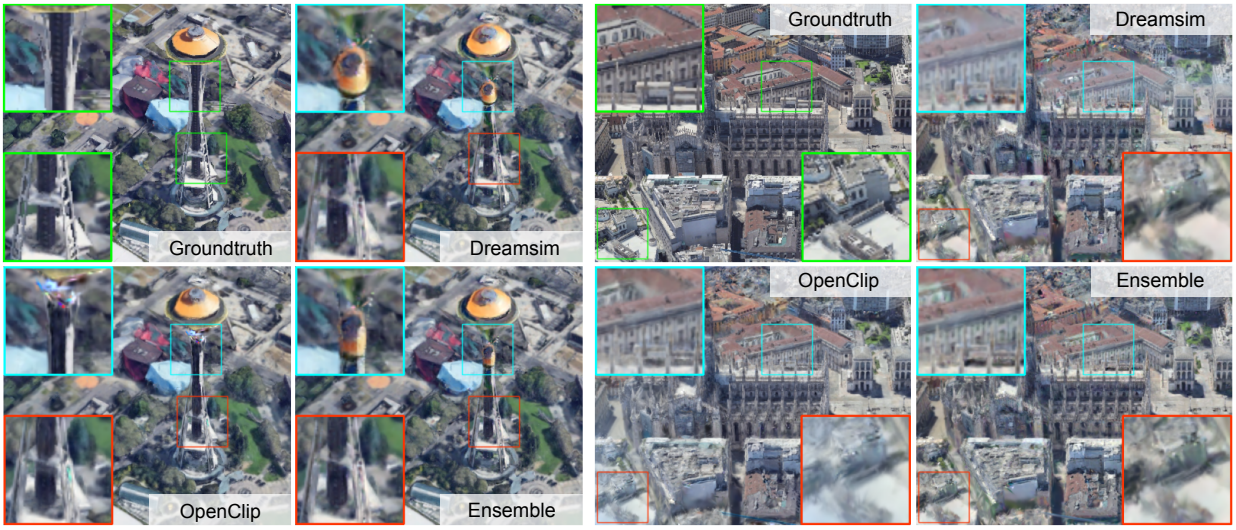
Similarly, in the case of Duomo di Milano, Basic's performance severely deteriorates near the ground elevation. Here, the rich geometric details of the multiple buildings are completely lost in Basic's reconstruction. This is not the case with DRAGON, where we can recover the architectural details of the buildings. Even at mid-elevations, DRAGON produces sharper details, while Basic hallucinates a black background at the top of the rendered images

We further illustrate the qualitative difference between Dragon-2H DreamSim and Dragon-2H DreamSim+CLIP in Fig. 9. When employing DreamSim as our sole perceptual regularizer, occasional high-frequency artifacts are introduced, as exemplified by the Space Needle image. When fine-tuning our model with CLIP, sharper edges are achieved in the center of the image. However, it is noticeable that additional haze appears in the lower corners, as observed in the Duomo Di Milano image.

6 Discussion and Conclusion

Results first demonstrate that DRAGON offers a significant advantage in terms of registering drone and ground footage together at once and retrieving accurate camera parameters. In contrast, running a registration package like COLMAP on its own essentially fails in matching features across sets with

Fig. 8. **Rendering results for Duomo di Milano.** We present ground truth images (a, top row), along with renderings produced by BasicDRAGON (b, middle), and name (c). Each column corresponds to a different elevation, from left to right: Ground, Mid 1, Mid 2, Mid 3, and Drone. We also provide metrics (PSNR, SSIM, and LPIPS) directly on the images. In general, DRAGON outperforms Basic qualitatively, particularly for lower elevations. The green and cyan boxes below the full images are zoomed in patches. For Mid 3, Basic has a higher PSNR score than DRAGON, but the latter produces images with more distinct details, particularly in the floor patterns and architecture.



(a) Space Needle (b) Duomo de Milano

Fig. 9. **Qualitative comparison of DRAGON when using DreamSim and CLIP regularizers for Space Needle and Duomo di Milano**. Left: DreamSim introduces artifacts, such as the duplication of the head of the Space Needle (blue box), while CLIP maintains semantic consistency without introducing any additional artifacts. Right: CLIP yields sharper edges of the buildings in the center (blue box), but it introduces additional haze in the lower corner of the image (red box). Conversely, the reconstruction in the corner area with DreamSim does not exhibit this haze.

significant disparities. Rendering results compared to various baseline algorithms show that DRAGON also provides benefits in terms of rendering quality. In particular, even though we estimated camera poses without any additional information, we achieve comparable or better rendering quality compared to Oracle D&G, which assumes perfect

registration.

We observe that perceptual regularization is a key step in improving rendering quality. Without regularization, 3DGS often introduces dramatic artifacts into extrapolated images, such as sky-like features (see Fig. 9) and sharp floaters due to sharp transitions in viewpoints from drone to ground or vice versa. While quantitative metrics do not suggest that combining CLIP and DreamSim losses offers an advantage to using DreamSim alone, we observe in the qualitative results (Fig. 9) that CLIP and DreamSim capture mutually exclusive visual phenomenon. For example, CLIP seems to better focus details on sharper edges of the buildings near the image center, but it introduces additional haze in the lower image corners. Further human perceptual studies may provide insight into how these perceptual losses interact with one another.

Interestingly, Oracle D&G achieves PSNR/SSIM scores of only roughly 17.87/0.59 on mid elevations, suggesting that even with perfect registration, the standard 3DGS framework exhibits rendering inaccuracies. Some of these shortcomings have been explored in a recent work, Vast-Gaussian [23]. The focus of our work was not to improve the core 3DGS framework, but to show how an NVS algorithm like 3DGS may be used to perform 3D modeling on large scenes with aerial and ground footage. Our method is agnostic to the exact NVS algorithm used and makes no specific assumptions tied to 3DGS.

While we focused this study on large building reconstruction, the problem of missing viewpoint region chunks (known as the "missing cone" problem in tomography [47]) is a general challenge for NVS methods. The iterative strategy of DRAGON may be used for other missing cone scenarios. However, one characteristic specific to building reconstruction that helps the iterative approach is that aerial footage offers a coarse, global context for the structure which may then be refined as the camera lowers in elevation.

There are several limitations of this work. First, DRAGON is a semi-synthetic dataset which allows for ideal and precise acquisition conditions. In contrast, real imagery will have artifacts and imprecise camera positioning. Second, DRAGON currently requires a prespecified series of camera poses at intermediate elevations at which to perform novel view synthesis during the iterative process. We currently require these as inputs because it is not trivial to specify them apriori, before a common global coordinate system is established via drone-ground registration. A future direction is to compute these trajectories directly from the drone/ground footage. Third, as shown in Fig. 9, our perceptual regularization functions do not completely remove artifacts, and can even inject certain new high-frequency artifacts. This is a well-known shortcoming of using deep neural networks as loss functions for image synthesis [48], [49].

Finally, measuring perceptual quality of renderings is a difficult challenge on its own. As shown in Fig. 7, metrics like PSNR/SSIM/LPIPS can be swayed by certain details, particularly those in the background, that may not be meaningful to understand the quality of the building reconstruction alone. One way of accounting for this in the future is to perform human perceptual studies, or design metrics that isolate the structure of interest during evaluation.

References

[1] L.-C. Chen, T.-A. Teo, J.-Y. Rau, J.-K. Liu, and W.-C. Hsu, "Building reconstruction from lidar data and aerial imagery," in *Proceedings. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS'05.*, vol. 4. IEEE, 2005, pp. 2846–2849.

[2] A. F. Elaksher, J. S. Bethel *et al.*, "Reconstructing 3d buildings from lidar data," *International Archives Of Photogrammetry Remote Sensing and Spatial Information Sciences*, vol. 34, no. 3/A, pp. 102–107, 2002.

[3] G. Forlani, C. Nardinocchi, M. Scaioni, P. Zingaretti *et al.*, "Building reconstruction and visualization from lidar data," *International Archives Of Photogrammetry Remote Sensing And Spatial Information Sciences*, vol. 34, no. 5/W12, pp. 151–156, 2003.

[4] 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*, vol. 65, no. 1, pp. 99–106, 2021.

[5] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, 2023.

[6] 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," in *European conference on computer vision*. Springer, 2020, pp. 405–421.

[7] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[8] S. Fu, N. Tamir, S. Sundaram, L. Chai, R. Zhang, T. Dekel, and P. Isola, "Dreamsim: Learning new dimensions of human visual similarity using synthetic data," *arXiv preprint arXiv:2306.09344*, 2023.

[9] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, and J. Jitsev, "Reproducible scaling laws for contrastive language-image learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2818–2829.

[10] J. Huang, J. Stoter, R. Peters, and L. Nan, "City3d: Large-scale building reconstruction from airborne lidar point clouds," *Remote Sensing*, vol. 14, no. 9, p. 2254, 2022.

[11] R. Wang, "3d building modeling using images and lidar: A review," *International Journal of Image and Data Fusion*, vol. 4, no. 4, pp. 273–292, 2013.

[12] Y. Zheng, Q. Weng, and Y. Zheng, "A hybrid approach for three-dimensional building reconstruction in indianapolis from lidar data," *Remote Sensing*, vol. 9, no. 4, p. 310, 2017.

[13] S. Chen, L. Mou, Q. Li, Y. Sun, and X. X. Zhu, "Mask-height r-cnn: An end-to-end network for 3d building reconstruction from monocular remote sensing imagery," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*. IEEE, 2021, pp. 1202–1205.

[14] W. Li, L. Meng, J. Wang, C. He, G.-S. Xia, and D. Lin, "3d building reconstruction from monocular remote sensing images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 548–12 557.

[15] B. Xu and C. Liu, "A 3d reconstruction method for buildings based on monocular vision," *Computer-Aided Civil and Infrastructure Engineering*, vol. 37, no. 3, pp. 354–369, 2022.

[16] 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 (ICCV)*, October 2021, pp. 5855–5864.

[17] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM transactions on graphics (TOG)*, vol. 41, no. 4, pp. 1–15, 2022.

[18] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, "Barf: Bundle-adjusting neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5741–5751.

[19] 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*, 2022, pp. 8248–8258.

[20] Y. Xiangli, L. Xu, X. Pan, N. Zhao, A. Rao, C. Theobalt, B. Dai, and D. Lin, "Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering," in *European conference on computer vision*. Springer, 2022, pp. 106–122.

[21] Y. Li, L. Jiang, L. Xu, Y. Xiangli, Z. Wang, D. Lin, and B. Dai, "Matrixcity: A large-scale city dataset for city-scale neural rendering and beyond," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 3205–3215.

[22] 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*, 2022, pp. 12 922–12 931.

[23] J. Lin, Z. Li, X. Tang, J. Liu, S. Liu, J. Liu, Y. Lu, X. Wu, S. Xu, Y. Yan *et al.*, "Vastgaussian: Vast 3d gaussians for large scene reconstruction," *arXiv preprint arXiv:2402.17427*, 2024.

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

[25] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*. Springer, 2014, pp. 834–849.

[26] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 4938–4947.

[27] Y. Fu, S. Liu, A. Kulkarni, J. Kautz, A. A. Efros, and X. Wang, "Colmap-free 3d gaussian splatting," *arXiv preprint arXiv:2312.07504*, 2023.

[28] Z. Wang, S. Wu, W. Xie, M. Chen, and V. A. Prisacariu, "Nerf–: Neural radiance fields without known camera parameters," *arXiv preprint arXiv:2102.07064*, 2021.

[29] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[30] K. Häming and G. Peters, "The structure-from-motion reconstruction pipeline–a survey with focus on short image sequences," *Kybernetika*, vol. 46, no. 5, pp. 926–937, 2010.

[31] A. Jain, M. Tancik, and P. Abbeel, "Putting nerf on a diet: Semantically consistent few-shot view synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5885–5894.

[32] C. Wang, M. Chai, M. He, D. Chen, and J. Liao, "Clip-nerf: Text-and-image driven manipulation of neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3835–3844.

[33] B. Y. Feng, S. Jabbireddy, and A. Varshney, "Viinter: View interpolation with implicit neural representations of images," in *SIGGRAPH Asia 2022 Conference Papers*, 2022, pp. 1–9.

[34] S. Lee and J. Lee, "Posediff: Pose-conditioned multimodal diffusion model for unbounded scene synthesis from sparse inputs," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 5007–5017.

[35] C. Deng, C. Jiang, C. R. Qi, X. Yan, Y. Zhou, L. Guibas, D. Anguelov *et al.*, "Nerdi: Single-view nerf synthesis with language-guided diffusion as general image priors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 20 637–20 647.

[36] A. Jain, B. Mildenhall, J. T. Barron, P. Abbeel, and B. Poole, "Zero-shot text-guided object generation with dream fields," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 867–876.

[37] R. Liu, R. Wu, B. Van Hoorick, P. Tokmakov, S. Zakharov, and C. Vondrick, "Zero-1-to-3: Zero-shot one image to 3d object," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9298–9309.

[38] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[39] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9650–9660.

[40] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[41] 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*, 2022, pp. 5470–5479.

[42] 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*, 2022, pp. 5501–5510.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[44] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, "Local light field fusion: Practical view synthesis with prescriptive sampling guidelines," *ACM Transactions on Graphics (TOG)*, 2019.

[45] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, "Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 11, pp. 2233–2246, 2012.

[46] 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*, 2018, pp. 586–595.

[47] J. Lim, K. Lee, K. H. Jin, S. Shin, S. Lee, Y. Park, and J. C. Ye, "Comparative study of iterative reconstruction algorithms for missing cone problems in optical diffraction tomography," *Optics express*, vol. 23, no. 13, pp. 16 933–16 948, 2015.

[48] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[49] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*. Springer, 2016, pp. 694–711.