Prompting Depth Anything for 4K Resolution Accurate Metric Depth Estimation

Haotong Lin[1,2*] Sida Peng[1] Jingxiao Chen[3] Songyou Peng[4] Jiaming Sun[1]
Minghuan Liu[3] Hujun Bao[1] Jiashi Feng[2] Xiaowei Zhou[1†] Bingyi Kang[2]
[1]Zhejiang University [2]ByteDance Seed [3]Shanghai Jiao Tong University [4]ETH Zurich

https://PromptDA.github.io/

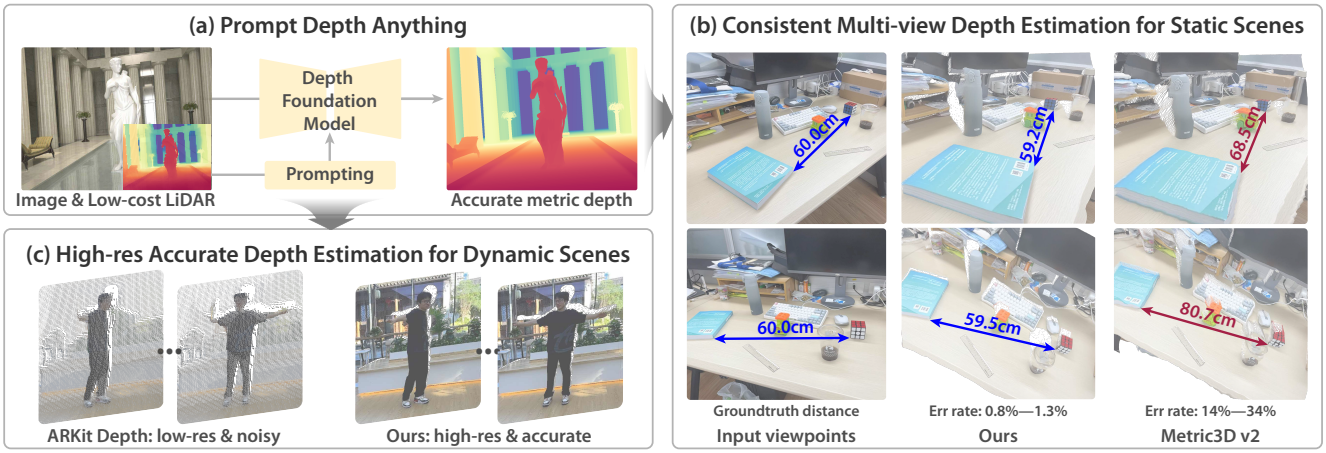

Figure 1. **Illustration and capabilities of *Prompt Depth Anything***. (a) *Prompt Depth Anything* is a new paradigm for metric depth estimation, which is formulated as prompting a depth foundation model with a metric prompt, specifically utilizing a low-cost LiDAR as the prompt. (b) Our method enables consistent depth estimation, addressing the limitations of Metric3D v2 [24] that suffer from inaccurate scale and inconsistency. (c) It achieves accurate 4K accurate depth estimation, significantly surpassing ARKit LiDAR Depth (240 × 320).

Abstract

Prompts play a critical role in unleashing the power of language and vision foundation models for specific tasks. For the first time, we introduce prompting into depth foundation models, creating a new paradigm for metric depth estimation termed ***Prompt Depth Anything****. Specifically, we use a low-cost LiDAR as the prompt to guide the Depth Anything model for accurate metric depth output, achieving up to 4K resolution. Our approach centers on a concise prompt fusion design that integrates the LiDAR at multiple scales within the depth decoder. To address training challenges posed by limited datasets containing both LiDAR depth and precise GT depth, we propose a scalable data pipeline that includes synthetic data LiDAR simulation and real data pseudo GT depth generation. Our approach sets new state-of-the-arts on the ARKitScenes and ScanNet++ datasets and benefits downstream applications, including 3D reconstruction and generalized robotic grasping.*

1. Introduction

High-quality depth perception is a fundamental challenge in computer vision and robotics. Recent monocular depth estimation has experienced a significant leap by scaling the model or data, leading to the flourishing of depth foundation models [18, 28, 67, 68]. These models demonstrate strong abilities in producing high-quality relative depth, but suffer from scale ambiguity, hindering their practical applications in autonomous driving and robotic manipulation, *etc*. Therefore, significant efforts have been made to achieve metric depth estimation, by either finetuning depth foundation models [6, 19] on metric datasets or training metric depth models with image intrinsics as additional inputs [8, 24, 44, 72]. However, neither of them can address the problem properly, as illustrated in Fig. 1(b).

A natural question thus arises: *Do these foundation models truly lack utility in accurate metric depth estimation*? This reminds us to closely examine the foundation models

*This work was done when Haotong Lin was in an internship at ByteDance Seed. †Corresponding author: Xiaowei Zhou.

in natural language [1, 9] and vision [37, 38, 50], which often involve pre-training and instruction tuning stages. A properly designed ***prompt*** and a ***instruction dataset*** can unlock the power of foundation models on downstream tasks. Inspired by these successes, we propose a new paradigm for metric depth estimation by treating it as a downstream task, *i.e.*, prompting a depth foundation model with metric information. We believe this prompt can take any form as long as the scale information is provided, *e.g.*, camera intrinsics. In this paper, we validate the feasibility of the paradigm by choosing low-cost LiDAR as the prompt for two reasons. First, it provides precise metric scale information. Second, it is widely available, even in common mobile devices (*e.g.*, Apple iPhone has a LiDAR).

Specifically, based on Depth Anything [68], we propose ***Prompt Depth Anything***, which achieves 4K resolution accurate metric depth estimation. At the core of our method is a concise prompt fusion architecture tailored for the DPT-based [46] depth foundation models [8, 68]. The prompt fusion architecture integrates the LiDAR depth at multiple scales within the DPT decoder, fusing the LiDAR features for depth decoding. The metric prompt provides precise spatial distance information, making the depth foundation model particularly serve as a local shape learner, resulting in accurate and high-resolution metric depth estimation.

Training *Prompt Depth Anything* requires both LiDAR depth and precise GT depth. However, existing synthetic data [49] lacks LiDAR depth, and real-world data [70] with LiDAR only has an imprecise GT depth of bad edges. To solve this challenge, we propose a scalable data pipeline that simulates low-resolution, noisy LiDAR for synthetic data and generates pseudo GT depth with high-quality edges for real data using a reconstruction method [2]. To mitigate errors in the pseudo GT depth from the 3D reconstruction, we introduce an edge-aware depth loss that leverages only the gradient of pseudo GT depth, which is prominent at edges. We experimentally demonstrate that these efforts result in highly accurate depth estimation.

We evaluate the proposed method on ARKitScenes [3] and ScanNet++ [70] datasets containing iPhone ARKit depth. It consistently exhibits state-of-the-art performance across datasets and metrics. Even our zero-shot model achieves better performance compared to other methods [6, 68] in non-zero-shot testing, highlighting the generalization ability of prompting a foundation model. We also show that the foundation model and prompt of *Prompt Depth Anything* can be replaced with DepthPro [8] and vehicle LiDAR [53], respectively. Furthermore, we demonstrate that it benefits several downstream applications, including 3D reconstruction and generalized robotic object grasping.

In summary, this work has the following contributions:

- *Prompt Depth Anything*, a new paradigm for metric depth estimation by prompting a depth foundation model with a low-cost LiDAR as the metric prompt.
- A concise prompt fusion architecture for depth foundation models, a scalable data pipeline, and an edge-aware depth loss to train *Prompt Depth Anything*.
- State-of-the-art performance on depth estimation benchmarks [3, 70], showing the extensibility of replacing depth foundation models and LiDAR sensors, and highlighting benefits for several downstream applications including 3D reconstruction and robotic object grasping.

2. Related Work

Monocular depth estimation. Traditional methods [23, 52] rely on hand-crafted features for depth estimation. With the advent of deep learning, this field has seen significant advancements. Early learning-based approaches [14, 15] are often limited to a single dataset, lacking generalization capabilities. To enhance generalization, diverse datasets [12, 32, 59–61, 63, 64, 69], affine-invariant loss [45], and more powerful network architectures [46] have been introduced. More recently, latent diffusion models [50], pre-trained on extensive image generation tasks, have been applied to depth estimation [20, 28]. These models exhibit good generalization, estimating relative depth effectively, though they remain scale-agnostic. To achieve metric depth estimation, early methods either model the problem as global distribution classification [4, 5, 17, 35] or fine-tune a depth model on metric depth datasets [6, 33, 34]. Recent methods [19, 24, 44, 71, 72] discuss the ambiguity in monocular metric depth estimation and address it by incorporating camera intrinsic parameters. Although recent methods [8, 20, 24, 28, 44, 68, 72] exhibit strong generalization ability and claim to be depth foundation models [8, 18, 24, 68], metric depth estimation remains a challenge as shown in Fig. 1(b). We seek to address this challenge by prompting the depth foundation models with a metric prompt, inspired by the success of prompting in vision and vision-language models [37, 38, 75].

Depth estimation with auxiliary sensors. Obtaining dense depth information through active sensors typically demands high power consumption. A more practical approach involves utilizing a low-power active sensor to capture sparse depth, which can then be completed into dense maps. Many studies investigate methods to fill in sparse depth data. Early works rely on filter-based [21, 26, 30] and optimization-based [16, 66] techniques for depth completion. More recent studies [10, 11, 13, 36, 39, 55–57, 65, 76] adopt learning-based approaches for depth completion. Typically, these methods are not tested on real indoor LiDAR data but rather on simulated sparse lidar for depth datasets such as NYUv2 [15] to reconstruct complete depth. This is because real testing setups require both low-power and high-power LiDAR sensors. More recent

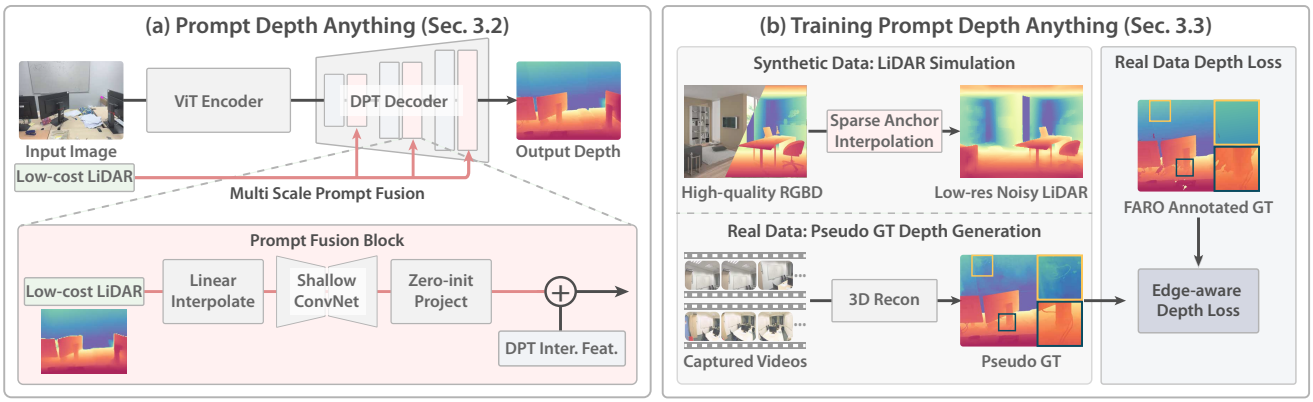

Figure 2. **Overview of *Prompt Depth Anything*.** (a) *Prompt Depth Anything* builds on a depth foundation model [68] with a ViT encoder and a DPT decoder, and adds a multi-scale prompt fusion design, using a prompt fusion block to fuse the metric information at each scale. (b) Since training requires both low-cost LiDAR and precise GT depth, we propose a scalable data pipeline that simulates LiDAR depth for synthetic data with precise GT depth, and generates pseudo GT depth for real data with LiDAR. An edge-aware depth loss is proposed to merge accurate edges from pseudo GT depth with accurate depth in textureless areas from FARO annotated GT depth on real data.

works have collected both low-power and high-power LiDAR data. To collect such data, DELTA [31] builds a suite to collect data using L5 and Intel RealSense 435i, while three other datasets [3, 48, 70] are collected using iPhone LiDAR and FARO LiDAR. We focus on the latter, as iPhone is widely available. A recent work similar to ours is Depth Prompting [41]. Our approach differs in that we use a network to take sparse depth as a prompt for the depth foundation model, achieving specific output. In contrast, they fuse sparse depth with features from the depth foundation model to post-process the foundation model output, which does not constitute prompting a foundation model.

3. Method

Monocular depth estimation models [8, 67, 68] are becoming depth foundation models for their generalization ability obtained from large-scale data. However, due to the inherent ambiguities, they cannot achieve high accuracy on metric depth estimation as shown in Fig. 1(b). Inspired by the success of prompting for vision [29, 38, 50] and language [1] foundation models, we propose *Prompt Depth Anything* prompting the depth foundation model with a metric prompt to achieve metric depth estimation. We take the low-cost LiDAR as the metric prompt in this work, as it has recently been integrated into lots of smartphones, making this setup highly practical. To be specific, we aim to prompt the depth foundation model to unleash its power for accurate metric depth estimation.

3.1. Preliminary: Depth Foundation Model

Current depth foundation models [7, 67, 68, 72] generally share similar network structures of DPT [46] networks. Specifically, given an image $\mathbf{I} \in \mathbb{R}^{C\times H\times W}$, they take a vision transformer (ViT) with multiple stages to extract tokenized image features $\{\mathbf{T}_i\}$, where $\mathbf{T}_i \in \mathbb{R}^{C_i\times(\frac{H}{p}\times\frac{W}{p}+1)}$ represents the feature map at stage S_i, D_i is the feature dimension at stage S_i, and p is the patch size. The DPT decoder reassembles features from different stages into image-like representations $\mathbf{F}_i \in \mathbb{R}^{D_i\times\frac{H}{p}\times\frac{W}{p}}$ with the reassemble operation [46]. Finally, a sequence of convolutional blending steps are applied to merge features $\mathbf{F}_i$ across different stages, predicting a dense depth map $\mathbf{D} \in \mathbb{R}^{H\times W}$.

We note that there exists another line of depth foundation models [18, 20, 28] that use the image diffusion model [51] to estimate depth maps. Due to the high computational cost of diffusion models, we only consider DPT-based depth foundation models [8, 68] as our base model for real-time performance in this work.

3.2. Prompt Depth Anything

In this section, we seek to find a concise way to incorporate a low-cost LiDAR (i.e., a low-resolution and noisy depth map) as a prompt into the depth foundation model. To this end, we propose a concise prompt fusion architecture tailored for the DPT-based [46] depth foundation models to integrate low-resolution depth information. As shown in Fig. 2(a), the prompt fusion architecture integrates low-resolution depth information at multiple scales within the DPT Decoder. Specifically, for each scale S_i in the DPT Decoder, a low-resolution depth map $\mathbf{L} \in \mathbb{R}^{1\times H_\mathbf{L}\times W_\mathbf{L}}$ is firstly bilinearly resized to match the spatial dimensions of the current scale $\mathbb{R}^{1\times H_i\times W_i}$. Then, the resized depth map is passed through a shallow convolutional network to extract depth features. After that, the extracted features are projected to the same dimension as the image features $\mathbf{F}_i \in \mathbb{R}^{C_i\times H_i\times W_i}$ using a zero-initialized convolutional layer. Finally, the depth features are added to the DPT intermediate features for depth decoding. The illustration of

Figure 3. **Effects on the synthetic data lidar simulation and real data pseudo GT generation with the edge-aware depth loss.** The middle and right columns are the depth prediction results of our different models. The two rows highlight the significance of sparse anchor interpolation for lidar simulation and pseudo GT generation with edge-aware depth loss, respectively.

this block design is shown in Fig. 2.

The proposed design has the following advantages. Firstly, it introduces only 5.7% additional computational overhead (1.789 TFLOPs v.s. 1.691 TFLOPs for a 756×1008 image) to the original depth foundation model, and effectively addresses the ambiguity issue inherent in the depth foundation model as demonstrated in Tab. 3(b). Secondly, it fully inherits the capabilities of the depth foundation model because its encoder and decoder are initialized from the foundation model [68], and the proposed fusion architecture is zero-initialized, ensuring that the initial output is identical to that of the foundation model. We experimentally verify the importance of inheriting from a pretrained depth foundation model as shown in Tab. 3(c).

Optional designs. Inspired by conditional image generation methods [27, 42, 75], we also explore various potential prompt conditioning designs into the depth foundation model. Specifically, we experimented with the following designs: a) Adaptive LayerNorm [27, 43] which adapts the layer normalization parameters of the encoder blocks based on the conditioning input, b) CrossAttention [58] which injects a cross attention block after each self-attention block and integrates the conditioning input through cross-attention mechanisms, and c) ControlNet [75] which copies the encoder blocks and inputs control signals to the copied blocks to control the output depth. As shown in Tab. 3(d,e,f), our experiments reveal that these designs do not perform as well as the proposed fusion block. A plausible reason is that they are designed to integrate cross-modal information (e.g., text prompts), which does not effectively utilize the pixel alignment characteristics between the input low-res LiDAR and the output depth. We detail these optional designs in the supp.

3.3. Training Prompt Depth Anything

Training *Prompt Depth Anything* simultaneously requires a low-cost LiDAR and precise GT depth. However, synthetic data [49] do not contain LiDAR depth, real-world data with noisy LiDAR depth [70] only have imprecise depth annotations. Therefore, we propose a LiDAR simulation method for synthetic data and generate pseudo GT depth from Zip-NeRF [2] with an edge-aware depth loss for real data.

Synthetic data: LiDAR simulation. A LiDAR depth map is low-resolution and noisy. The naive approach for simulating it is to directly downsample the synthetic data depth map. However, this method leads to the model learning depth super-resolution, as shown in Fig. 3, meaning that the model does not correct the LiDAR noise. To simulate the noise, we introduce a sparse anchor interpolation method. Specifically, we first downsample the GT depth map to low-resolution (192×256, exactly the depth resolution of iPhone ARKit Depth). Then we sample points on this depth map using a distorted grid with a stride (7 in practice). The remaining depth values are interpolated from these points using RGB similarity with KNN. As shown in Fig. 3, it effectively simulates LiDAR noise and results in better depth prediction. We provide visualization results of the simulated LiDAR in the supp.

Real Data: Pseudo GT depth generation. We also add real data [70] to our training data. The annotated depth in ScanNet++ [70] is re-rendered from a mesh scanned by a high-power LiDAR sensor (FARO Focus Premium laser scanner). Due to the presence of many occlusions in the scene, several scan positions (typically 4 in a medium-sized scene in ScanNet++) result in an incomplete scanned mesh, leading to depth maps with numerous holes and poor edge quality, as illustrated in Fig. 2(b). Motivated by the success of reconstruction methods [2, 40], we propose using Zip-NeRF [2] to recover high-quality depth maps. Specifically, we train Zip-NeRF for each scene in ScanNet++ and re-rendered pseudo GT depth. To provide Zip-NeRF with high-quality and dense observations, we detect unblurred frames in Scannet++iPhone videos, and additionally utilize DSLR videos to provide high-quality dense-view images.

Real Data: Edge-aware depth loss. Although Zip-NeRF can generate high-quality edge depth, reconstructing textureless and reflective regions remains challenging as shown in Fig. 2(b). In contrast, these areas (e.g., walls, floors, and ceilings etc.) are usually planar with few occlusions, and the annotations depth in FARO rendered depth is good in these regions. This motivates us to leverage the strengths of both. We propose an edge-aware depth loss to meet these requirements. Specifically, we use the FARO scanned mesh depth and the gradient of the pseudo GT depth to supervise output

depth and the gradient of the output depth, respectively:

$$\mathcal{L}_{\text{edge}} = L_1(\mathbf{D}_{\text{gt}}, \hat{\mathbf{D}}) + \lambda \cdot \mathcal{L}_{\text{grad}}(\mathbf{D}_{\text{pseudo}}, \hat{\mathbf{D}}), \quad (1)$$

$$\mathcal{L}_{\text{grad}}(\mathbf{D}_{\text{pseudo}}, \hat{\mathbf{D}}) = (|\frac{\partial(\hat{\mathbf{D}}-\mathbf{D}_{\text{pseudo}})}{\partial x}| + |\frac{\partial(\hat{\mathbf{D}}-\mathbf{D}_{\text{pseudo}})}{\partial y}|). \quad (2)$$

In practice, we set $\lambda = 0.5$. The depth gradient is mainly prominent at the edges, which is exactly where the pseudo GT depth excels. The gradient loss encourages the model to learn the accurate edges from the pseudo GT depth, while the L1 loss encourages the model to learn the overall depth, ultimately leading to excellent depth prediction. We experimentally verify the effectiveness of the edge-aware depth loss in Tab. 3(j) and Fig. 3.

3.4. Implementation Details

In this section, we provide essential information about the network design, depth normalization, and training details. Please refer to the supp. for more details.

Network details. We utilize the ViT-large model as our backbone model. The shallow convolutional network comprises two convolutional layers with a kernel size of 3 and a stride of 1. More details can be found in the supp. Detailed running time analysis can be found in Sec. 4.3.

Depth normalization. The irregular range of input depth data can hinder network convergence. To address this, we normalize the LiDAR data using linear scaling to the range [0, 1], based on its minimum and maximum values. The network output is also normalized with the same scaling factor from LiDAR data, ensuring consistent scales and facilitating easier convergence during training.

Training details. We initiate training from the metric model released by Depth Anything v2 [68], incorporating a 10K step warm-up phase. During this warm-up phase, we fine-tune this metric model to output a normalized depth derived from the linear scaling of LiDAR data. Subsequently, we train our model for 200K steps. During the training process, the batch size is set to 2, utilizing 8 GPUs. We employ the AdamW optimizer, with a learning rate of 5e-6 for the ViT backbone and 5e-5 for the other parameters.

4. Experiments

4.1. Experimental Setup

We mainly conduct experiments on the HyperSim synthetic dataset [49] and two real-world datasets: ScanNet++ [70] and ARKitScenes [3], which provide iPhone RGB-LiDAR data (192 × 256 resolution) and annotated depth from a high-power LiDAR (1440 × 1920 resolution). We follow the suggested training and evaluation protocol in [3] for ARKitScenes, where 40K images are used for training and 5K images for evaluation. For the ScanNet++ dataset, we randomly select 20 scenes from its 50 validation scenes, amounting to approximately 5K images for our validation and the training set are from its 230 training scenes, containing about 60K images. To ensure a fair comparison, we additionally train a model with HyperSim training set to achieve zero-shot testing on ScanNet++ and ARKitScenes datasets. Besides depth accuracy metrics, we also report the TSDF reconstruction results of our method on ScanNet++, which reflects the depth consistency. We describe the details of the evaluation metrics in the supp.

Zero Shot	Net. / Post. / w/o LiDAR	384 × 512		768x1024		1440x1920	
		L1 ↓	RMSE ↓	L1 ↓	RMSE ↓	L1 ↓	RMSE ↓
No	**Ours**	**0.0135**	**0.0326**	**0.0132**	**0.0315**	**0.0138**	**0.0316**
	MSPF	0.0153	0.0369	0.0149	0.0362	0.0152	0.0363
	Depth Pro*	0.0437	0.0672	0.0435	0.0665	0.0425	0.0654
	DepthAny. v2*	0.0464	0.0715	0.0423	0.0660	0.0497	0.0764
	ZoeDepth*	0.0831	0.2873	0.0679	0.1421	0.0529	0.0793
	Depth Pro*	0.1222	0.1424	0.1225	0.1427	0.1244	0.1444
	DepthAny. v2*	0.0978	0.1180	0.0771	0.0647	0.0906	0.1125
	ZoeDepth*	0.2101	0.2784	0.1780	0.2319	0.1566	0.1788
Yes	**Ours$_{\text{syn}}$**	**0.0161**	**0.0376**	**0.0163**	**0.0371**	**0.0170**	**0.0376**
	D.P.	0.0251	0.0422	0.0253	0.0422	0.0249	0.0422
	BPNet	0.1494	0.2106	0.1493	0.2107	0.1491	0.2100
	ARKit Depth	0.0251	0.0424	0.0250	0.0423	0.0254	0.0426
	DepthAny. v2	0.0716	0.1686	0.0616	0.1368	0.0494	0.0764
	DepthAny. v1	0.0733	0.1757	0.0653	0.1530	0.0527	0.0859
	Metric3D v2	0.0626	0.2104	0.0524	0.1721	0.0402	0.1045
	ZoeDepth	0.1007	0.1917	0.0890	0.1627	0.0762	0.1135
	Lotus	0.0853	0.1793	0.1038	0.1782	0.1941	0.2741
	Marigold	0.0908	0.1849	0.0807	0.1565	0.0692	0.1065
	Metric3D v2	0.1777	0.2766	0.1663	0.2491	0.1615	0.2131
	ZoeDepth	0.6158	0.9577	0.5688	0.6129	0.5316	0.5605

Table 1. **Quantitative comparisons on ARKitScenes dataset.** The terms Net., Post. and w/o LiDAR refer to the LiDAR depth usage of models, where "Net." denotes network fusion, "Post." indicates post-alignment using RANSAC, and "w/o LiDAR" means the output is metric depth. Methods marked with * are finetuned with their released models and code on ARKitScenes [3] and ScanNet++ [70] datasets.

4.2. Comparisons with the State of the Art

We compare our method against the current SOTA depth estimation methods from two classes: *Monocular depth estimation (MDE)* and *depth completion/upsampling*. For MDE methods, we compare our method with Metric3D v2 [24], ZoeDepth [6], DepthPro [8], Depth Anything v1 and v2 [67, 68] (short for DepthAny. v1 and v2), Marigold [28] and Lotus [20]. For depth completion/upsampling methods, we compare our method with BPNet [57], Depth Prompting [41] (short for D.P.), MSPF [62]. To make a fair comparison with MDE methods, we align their predictions with ARKit LiDAR depth using the RANSAC align method. According to whether they have seen the testing data types during training, we divide methods into two categories: *zero-shot* and *non zero-shot*. We train a model Ours$_{\text{syn}}$ only with HyperSim training set to make comparisons with the zero-shot methods. As shown in Tabs. 1 and 2 and Figs. 4 and 5, our method consistently outperforms the existing methods. Note that

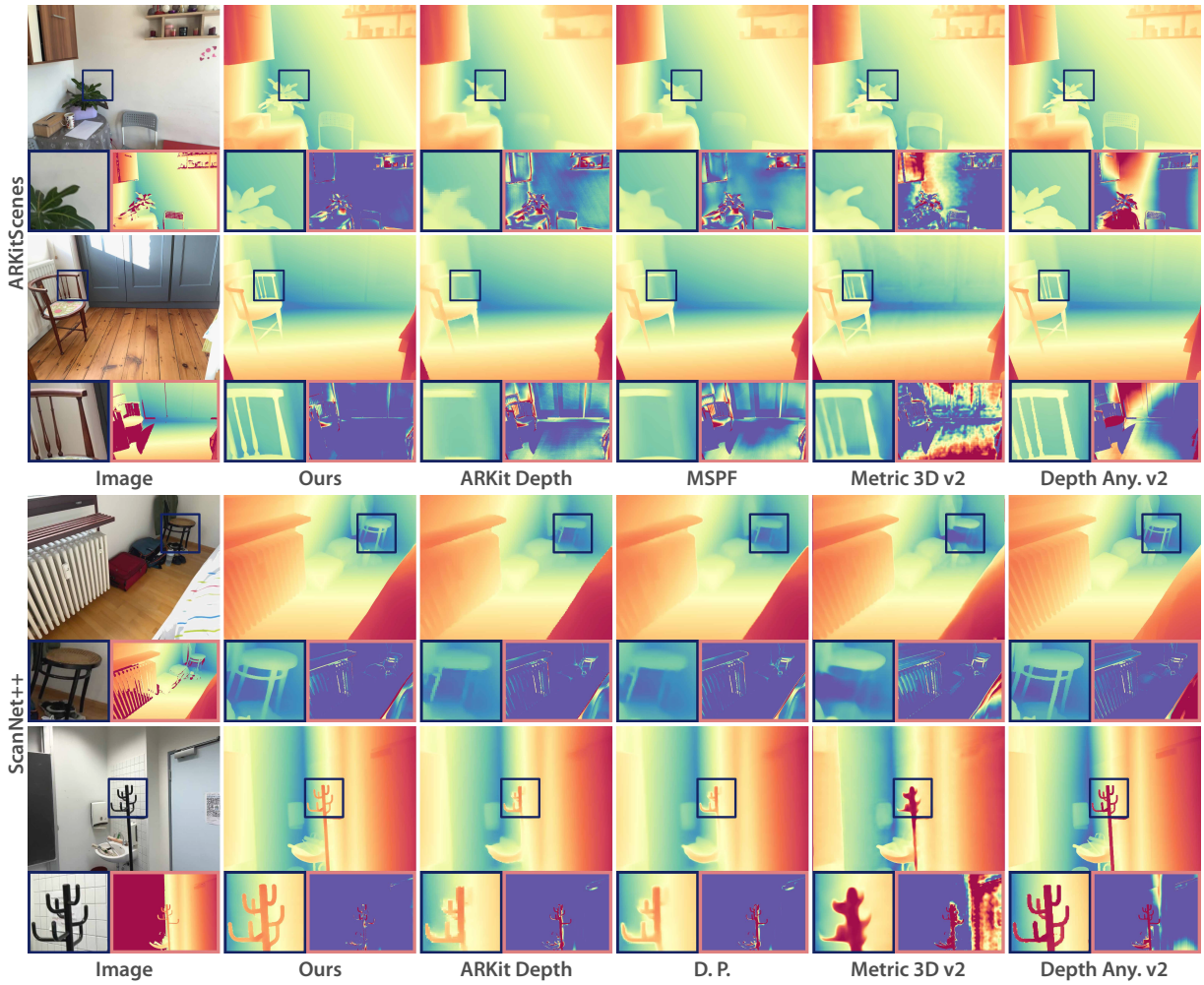

Figure 4. **Qualitative comparisons with the state-of-the-art.** "Metric3D v2" and "Depth Any. v2" are scale-shift corrected with ARKit depth. The pink boxes denote the GT depth and depth percentage error map, where red represents high error, and blue indicates low error.

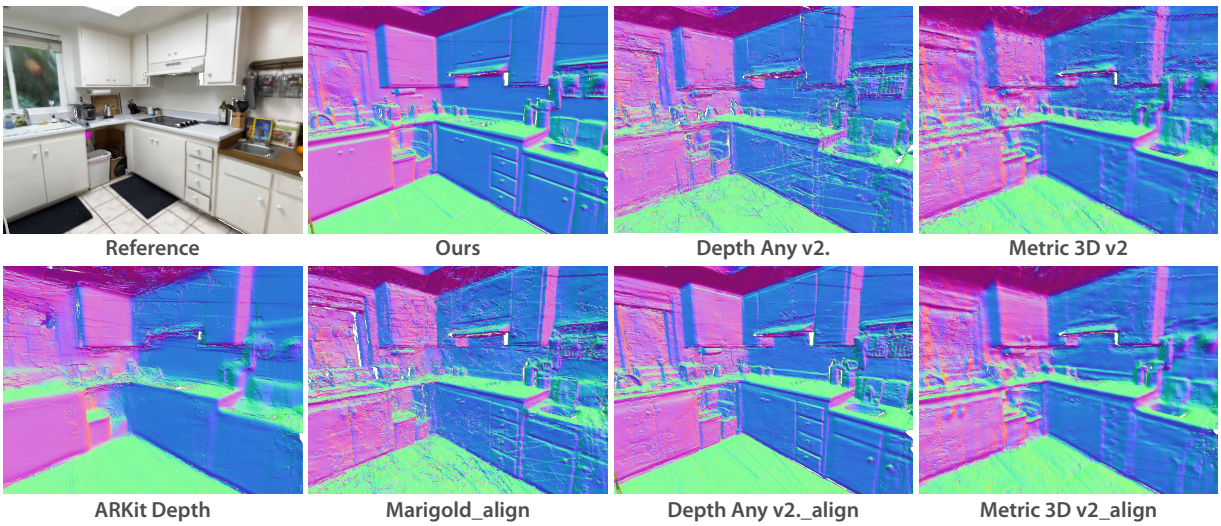

Figure 5. **Qualitative comparisons of TSDF reconstruction.** *_align denotes the scale-shift corrected depth with ARKit depth.

Zero Shot	Net. / Post. / w/o LiDAR	Depth Estimation				TSDF Reconstruction				
		L1 ↓	RMSE ↓	AbsRel ↓	$\delta_{0.5}$ ↑	Acc ↓	Comp ↓	Prec ↑	Recall ↑	F-score ↑
No	**Ours**	**0.0250**	**0.0829**	**0.0175**	**0.9781**	**0.0699**	**0.0616**	**0.7255**	**0.8187**	**0.7619**
	MSPF*	0.0326	0.0975	0.0226	0.9674	0.0772	0.0695	0.6738	0.7761	0.7133
	DepthAny. v2*	0.0510	0.1010	0.0371	0.9437	0.0808	0.0735	0.6275	0.7107	0.6595
	ZoeDepth*	0.0582	0.1069	0.0416	0.9325	0.0881	0.0801	0.5721	0.6640	0.6083
	DepthAny. v2*	0.0903	0.1347	0.0624	0.8657	0.1264	0.0917	0.4256	0.5954	0.4882
	ZoeDepth*	0.1675	0.1984	0.1278	0.5807	0.1567	0.1553	0.2164	0.2553	0.2323
Yes	**Ours$_{syn}$**	**0.0327**	**0.0966**	**0.0224**	**0.9700**	**0.0746**	**0.0666**	**0.6903**	**0.7931**	**0.7307**
	D.P.	0.0353	0.0983	0.0242	0.9657	0.0820	0.0747	0.6431	0.7234	0.6734
	ARKit Depth	0.0351	0.0987	0.0241	0.9659	0.0811	0.0743	0.6484	0.7280	0.6785
	DepthAny. v2	0.0592	0.1145	0.0402	0.9404	0.0881	0.0747	0.5562	0.6946	0.6127
	Depth Pro	0.0638	0.1212	0.0510	0.9212	0.0904	0.0760	0.5695	0.6916	0.6187
	Metric3D v2	0.0585	0.3087	0.0419	0.9529	0.0785	0.0752	0.6216	0.6994	0.6515
	Marigold	0.0828	0.1412	0.0603	0.8718	0.0999	0.0781	0.5128	0.6694	0.5740
	DepthPro	0.2406	0.2836	0.2015	0.5216	0.1537	0.1467	0.2684	0.3752	0.3086
	Metric3D v2	0.1226	0.3403	0.0841	0.8009	0.0881	0.0801	0.5721	0.6640	0.6083

Table 2. **Quantitative comparisons on ScanNet++ dataset.** The terms Net., Post. and w/o LiDAR refer to the LiDAR depth usage of models as the last table. Methods marked with * are finetuned with their released code on ARKitScenes [3] and ScanNet++ [70] datasets.

	ARKitScenes		ScanNet++		
	L1 ↓	AbsRel ↓	Acc ↓	Comp ↓	F-Score ↑
(a) Ours$_{syn}$ (synthetic data)	0.0163	0.0142	0.0746	0.0666	0.7307
(b) w/o prompting	0.0605	0.0505	0.0923	0.0801	0.5696
(c) w/o foundation model	0.0194	0.0169	0.0774	0.0713	0.7077
(d) AdaLN prompting	0.0197	0.0165	0.0795	0.0725	0.6943
(e) Cross-atten. prompting	0.0523	0.0443	0.0932	0.0819	0.5595
(f) Controlnet prompting	0.0239	0.0206	0.0785	0.0726	0.6899
(g) a + ARKitScenes data	0.0134	0.0115	0.0744	0.0662	0.7341
(h) g + ScanNet++ anno. GT	0.0132	0.0114	0.0670	0.0614	0.7647
(i) g + ScanNet++ pseudo GT	0.0139	0.0121	0.0835	0.0766	0.6505
(j) **Ours** (h,i+edge loss)	0.0132	0.0115	0.0699	0.0616	0.7619

Table 3. **Quantitative ablations on ARKitScenes and ScanNet++ datasets.** Please refer to Sec. 4.3 for detailed descriptions.

Ours$_{syn}$ achieves better performance than all non-zero-shot models [62, 68] on ScanNet++, highlighting the generalization ability of prompting a depth foundation model.

4.3. Ablations and Analysis

Prompting a depth foundation model. We assess its importance with two experiments: 1) Removing the prompting. Tab. 3(b) shows a significant performance drop. 2) Removing the foundation model initialization [68]. Tab. 3(c) shows a noticeable performance decline.

Prompting architecture design. We study different designs: AdaLN, Cross-attention, and ControlNet as discussed in Sec. 3.2. Tab. 3(d,e,f) reveals that ControlNet performs best but still falls short of our method.

Training data and edge-aware depth loss. We initially incorporate ARKitScenes data, which only enhances performance on ARKitScenes (Tab. 3(g)). Then we add ScanNet++, which improves results on both ARKitScenes and ScanNet++ (Tab. 3(h)). However, the depth visualization remains less than ideal (Fig. 3). Tab. 3(i) show that direct supervision with pseudo GT depth from reconstruction methods decreases performance. Ultimately, employing the edge-aware depth loss that utilizes pseudo GT depth and FARO annotated GT achieves comparable performance with Tab. 3(h) but with superior thin structure depth performance as shown in Fig. 3. We provide more qualitative ablation results in the supp.

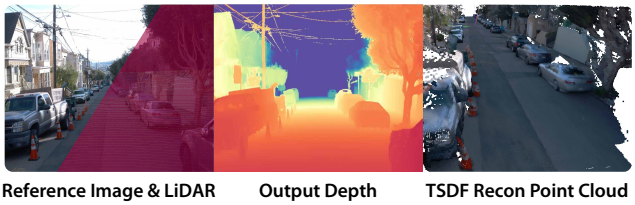

Figure 6. **Outdoor reconstruction by taking the vehicle LiDAR as metric prompt.** Please refer to the supp. for more video results.

Running time analysis. Our model with ViT-L runs at 20.4 FPS for an image resolution of 768×1024 on a A100 GPU. As ARKit6 supports 4K image recording, we test our model at a resolution of 2160×3840 and achieve 2.0 FPS. Note that our model can also be implemented with ViT-S, where the corresponding speeds are 80.0 and 10.3 FPS. More testing results can be found in the supp.

4.4. Zero-shot Testing on Diverse Scenes

Although our model is trained on indoor scenes, it generalizes well to various scenarios, including new rooms, gyms with thin structures, poorly lit museums, human and outdoor environments, as shown in Fig. 7, highlighting the effectiveness of prompting a depth foundation model. Please refer to the supp. for video results.

4.5. Application: 3D Reconstruction

Our consistent and scale-accurate depth estimation benefits the indoor 3D reconstruction as shown in Tab. 2 and Fig. 5. Besides, the prompt of our model can be easily replaced

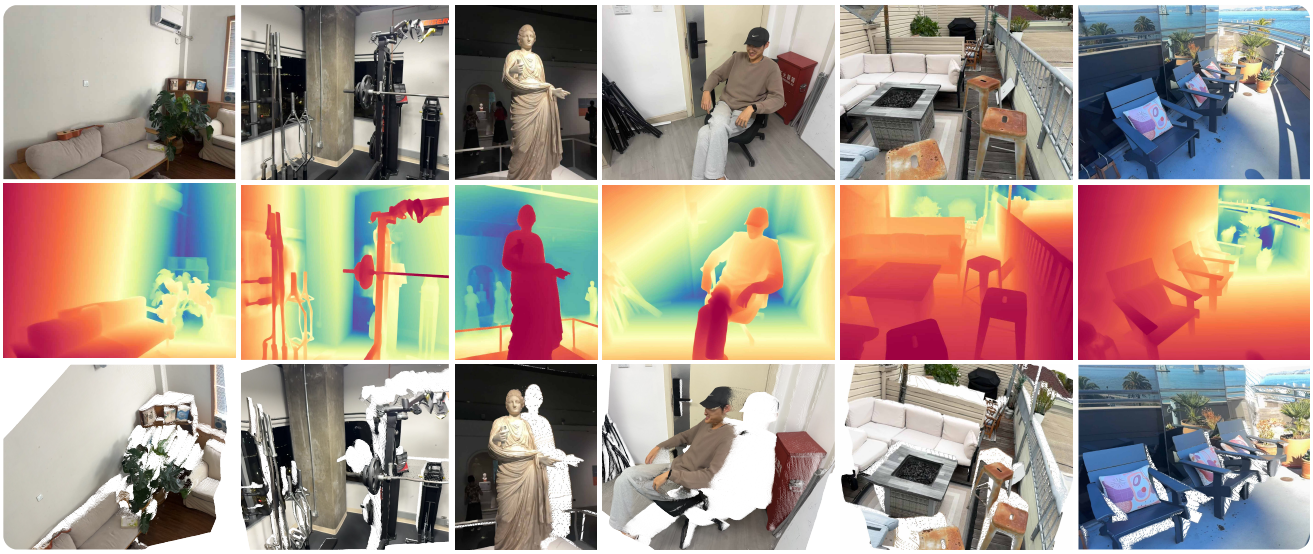

Figure 7. **Zero-shot testing on diverse scenes.**

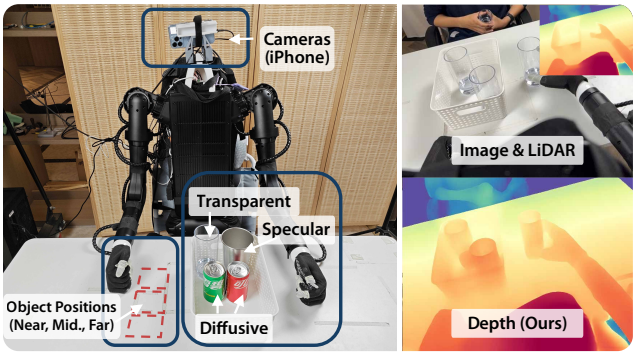

Figure 8. **Robotic grasping setup and input signal types.** Our goal is to grasp objects of various types using image/LiDAR/depth inputs. Red rectangles indicate potential object positions.

with vehicle LiDAR, which enables our model to achieve large-scale outdoor scene reconstruction as shown in Fig. 6. We detail the setup and include more video results for dynamic streets in the supp.

4.6. Application: Generalized Robotic Grasping

We set up a robotic platform to test our model in generalized robotic manipulation (Fig. 8), which typically requires depth or RGB as observations. Good depth estimation enhances the generalization ability because it accurately describes the 3D information of surroundings [25, 74]. Specifically, we train an ACT policy [77] to grasp various objects into the box, using different types of input signals such as RGB, LiDAR, and depth data from our model. We empirically find that our model generalizes well to unseen objects like transparent and specular objects when trained on diffusive objects, outperforming RGB and LiDAR inputs as shown in Tab. 4. This is because RGB is dominated by color, which leads to poor generalization across objects, and

Input Signal	Diffusive		Transparent	Specular
	Red Can	Green Can		
Ours	**1.0/1.0/1.0**	**1.0/1.0/1.0**	**0.3/1.0/1.0**	**0.8/1.0/0.9**
LiDAR	**1.0/1.0/1.0**	1.0/1.0/0.2	0.5/0.4/0.0	0.7/1.0/0.0
RGB	1.0/1.0/0.0	1.0/1.0/0.0	0.2/1.0/0.0	0.0/0.9/0.9

Table 4. **Grasping success rate on various objects**. Three numbers indicate objects placed at near, middle, and far positions. The grasping policy is trained on diffusive and tested on all objects.

the iPhone LiDAR depth is noisy and lacks the capability to perceive transparent objects. Please refer to the supp. for detailed setup descriptions and videos.

5. Conclusion and Discussions

This paper introduced a new paradigm for metric depth estimation, formulated as prompting a depth foundation model with metric information. We validated the feasibility of the paradigm by choosing the low-cost LiDAR depth as the prompt. A scalable data pipeline was proposed to generate synthetic LiDAR depth and pseudo GT depth for training. Extensive experiments demonstrate the superiority of our method against existing monocular depth estimation and depth completion/upsampling methods. Furthermore, we showed that it benefits for downstream tasks including 3D reconstruction and generalized robotic grasping.

Limitations and future work. This work has some known limitations. For instance, when using the iPhone LiDAR as the prompt, it cannot handle long-range depth, as the iPhone LiDAR detects very noisy depth for far objects. Additionally, we observed some temporal flickering of LiDAR depth, leading to a flickering depth prediction. These issues can be addressed in future works by considering more advanced prompt learning techniques that can extend the effective range and temporal prompt tuning.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv*, 2023. 2, 3

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *ICCV*, pages 19697–19705, 2023. 2, 4

[3] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, et al. Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgb-d data. *NeurIPS*, 2021. 2, 3, 5, 7, 12, 14, 15

[4] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. AdaBins: Depth estimation using adaptive bins. In *CVPR*, 2021. 2

[5] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. LocalBins: Improving depth estimation by learning local distributions. In *ECCV*, 2022. 2

[6] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth. *arXiv*, 2023. 1, 2, 5, 15

[7] Reiner Birkl, Diana Wofk, and Matthias Müller. Midas v3. 1–a model zoo for robust monocular relative depth estimation. *arXiv*, 2023. 3

[8] Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R Richter, and Vladlen Koltun. Depth pro: Sharp monocular metric depth in less than a second. *arXiv*, 2024. 1, 2, 3, 5, 12, 13, 15

[9] Tom B Brown. Language models are few-shot learners. *arXiv*, 2020. 2

[10] Xinjing Cheng, Peng Wang, and Ruigang Yang. Learning depth with convolutional spatial propagation network. *IEEE TPAMI*, 42(10):2361–2379, 2019. 2

[11] Xinjing Cheng, Peng Wang, Chenye Guan, and Ruigang Yang. Cspn++: Learning context and resource aware convolutional spatial propagation networks for depth completion. In *AAAI*, pages 10615–10622, 2020. 2

[12] Jaehoon Cho, Dongbo Min, Youngjung Kim, and Kwanghoon Sohn. Diml/cvl rgb-d dataset: 2m rgb-d images of natural indoor and outdoor scenes. *arXiv*, 2021. 2

[13] Andrea Conti, Matteo Poggi, Valerio Cambareri, and Stefano Mattoccia. Depth on demand: Streaming dense depth from a low frame rate active sensor. *arXiv*, 2024. 2

[14] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, pages 2650–2658, 2015. 2

[15] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *NeurIPS*, 27, 2014. 2

[16] David Ferstl, Christian Reinbacher, Rene Ranftl, Matthias Rüther, and Horst Bischof. Image guided depth upsampling using anisotropic total generalized variation. In *ICCV*, pages 993–1000, 2013. 2

[17] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018. 2

[18] Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image. In *ECCV*, pages 241–258. Springer, 2025. 1, 2, 3, 12

[19] Vitor Guizilini, Igor Vasiljevic, Dian Chen, Rareș Ambruș, and Adrien Gaidon. Towards zero-shot scale-aware monocular depth estimation. In *ICCV*, pages 9233–9243, 2023. 1, 2

[20] Jing He, Haodong Li, Wei Yin, Yixun Liang, Leheng Li, Kaiqiang Zhou, Hongbo Liu, Bingbing Liu, and Ying-Cong Chen. Lotus: Diffusion-based visual foundation model for high-quality dense prediction. *arXiv*, 2024. 2, 3, 5

[21] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. *TPAMI*, 35(6):1397–1409, 2012. 2

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 16

[23] Derek Hoiem, Alexei A Efros, and Martial Hebert. Recovering surface layout from an image. *IJCV*, 75:151–172, 2007. 2

[24] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Hao Chen, Kaixuan Wang, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *TPAMI*, 2024. 1, 2, 5, 15

[25] Pu Hua, Minghuan Liu, Annabella Macaluso, Yunfeng Lin, Weinan Zhang, Huazhe Xu, and Lirui Wang. Gensim2: Scaling robot data generation with multi-modal and reasoning llms. *arXiv*, 2024. 8

[26] Benjamin Huhle, Timo Schairer, Philipp Jenke, and Wolfgang Straßer. Fusion of range and color images for denoising and resolution enhancement with a non-local filter. *Computer vision and image understanding*, 114(12):1336–1345, 2010. 2

[27] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pages 4401–4410, 2019. 4

[28] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *CVPR*, pages 9492–9502, 2024. 1, 2, 3, 5, 12

[29] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, pages 4015–4026, 2023. 3

[30] Johannes Kopf, Michael F Cohen, Dani Lischinski, and Matt Uyttendaele. Joint bilateral upsampling. *ACM TOG*, 26(3): 96–es, 2007. 2

[31] Yijin Li, Xinyang Liu, Wenqi Dong, Han Zhou, Hujun Bao, Guofeng Zhang, Yinda Zhang, and Zhaopeng Cui. Deltar: Depth estimation from a light-weight tof sensor and rgb image. In *ECCV*, pages 619–636. Springer, 2022. 3

[32] Zhengqi Li and Noah Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 2

[33] Zhenyu Li, Shariq Farooq Bhat, and Peter Wonka. Patchfusion: An end-to-end tile-based framework for high-resolution monocular metric depth estimation. 2024. 2

[34] Zhenyu Li, Shariq Farooq Bhat, and Peter Wonka. Patchrefiner: Leveraging synthetic data for real-domain high-resolution monocular metric depth estimation. 2024. 2

[35] Zhenyu Li, Xuyang Wang, Xianming Liu, and Junjun Jiang. BinsFormer: Revisiting adaptive bins for monocular depth estimation. *TIP*, 33:3964–3976, 2024. 2

[36] Yuankai Lin, Tao Cheng, Qi Zhong, Wending Zhou, and Hua Yang. Dynamic spatial propagation network for depth completion. In *AAAI*, pages 1638–1646, 2022. 2

[37] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023. 2

[38] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 2, 3

[39] Xin Liu, Xiaofei Shao, Bo Wang, Yali Li, and Shengjin Wang. Graphcspn: Geometry-aware depth completion via dynamic gcns. In *ECCV*, pages 90–107. Springer, 2022. 2

[40] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *ECCV*, 65(1):99–106, 2020. 4

[41] Jin-Hwi Park, Chanhwi Jeong, Junoh Lee, and Hae-Gon Jeon. Depth prompting for sensor-agnostic depth estimation. In *CVPR*, pages 9859–9869, 2024. 3, 5

[42] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, pages 4195–4205, 2023. 4

[43] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 4, 14

[44] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. UniDepth: Universal monocular metric depth estimation. In *CVPR*, 2024. 1, 2

[45] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *TPAMI*, 2020. 2

[46] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, pages 12179–12188, 2021. 2, 3

[47] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv*, 2024. 12

[48] Xuqian Ren, Wenjia Wang, Dingding Cai, Tuuli Tuominen, Juho Kannala, and Esa Rahtu. Mushroom: Multi-sensor hybrid room dataset for joint 3d reconstruction and novel view synthesis. In *WACV*, pages 4508–4517, 2024. 3

[49] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *ICCV*, 2021. 2, 4, 5, 14

[50] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 2446–2454, 2022. 2, 3

[51] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022. 3

[52] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. *TPAMI*, 31(5):824–840, 2008. 2

[53] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, pages 2446–2454, 2020. 2, 16

[54] Tao Sun, Mattia Segu, Janis Postels, Yuxuan Wang, Luc Van Gool, Bernt Schiele, Federico Tombari, and Fisher Yu. SHIFT: a synthetic driving dataset for continuous multi-task domain adaptation. In *CVPR*, pages 21371–21382, 2022. 16

[55] Zhanghao Sun, Wei Ye, Jinhui Xiong, Gyeongmin Choe, Jialiang Wang, Shuochen Su, and Rakesh Ranjan. Consistent direct time-of-flight video depth super-resolution. In *CVPR*, pages 5075–5085, 2023. 2

[56] Jie Tang, Fei-Peng Tian, Wei Feng, Jian Li, and Ping Tan. Learning guided convolutional network for depth completion. *IEEE TIP*, 30:1116–1129, 2020.

[57] Jie Tang, Fei-Peng Tian, Boshi An, Jian Li, and Ping Tan. Bilateral propagation network for depth completion. *CVPR*, 2024. 2, 5, 16

[58] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. 4, 14

[59] Chaoyang Wang, Simon Lucey, Federico Perazzi, and Oliver Wang. Web stereo video supervision for depth prediction from dynamic scenes. In *3DV*, 2019. 2

[60] Qiang Wang, Shizhen Zheng, Qingsong Yan, Fei Deng, Kaiyong Zhao, and Xiaowen Chu. Irs: A large naturalistic indoor robotics stereo dataset to train deep models for disparity and surface normal estimation. In *ICME*, 2021.

[61] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *IROS*, 2020. 2

[62] Chuhua Xian, Kun Qian, Zitian Zhang, and Charlie CL Wang. Multi-scale progressive fusion learning for depth map super-resolution. *arXiv*, 2020. 5, 7

[63] Ke Xian, Chunhua Shen, Zhiguo Cao, Hao Lu, Yang Xiao, Ruibo Li, and Zhenbo Luo. Monocular relative depth perception with web stereo data supervision. In *CVPR*, 2018. 2

[64] Ke Xian, Jianming Zhang, Oliver Wang, Long Mai, Zhe Lin, and Zhiguo Cao. Structure-guided ranking loss for single image depth prediction. In *CVPR*, 2020. 2

[65] Guangkai Xu, Wei Yin, Jianming Zhang, Oliver Wang, Simon Niklaus, Simon Chen, and Jia-Wang Bian. Towards domain-agnostic depth completion. *Machine Intelligence Research*, pages 1–18, 2024. 2

[66] Jingyu Yang, Xinchen Ye, Kun Li, Chunping Hou, and Yao Wang. Color-guided depth recovery from rgb-d data using an

adaptive autoregressive model. *IEEE TIP*, 23(8):3443–3458, 2014. 2

[67] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR*, pages 10371–10381, 2024. 1, 3, 5, 15

[68] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. 2024. 1, 2, 3, 4, 5, 7, 12, 14

[69] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *CVPR*, 2020. 2

[70] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, pages 12–22, 2023. 2, 3, 4, 5, 7, 12, 13, 14, 15

[71] Wei Yin, Jianming Zhang, Oliver Wang, Simon Niklaus, Long Mai, Simon Chen, and Chunhua Shen. Learning to recover 3d scene shape from a single image. In *CVPR*, pages 204–213, 2021. 2

[72] Wei Yin, Chi Zhang, Hao Chen, Zhipeng Cai, Gang Yu, Kaixuan Wang, Xiaozhi Chen, and Chunhua Shen. Metric3d: Towards zero-shot metric 3d prediction from a single image. In *CVPR*, pages 9043–9053, 2023. 1, 2, 3

[73] Tao Yu, Zerong Zheng, Kaiwen Guo, Pengpeng Liu, Qionghai Dai, and Yebin Liu. Function4d: Real-time human volumetric capture from very sparse consumer rgbd sensors. In *CVPR*, 2021. 12

[74] Yanjie Ze, Zixuan Chen, Wenhao Wang, Tianyi Chen, Xialin He, Ying Yuan, Xue Bin Peng, and Jiajun Wu. Generalizable humanoid manipulation with improved 3d diffusion policies. *arXiv*, 2024. 8

[75] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, pages 3836–3847, 2023. 2, 4, 14

[76] Youmin Zhang, Xianda Guo, Matteo Poggi, Zheng Zhu, Guan Huang, and Stefano Mattoccia. Completionformer: Depth completion with convolutions and vision transformers. In *CVPR*, pages 18527–18536, 2023. 2

[77] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *Robotics: Science and Systems*, 2023. 8, 16

Figure 9. Our accurate and high-resolution depth enables dynamic 3D reconstruction from a single moving camera. Here we illustrate the reconstruction results of a human walking in the library. The foreground is segmented with a SAM2 [47] model.

In the supplementary material, we present more discussions, additional results, and implementation details. Please find more video results in our supplementary video.

A. Additional Discussions

A.1. Generalizability to Different Resolutions

This section discusses the generalization capability of our model across different image and lidar depth resolutions provided by ARKit4 and ARKit6. ARKit4 captures images at a maximum resolution of 1440×1920 at 60Hz and lidar depth at 192×256, while ARKit6 captures images at a maximum resolution of 3024×4032 at 30Hz and lidar depth at 240×320. Both ScanNet++ [70] and ARKitScenes [3] are collected using ARKit4. Although our model is trained using ScanNet++ and ARKitScenes data at a resolution of 1440×1920, we find that it generalizes well to ARKit6 images and depth at different resolutions. As shown in Fig. 10, we include a comparison of depth estimation for images of different resolutions, with an image resolution of 2160×3840 and a lidar depth resolution of 144×256, captured from the ARKit6 API.

A.2. Why Do We Need Synthetic Data?

The advantages of synthetic data include high-quality ground truth depth, which has been crucial for the success of many recent depth estimation works [8, 18, 28, 68]. We also utilize synthetic data to achieve high-quality depth estimation results. Furthermore, the availability of real data with lost-cost LiDAR and high-power LiDAR is currently limited [3, 70], primarily to indoor scenes, while synthetic data can further enhance diversity; for instance, our experiments have shown that including human synthetic data [73] improves our method's generalization to human subjects.

	ARKitScenes		ScanNet++		
	L1 ↓	AbsRel ↓	Acc ↓	Comp ↓	F-Score ↑
(a) Depth Any. as foundation	0.0132	0.0115	0.0699	0.0616	0.7619
(b) Depth Pro as foundation	0.0169	0.0150	0.0754	0.0676	0.7202
(c) Depth Pro	0.1225	0.1038	0.0904	0.0760	0.6187

Table 5. **Additional quantitative ablations.** Please refer to Appendix A.4 for detailed descriptions.

A.3. Why Do We Need Real Data?

Training with real data can further address the inability of synthetic LiDAR simulation to replicate LiDAR noise patterns, thereby enhancing depth estimation capabilities. By utilizing synthetic data, we have achieved preliminary results. However, as demonstrated by the quantitative experiments in the main paper, the use of real data further enhances the performance. Here, we include additional qualitative results in Fig. 11, which show that real data is beneficial because LiDAR simulation methods cannot fully replicate the noise of real LiDAR.

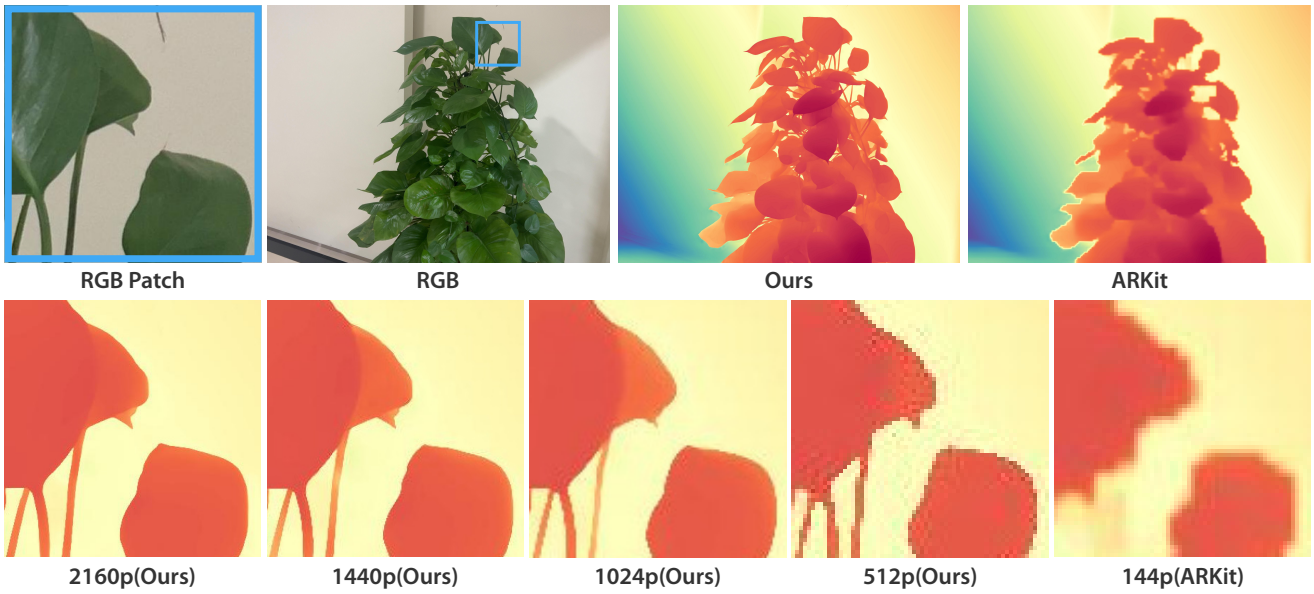

Figure 10. **Generalizability to different resolutions.** Our model can infer depth for images of different resolutions from 512p to 2160p.

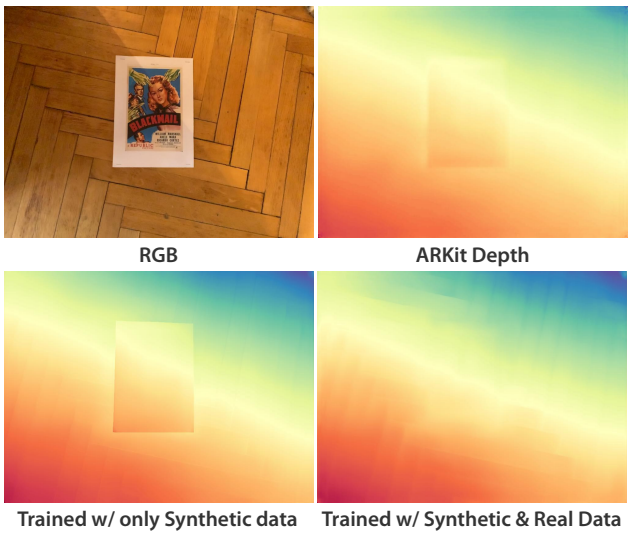

Figure 11. **Effects of using real data.**

A.4. Replacing Depth Foundation Models

Since our model is a general design for DPT, it can be easily adapted to other depth foundation models that also utilize the DPT structure, such as Depth Pro [8]. Our experiments demonstrate that it significantly enhances the performance of Depth Pro, as shown in Tab. 5-(b,c), although it does not outperform our choice of Depth Anything Tab. 5-(a).

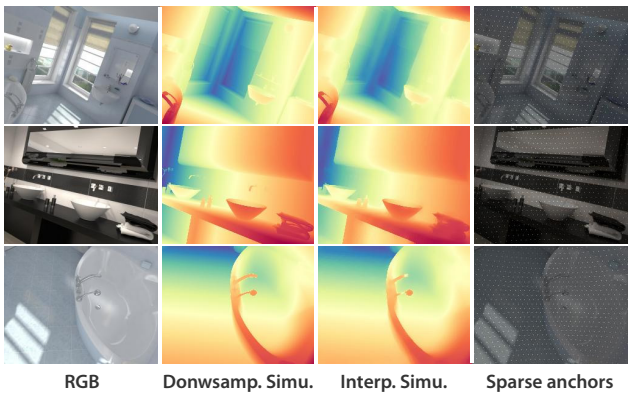

Figure 12. **Visualization results of simulated LiDAR.** "Interp. Simu." is the proposed interpolation method, which is interpolated from sparse anchors depth. This method effectively simulates the noise of real LiDAR data. We also provide the naive downsampled simulated LiDAR for comparison.

B. Additional Results

Visualization results of simulated LiDAR. We provide the visualization results of our simulated LiDAR in Fig. 12.

ZipNeRF reconstruction results. High-quality and dense observations are essential for effective 3D reconstruction. However, the iPhone data from ScanNet++ [70] frequently exhibits motion blur. To address this, we resample videos from ScanNet++ to remove blurring frames. Specifically, we calculate the variance of Laplacians for each image to assess its sharpness and use the sharpness score to select frames. For a 60fps video, we select one frame every 30

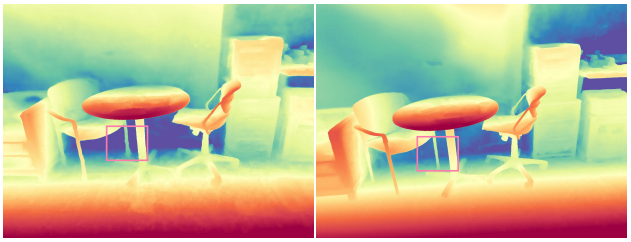

Figure 13. **ZipNeRF depth of different training frames.** Training with resampled frames removing blurred frames leads to a better ZipNeRF reconstruction.

frames, ensuring no repeated selection within any 6 consecutive frames, and guarantee at least one selection within every 2 seconds. We find that this method significantly reduces motion blur and leads to a better ZipNeRF reconstruction as shown in Fig. 13. Additionally, we utilize both the DSLR and iPhone data released by ScanNet++ to optimize ZipNeRF, which substantially improved our experimental results. Training ZipNeRF on ScanNet++ data required approximately $280 \times 2.5 \times 8$ GPU hours. We will release our processed data to benefit the research community.

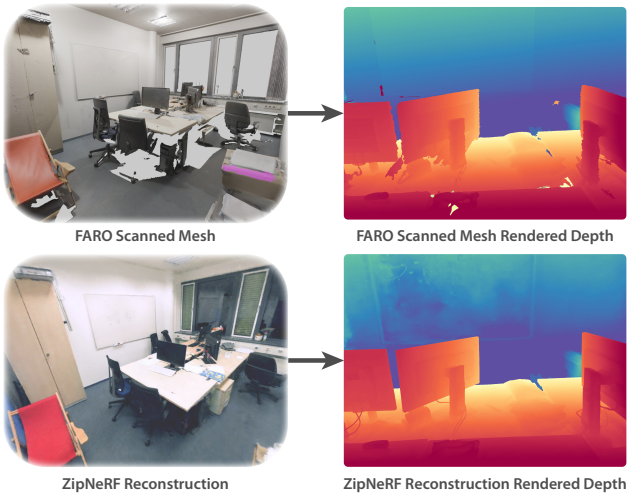

Figure 14. **Illustration of different depth annotation types.** Please refer to Appendix B for more descriptions.

Illustration of different annotation types. We provide an illustration of different annotation types in Fig. 14. Here we clearly observe the issues and advantages of different depth annotation types. The GT depth in ScanNet++ is annotated using a FARO scanned mesh. Due to the presence of many occlusions in the scene, the scanned mesh is incomplete, resulting in depth maps with numerous holes and poor edge quality. The pseudo GT depth annotated using NeRF reconstruction has accurate edges but performs poorly in planar regions. Therefore, an edge-aware loss is proposed to merge their advantages.

C. More Details

C.1. Details about Our Model

We employ the ViT-large model from Depth Anything v2 [68] as our backbone model. The shallow convolutional network consists of two convolutional layers, each with a kernel size of 3 and a stride of 1, utilizing ReLU as the nonlinear activation function. The zero-initialized projection layer is a 1×1 convolutional layer. For training on the ScanNet++ [70] dataset, we apply the loss function proposed in the main paper. For training on the ARKitScenes [3] dataset, we exclusively use the L1 loss. For training on synthetic [49] data, we employ both gradient and pixel-wise L1 loss simply from ground-truth depth supervision.

C.2. Optional Design Details

As mentioned in the main paper, in addition to the proposed design, we also explore optional designs including AdaLN [43], Cross-attention [58], and ControlNet [75]. We include a figure to illustrate these designs in Fig. 15. Our experiments (Tab. 3 in the main paper) show that ControlNet performs the best among these alternatives, but it is still not as effective as our proposed design. The plausible reason is that they are designed to integrate cross-modal information (e.g., text prompts), which does not effectively utilize the pixel alignment characteristics between the input low-resolution depth and the output depth. We also combine the proposed design with ControlNet to investigate potential further improvements. However, no additional improvements are observed (ours vs. combination are 0.730 vs. 0.731 in terms of F-score metric on ScanNet++), but the computational costs increase. Therefore, we keep the proposed design in the final version.

C.3. Evaluation Metrics

For depth metrics, we report L1, RMSE, AbsRel and $\delta_{0.5}$. Their definitions can be found in Tab. 6.

Metric	Definition
L1	$\frac{1}{N}\sum_{i=1}^{N}\lvert\mathbf{D}_i - \hat{\mathbf{D}}_i\rvert$
RMSE	$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\mathbf{D}_i - \hat{\mathbf{D}}_i)^2}$
AbsRel	$\frac{1}{N}\sum_{i=1}^{N}\lvert\mathbf{D}_i - \hat{\mathbf{D}}_i\rvert/\mathbf{D}_i$
$\delta_{0.5}$	$\frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left(\max\left(\frac{\mathbf{D}_i}{\hat{\mathbf{D}}_i}, \frac{\hat{\mathbf{D}}_i}{\mathbf{D}_i}\right) < 1.25^{0.5}\right)$

Table 6. **Depth metric definitions.** $\mathbf{D}$ and $\hat{\mathbf{D}}$ are the ground-truth and predicted depth, respectively. $\mathbb{I}$ is the indicator function.

For reconstruction metrics, we report Acc, Comp, Prec,

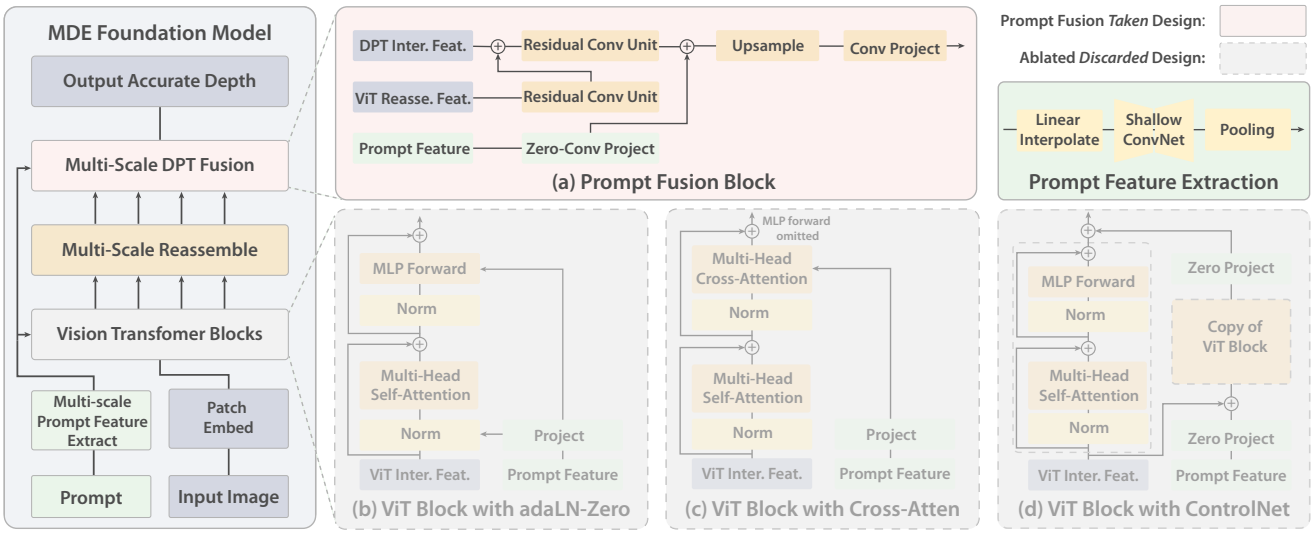

Figure 15. **Illustrations of our method and optional designs.** Please refer to Appendix C.2 for more details.

Recall, F-score. Their definitions can be found in Tab. 7. We use a voxel size of 0.04m for TSDF reconstruction.

Metric	Definition
Acc	$\text{mean}_{p \in P}(\min_{p^* \in P^*} \|\|p - p^*\|\|)$
Comp	$\text{mean}_{p^* \in P^*}(\min_{p \in P} \|\|p - p^*\|\|)$
Prec	$\text{mean}_{p \in P}(\min_{p^* \in P^*} \|\|p - p^*\|\| < .05)$
Recal	$\text{mean}_{p^* \in P^*}(\min_{p \in P} \|\|p - p^*\|\| < .05)$
F-score	$\frac{2 \times \text{Perc} \times \text{Recal}}{\text{Prec} + \text{Recal}}$

Table 7. **Reconstruction metric definitions.** P and P^* are the point clouds sampled from predicted and ground truth mesh.

C.4. Baseline Details

For the results presented in the main paper and supplementary video for Metric3D v2 [24] and Depth Pro [8], we input the ground-truth focal length into their models. The ZoeDepth* [6] model is trained using reproduced code from Depth Anything v1 [67], and we utilize the base model of Depth Anything v1 for conducting experiments. The MSPF results for ARKitScenes dataset are taken from [3], and we retrain it using ScanNet++ [70] training data for testing on Scannet++ with the reproduced code from ARKitScenes.

C.5. Ransac Alignment Details

For monocular depth estimation methods, we perform a post-alignment to ensure fair comparison. We utilize RANSAC alignment to align their output depth with the iPhone LiDAR depth. Specifically, we first resize the output depth to match the dimensions of the iPhone LiDAR depth, then randomly formed several groups of samples. Each group of sample points is used to calculate a scale and shift, followed by voting using all points. The voting threshold is set to the median of the differences between the entire set of numbers and the median(Median Absolute Deviation). Then we apply the scale and shift to the predicted depth to align it with the ground-truth depth. This method is more robust compared to the commonly used polyfit alignment in monocular depth estimation, typically improving the F-score by 8-10% on ScanNet++ dataset.

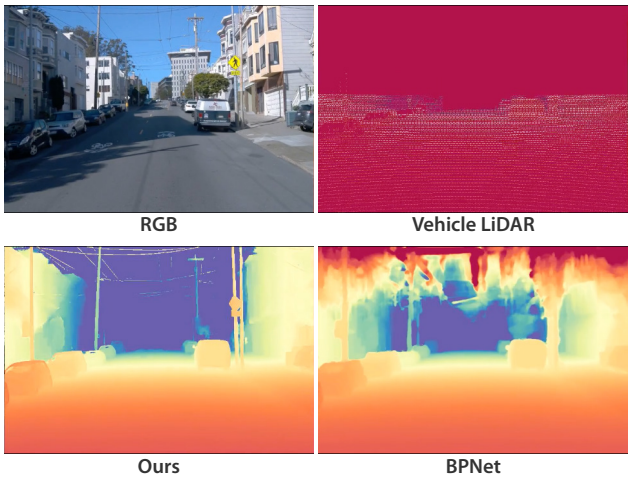

Figure 16. **Qualitative comparison of vehicle LiDAR completion.** We include more video results in the supplementary video.

D. Prompting with a vehicle LiDAR

We evaluate our method on the Waymo dataset to assess its performance with vehicle LiDAR. Vehicle LiDAR significantly differs from the LiDAR used in smartphones, as it is generally coarse and consists of X-beam sparse LiDAR

(typically 64 beams for Waymo dataset [53]). Therefore, before inputting the data into the network, we perform KNN completion on the vehicle LiDAR depth ($k = 4$). We train our model on the Shift dataset [54], a synthetic dataset designed for autonomous driving, which includes RGB and depth data. The LiDAR data is simulated using the approach detailed in the main paper. We evaluate our model on the Waymo dataset. We make comparisons with BP-Net [57] in Fig. 16. Our method demonstrates precise depth estimation and we include more video results and street reconstruction results in the supplementary video.

E. Generalized Robotic Grasping Details

Detailed setups. We control the right arm of a Unitree H1 humanoid robot while fixing its lower body. The task is to grasp the object on the table and put it into the box, one at a time. The object is randomly placed at nearby, middle, and far positions. The robot policy runs at 30 Hz. However, due to overheating issues in our lab environment, the iPhone can only stably capture images at 15 Hz, resulting in the visual input being updated every two control steps.

We first teleoperate the robot to collect 60, 80 trajectories for diffusive objects (red & green cans) and transparent objects (glass bottles); then, we take the diffusive set of data as training set to train ACT [77] policies with different types of visual inputs, including the estimated depth by our model, ARKit depth directly from the iPhone, and also RGB images; during evaluation, we test the grasping performance corresponding to different visual inputs on all objects.

Model architectures. We use the same network structure with ACT [77] with one image input. ACT policy crops all types of visual input at 480x640 resolution and processes images with a pre-trained ResNet18 backbone[22]. For depth images, the first layer of the pre-trained network is replaced with a 1-channel convolutional network. The pretrained ResNet18 helps enhance the generalization of policy. Without the pretrained parameters, the policy with depth input only grasps the same position.

We include more video results in the supplementary video.