A Appendix

A.1 Detailed Literature Review

Monocular Depth Estimation. Recently, monocular depth estimation has advanced rapidly towards foundation-scale models. Metric3D (Yin et al., 2023; Hu et al., 2024) proposes a canonical camera space transformation to solve the depth ambiguity caused by various focal lengths. Unidepth (Piccinelli et al., 2024) introduces pseudo-spherical output space representation to disentangle camera and depth representations, and a self-prompting camera module to support camera-free inference. Depth Anything (Yang et al., 2024a) and its successor Depth Anything V2 (Yang et al., 2024b) establish foundation models for monocular depth estimation through large-scale pretraining. Depth Pro (Bochkovskii et al., 2024) synthesizes high-resolution depth predictions based on a multi-scale vision transformer and an edge gradient loss. Marigold (Ke et al., 2024) presents a fine-tuning protocol for Stable Diffusion and a model for affine-invariant depth estimation. Geowizard (Fu et al., 2024) also distills the rich knowledge in the pre-trained Stable Diffusion. It proposes a geometry switcher that jointly produces depth and normal using a single model. DepthFM (Gui et al., 2025) presents a flow matching approach that improves sampling, data fidelity, training, and data efficiency. VGGT (Wang et al., 2025a) introduces a feed-forward neural network that can directly estimate all key 3D scene properties, including depth estimation. MoGe (Wang et al., 2025b) introduces an affine-invariant point map representation, an efficient point map alignment solver, and a multi-scale geometry loss for accurate monocular geometry estimation of open-domain images. Map Anything (Keetha et al., 2025) unifies local estimates into a global metric frame by using a factored representation of multi-view geometry (depth maps, ray maps, poses, and a global scale factor).

Monocular Depth Completion. Depth completion aims to predict a dense depth map from an RGB image guided by a sparse depth map. NLSPN (Park et al., 2020) proposes a non-local spatial propagation module with confidence-aware affinity normalization to enhance relevant interactions and mitigate errors in depth propagation. SpAgNet (Conti et al., 2023) injects sparse depth points into a Scale-and-Place module instead of convolutions to handle uneven and sparse input distributions more robustly. CompletionFormer (Zhang et al., 2023) proposes a Joint Convolution-Attention and Transformer block that integrates local connectivity with global context. VPP4DC (Bartolomei et al., 2024) leverages the generalization capability of modern stereo networks to address depth completion by processing fictitious stereo pairs generated through a virtual pattern projection paradigm. BP-Net (Tang et al., 2024) propagates depth at the earliest stage to avoid directly convolving on sparse data. OMNI-DC (Zuo et al., 2024) introduces a multi-resolution depth integrator to handle extremely sparse inputs and employs a Laplacian loss to better model training ambiguity. Marigold-DC (Viola et al., 2024) builds on a pretrained latent diffusion model and injects the depth observations as test-time guidance via an optimization scheme that runs in tandem with the iterative inference of denoising diffusion. PromptDA (Lin et al., 2025) utilizes a low-cost LiDAR as a prompt to guide the Depth Anything model, enabling accurate metric depth output with resolutions of up to 4K. Prior Depth Anything (Wang et al., 2025d) introduces a coarse-to-fine pipeline that integrates precise but incomplete metric depth with complete but relative geometric predictions.

Depth Estimation with 3DGS. 3D Gaussian Splatting (3DGS) represents a cutting-edge paradigm in 3D reconstruction, where contemporary approaches increasingly exploit monocular depth estimation to enhance reconstruction fidelity and geometry. DepthSplat (Xu et al., 2025) leverages pre-trained monocular depth features for high-quality 3D Gaussian splatting and demonstrates its use as an unsupervised pre-training objective for depth models. CDGS (Zhang et al., 2025) leverages multi-cue confidence maps from monocular depth and sparse Structure-from-Motion depth to adjust supervision, thereby enhancing adaptive 3D Gaussian splatting. Mode-GS (Lee et al., 2024) integrates pixel-aligned anchors from monocular depth and generates Gaussian splats around them via residual-form Gaussian decoders. DHGS (Deng et al., 2025) combines 3D Gaussian splatting with depth-supervised learning using homogeneous coordinate embedding and adaptive monocular-SfM depth fusion, resolving scale ambiguity in distant views and enhancing local geometry via confidence-aware loss weighting. RDG-GS (Zhan et al., 2025) utilizes relative depth guidance to refine the Gaussian field, steering it towards view-consistent spatial geometric representations. CODN-GS (Hu et al., 2025) employs a normal-depth-normal transformation for accurate geometric feature capture and uses robust monocular depth supervision refined through global and local adjustments. 3DGS-Enhancer (Liu et al., 2024a) leverages 2D video diffusion priors to tackle 3D

view consistency by enforcing temporal consistency within video generation. Chung et al. (Chung et al., 2024) utilize an adjusted depth map from a pre-trained monocular model, which is aligned with sparse Structure-from-Motion points as a geometric reference. NexusGS (Zheng et al., 2025) leverages optical flow and camera poses to generate accurate depth maps, ensuring dense point cloud coverage and stable 3DGS training under sparse views. MoDGS (Qingming et al., 2025) introduces a 3D-aware initialization for learning deformation fields and employs a robust depth loss to guide the learning of dynamic scene geometry.

A.2 Proof of VRT Theorem

The basic notation is defined in Section 3.2. Now, we aim to characterize the formulation $\frac{\mathrm{VAR}[g_\theta^{\text{new}}]}{\mathrm{VAR}[g_\theta^{\text{trad}}]}$ step by step. For clarity, we begin with a single image sample for derivation. Based on Equation (2), the traditional paradigm gradient $g_\theta^{\text{trad}} = \frac{\partial \mathcal{L}}{\partial \theta}$ is determined by $\frac{\partial \mathcal{L}}{\partial f} \cdot \frac{\partial f}{\partial \theta}$. Hence, if we define the expectation over the gradient for each image sample as $\mathbb{E}[\frac{\partial \mathcal{L}}{\partial \theta}]$, then the variance $\mathrm{VAR}[g_\theta^{\text{trad}}]$ can be represented by

$$\mathrm{VAR}[g_\theta^{\text{trad}}] = \mathrm{VAR}\left[\left(\frac{\partial \mathcal{L}}{\partial f} \cdot \frac{\partial f}{\partial \theta}\right)\right] \tag{25}$$

$$= \mathbb{E}\left[\left(\frac{\partial \mathcal{L}}{\partial f} \cdot \frac{\partial f}{\partial \theta}\right)^2\right] - \mathbb{E}\left[\left(\frac{\partial \mathcal{L}}{\partial f} \cdot \frac{\partial f}{\partial \theta}\right)\right]^2. \tag{26}$$

Under the Weak Dependence Assumption (Assumption 3), we have $|\mathrm{Corr}(\frac{\partial \mathcal{L}}{\partial f}, \frac{\partial f}{\partial \theta})| \to 0$, indicating that the two terms can be regarded as nearly independent, and thus

$$\mathbb{E}\left[\left(\frac{\partial \mathcal{L}}{\partial f} \cdot \frac{\partial f}{\partial \theta}\right)\right] \approx \mathbb{E}\left[\frac{\partial \mathcal{L}}{\partial f}\right] \cdot \mathbb{E}\left[\frac{\partial f}{\partial \theta}\right]. \tag{27}$$

Since the model weights θ are usually randomly initialized at the beginning of training, the positive and negative gradients of $\frac{\partial f}{\partial \theta}$ are approximately symmetric around 0. Therefore, we can believe $\mathbb{E}\left[\frac{\partial f}{\partial \theta}\right] \approx 0$, indicating that the model sensitivity is approaching zero at the initialization period. And thus, we get $\mathbb{E}\left[\left(\frac{\partial \mathcal{L}}{\partial f} \cdot \frac{\partial f}{\partial \theta}\right)\right]^2 \approx 0$, and

$$\mathrm{VAR}[g_\theta^{\text{trad}}] \approx \mathbb{E}\left[\left(\frac{\partial \mathcal{L}}{\partial f} \cdot \frac{\partial f}{\partial \theta}\right)^2\right] \tag{28}$$

$$= \mathbb{E}\left[\left(\frac{\partial \mathcal{L}}{\partial f}\right)^2 \cdot \left(\frac{\partial f}{\partial \theta}\right)^2\right] \tag{29}$$

$$\approx \mathbb{E}\left[\left(\frac{\partial \mathcal{L}}{\partial f}\right)^2\right] \cdot \mathbb{E}\left[\left(\frac{\partial f}{\partial \theta}\right)^2\right], \tag{30}$$

because $|\mathrm{Corr}(\frac{\partial \mathcal{L}}{\partial f}, \frac{\partial f}{\partial \theta})| \to 0$, we can believe that $\frac{\partial \mathcal{L}}{\partial f}$ and $\frac{\partial f}{\partial \theta}$ are almost independent, as well as the squares of them.

Following the same property, we can derive a similar formulation of $\mathrm{VAR}[g_\theta^{\text{new}}]$ such that

$$\mathrm{VAR}[g_\theta^{\text{new}}] \approx \mathbb{E}[\mathcal{I}(S)^2] \cdot \mathbb{E}\left[\left(\frac{\partial \mathcal{L}}{\partial f}\right)^2\right] \cdot \mathbb{E}\left[\left(\frac{\partial \alpha}{\partial \theta}\right)^2\right], \tag{31}$$

and thus

$$\frac{\mathrm{VAR}[g_\theta^{\mathrm{new}}]}{\mathrm{VAR}[g_\theta^{\mathrm{trad}}]} \approx \mathbb{E}[\mathcal{I}(S)^2] \cdot \frac{\mathbb{E}\left[\left(\frac{\partial \alpha}{\partial \theta}\right)^2\right]}{\mathbb{E}\left[\left(\frac{\partial f}{\partial \theta}\right)^2\right]}. \tag{32}$$

Finally, under the Bounded Multiplier Assumption (Assumption 1) and the Metric-Depth Variation Assumption (Assumption 2), we can derive an approximated upper boundary over the whole dataset samples as

$$\mathbb{E}\left[\frac{\mathrm{Var}[g_\theta^{\mathrm{new}}]}{\mathrm{Var}[g_\theta^{\mathrm{trad}}]}\right] \lesssim \mathbb{E}[\mathcal{I}(S)^2] \cdot \frac{\kappa^2}{\Lambda^2}. \tag{33}$$

A.3 Implementation details

Metric multiplier transformation. In practice, according to statistical counting (see the third sub-figure of Figure 2), we find that most of the α_{gt} values concentrate around 1. Moreover, given input resolution as 518×686 with only 86 anchor points (fairly sparse), α_{gt} mostly (within $3 \cdot \mathrm{std}(\alpha_{\mathrm{gt}})$) falls into the interval of $[0, 2]$, and thus we set the hyperparameter $\alpha_{\max} = 2$ and $\phi(\alpha_{\mathrm{res}}^\theta) = 2 \cdot \alpha_{\mathrm{res}}^\theta$ over most datasets to avoid multiple setting across different datasets.

Stride for sparse anchors. Following PromptDA, we introduce a sparse anchor interpolation method for synthetic datasets as mentioned above Equation (8). To align with PromptDA (Lin et al., 2025), we also downsample the GT depth map to low resolution (192 × 256) and sample points with a stride of 7 under the first configuration, which has about 1,000 anchors per image. Since PromptDA and Marigold-DC have not released their training or sampling code, we reproduce the sampling method of PromptDA, simulating the noise of real LiDAR data, and expand it to more sparse settings. We also conduct the experiments for more sparse anchors with higher resolution(518 × 686), with each being a stride of 16(about 1,000 anchors), a stride of 32(about 300 anchors) and a stride of 64(about 80 anchors).

Notably, the number/percentage of depth anchors has a direct and significant impact on the results. For DepthLab (Liu et al., 2024b), it selected combinations of strokes, circles, and squares with **0.5%** to **1%** of pixels in the depth map, while Marigold-DC sampled **150** to **1,500** points at a resolution of 640×480. Our GSD can outperform with even fewer anchors. The sparsity of our sampling method can be found in Table 6.

Table 6: Stride, points and pct relationship

stride	resolution	#points	percentage
7	192 × 256	1000	2.0%
16	480 × 640	1200	0.39%
32	480 × 640	300	0.098%
64	480 × 640	75	0.0024%

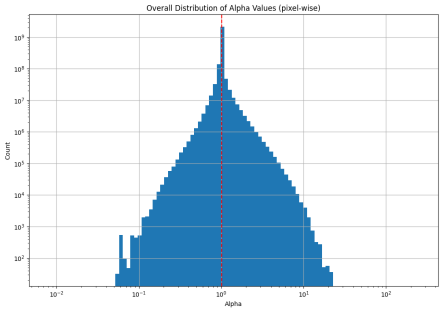

Figure 6: α distribution over pixels

Training configurations. Our model was trained on 8 NVIDIA V100 GPUs with 32GB of memory for 30 epochs, using a batch size of 2 and the AdamW optimizer with a learning rate of 2e-4. For Hypersim (Roberts et al., 2021) the image resolution is 518 × 686 downscaling to 192 × 256 and a LiDAR stride of 7 following PromptDA. For KITTI DC dataset we trained 10 epochs with the resolution of 378 × 1252. For VKITTI we trained 10 epochs with the resolution 364 × 1204 downscaling to 264 × 912 and a LiDAR stride of 7.

Evaluation protocol and further discussions. We compare our method to various baselines as shown in Table 1. All the methods have been trained on Hypersim, and we have fine-tuned them on KITTI for a fair comparison. The post-fusion method of Depth Anything v2 refers to scale and shift-based least squares alignment with relative depth prediction. We follow the official codebase to fine-tune Depth Anything V2 for metric depth estimation on their provided checkpoints for indoor and outdoor scenes, respectively, as DAv2 Metric. According to Prompting Depth Anything, their released checkpoint is pretrained on Hypersim and then the other two datasets. We reproduce the training process of PromptDA according to its paper, which achieves a much better result. Marigold-DC is a test-time training method that optimizes each dense depth map for several steps. As it is time-consuming, we only select one out of ten samples while taking the standard 50 inference steps for optimal performance.

Since most methods have not released their sampling sparse depth code for depth estimation datasets, it is a challenging task to align the anchor setting and re-evaluate all the metrics. We only conduct and re-evaluate four methods under our sampling and interpolation paradigm.

Model details. We inherit the pre-trained visual encoder DINOv2 from Depth Anything v2, using Vision Transformer Large, and take the last feature layer as a 1024-dimensional vector, denoted as F_{vit}. The CNN encoder has a downscale factor of 4, and the output F_{cnn} dimension is 128. The channels of our fusion UNet are 128 for ViT-Large. Our Gaussian head decodes the refined features into 37 channels in detail: 1 for opacity, 2 for the origin offset of the ray, 3 for sphere scale (variance), 4 for quaternions related to the orientation/covariance, and 27 for SH coefficients (SH degree is 2). The total number of parameters in our model is 308 million (3.8 million trainable) for ViT-L and 25 million (3.1 million trainable) for ViT-S. Our model architecture is highly flexible, allowing for the replacement of the ViT with state-of-the-art DINOv3 or SAM modules. The CNN encoder can be replaced by a pre-trained ResNet, and the ViT encoder can be unfrozen for fine-tuning. We can even introduce another branch of image feature from CLIP or SigLIP2 for semantic enhancement. Equipped with the prior injection of the image encoding module pretrained on large-scale real-world datasets, we expect to achieve better experimental results.

A.4 Dataset details

Training dataset. Our checkpoint is trained only on Hypersim (Roberts et al., 2021) for visualization and is compared with other methods in 1 and 2. It is worth noting that the Hypersim training set is an indoor synthetic dataset with only 59k training samples. We also train GSD from scratch on the KITTI Completion (Geiger et al., 2013) dataset—a real-world driving scene dataset with paired RGB images and sparse LiDAR depth for comparison, as shown in 1. Its semi-dense ground truth is derived from the temporal accumulation of consecutive LiDAR frames. For the outdoor zero-shot setting, we trained GSD on the Virtual KITTI dataset with about 21k samples.

Evaluation datasets In alignment with Prompting Depth Anything, we first make a comparison on the setting that downscales the ground-truth to the resolution of 192 $\times$ 256 with a stride of 7. Considering the potential application to high-resolution and authentic images, we also evaluated several stride settings on the Hypersim validation set. For outdoor scenes with irregular LiDAR anchors, we utilize the full validation split of 6,694 samples for the KITTI DC dataset.

We also evaluated our GSD model in a zero-shot manner on five unseen real-world datasets. The evaluation datasets encompass both indoor and outdoor scenarios, covering a diverse range of image resolutions, sparse depth densities, acquisition devices, and noise levels. For NYUv2 and ScanNet, we evaluate at a resolution of 640$\times$480. For KITTI, we use a resolution of 352$\times$1216, ETH3D uses 756$\times$1134, and DIODE uses 768×1024. All of the datasets above, except for KITTI DC, do not have LiDAR anchors; therefore, we sample the ground-truth with a stride of 16 for evaluation.

A.5 MORE VISUALIZATION RESULTS

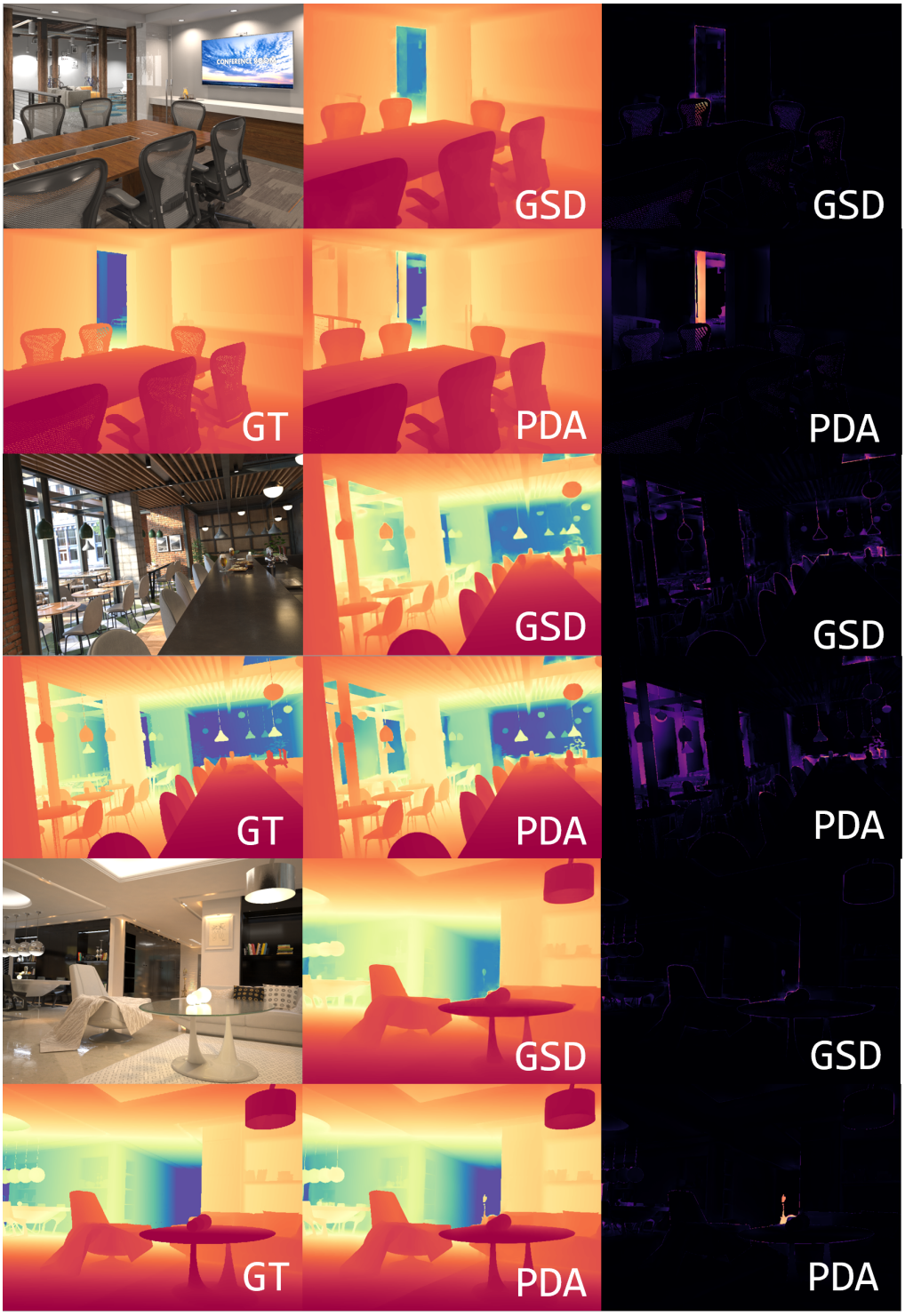

Figure 7: More visualization results compared with PromptDA. The first line covers RGB, our prediction and our errors; the second line covers GT, PromptDA prediction, PromptDA errors;stride=7, d_{init} 192×256. Due to the opacity attribution of 3DGS, our GSD can better recognize transparent objects as shown in the above two images than PromptDA.

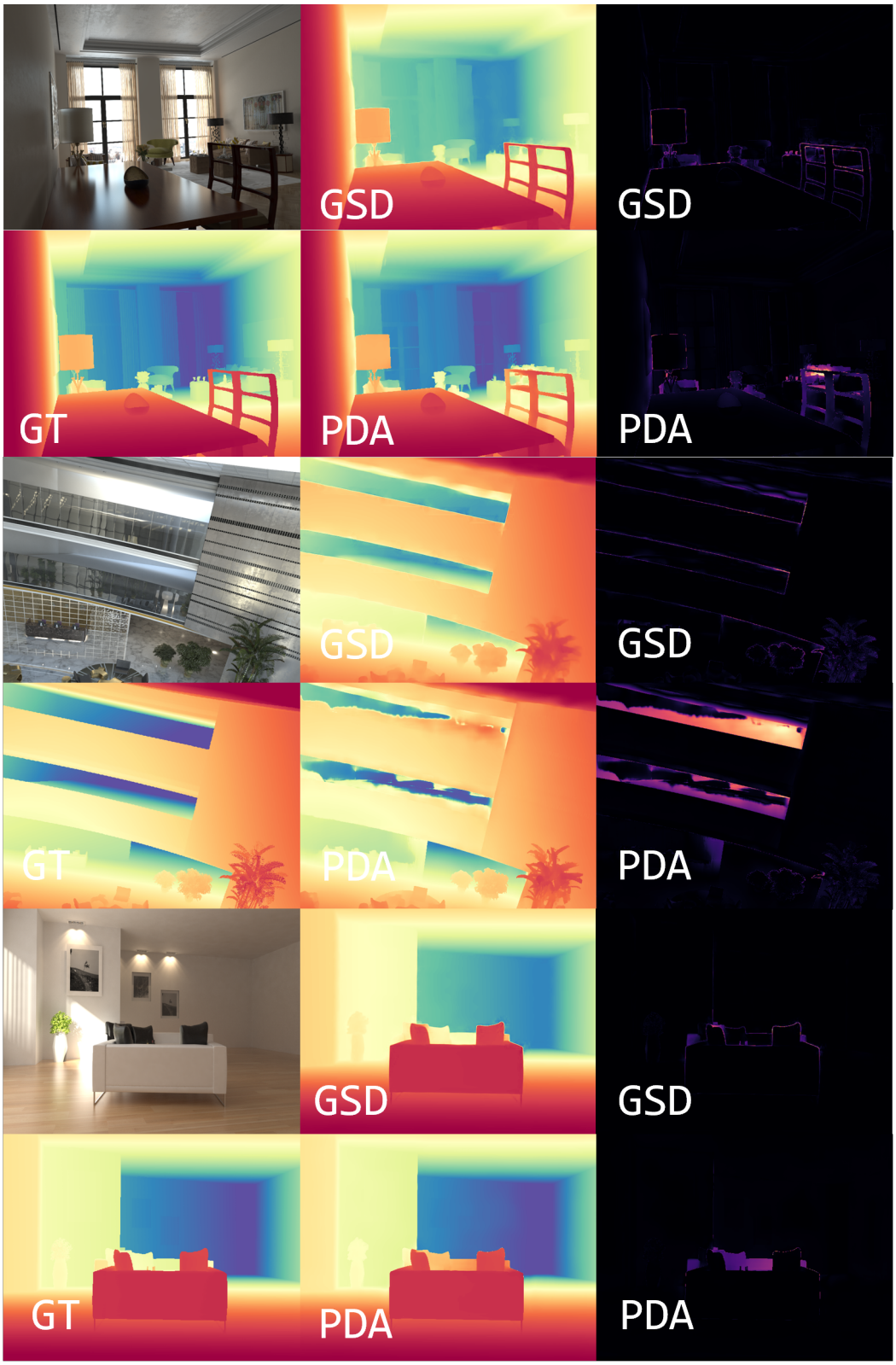
GSD
GSD
GT
PDA
PDA
GSD
GSD
GT
PDA
PDA
GSD
GSD
GT
PDA
PDA

Figure 8: More visualization results compared with PromptDA. The first line covers RGB, our prediction and our errors; the second line covers GT, PromptDA prediction, PromptDA errors;stride=7, d_{init} 192×256. Due to our strong regulation of depth anchors, GSD can follow the lidar prompt better, whereas PromptDA attempts to judge the distance by intuition such as the table behind the chair and the farther side of sofa.

A.6 LLM USAGE STATEMENT

We acknowledge the use of large language models (LLMs) to assist in polishing the writing and improving the grammatical fluency of this manuscript. The human authors performed all ideation, technical development, experimental analysis, and final editing.