Depth Any Camera: Zero-Shot Metric Depth Estimation from Any Camera

Supplementary Material

Figure 7. **Pitch-Aware Image-to-ERP Conversion.** *Top*: The original images, taking HM3D [35] samples for examples. *Middle*: ERP patches converted from the original images without camera pitch awareness by setting tangent image center at latitude $\lambda_c = 0$. *Bottom*: ERP patches prepared via camera pitch-aware ERP conversion, where in our convention $\lambda_c = -\text{Pitch}$.

7. Supplemental Experiments

7.1. Full Zero-Shot Metric Depth Experiments

Full experiments with a few additional experiments comparing DAC to the state-of-the-arts in zero-short metric depth estimation are shown in Table 5. The additional experiments include:

- **Zero-Shot to Perspective Data.** In addition to the large FoV dataset results presented in the main text, we include evaluations on two widely tested perspective datasets, NYUv2 [30] and KITTI [12], to demonstrate that our method can also achieve zero-shot generalization on standard perspective datasets. Notably, DAC outperforms iDisc [32] trained with the Metric3Dv2 [20] pipeline, which we attribute to DAC's ability to leverage the synergy of diverse data with varying FoVs and pitch coverage. The remaining gap compared to the state-of-the-art is likely due to the significantly smaller training dataset and the smaller SwinL [28] backbone used in DAC compared to the larger ViT-L [7] backbones adopted by other methods.
- **DAC with SwinL [28] Backbone.** We also update our DAC model and iDisc model with a larger backbone, Swin-L [28], to further showcase the performance of our approach when scaling to larger models. Note that the Swin-L backbone remains smaller than the Dinov2-ViT-L [31] backbone used in Metric3Dv2 [20], and as well the ViT-L [7] backbone applied in UniDepth [33]. As observed, although Swin-L-based DAC models lead to significant improvements on generalization to NYU and KITTI360 datasets, their improvements on Scannet++ and KITTI datasets are marginal, and they under perform Resnet101 counterparts on 360° datasets. We interpreter the reason is that transformer backbones are designed for scale-invariance reasoning rather than for the scale-equivariance inference required in 3D tasks. More adapted design of transformer architectures are demanding for further push the upper bound of training of foundation depth models.

7.2. Full Modular Ablation Study

Table 6 presents the complete experimental results for the ablation study of DAC's key components: **pitch-aware ERP conversion** and **pitch augmentation**, **FoV-Align**, and **Multi-Reso Training**. It also includes comparisons to alternative network architectures and training frameworks. All the methods presented in this table are training on HM3D-tiny [35] including about 300K samples. iDisc [32]-based and DAC models are all based on Resnet101 [18] backbone, and trained with 40K iterations with batch size 48. While Metric3Dv2 [20] model is based on its original Dinov2-ViT-L [31] backbone, trained on the same dataset with 120K iterations and batch size 48.

The **pitch-aware ERP conversion** and **ERP-space pitch augmentation** ablations, highlight the effectiveness of our core Image-to-ERP conversion in enabling the DAC framework. As shown in Table 6, pitch-aware ERP conversion plays a pivotal role in generalizing perspective-trained models to large FoV datasets. This capability stems from projecting input images to different latitude regions of the ERP space—areas typically visible only in large FoV data—illustrated in Fig. 7. By leveraging this approach, the wide pitch angle variance in datasets like HM3D [35] becomes a strength rather than a challenge.

Note that the camera orientations wrt. the world coordinates can be either provided by the dataset [35, 39, 61], or estimated from tradition geometry [40] or recent deep learning models [22]. Since our training process is usually integrated with ERP space geometric augmentations, our framework do not require the camera pose estimation very accurate for the purpose of depth estimation.

Additionally, ERP-space pitch augmentation provides marginal improvements for 360° datasets and minimal gains for Scannet++ fisheye data, likely because HM3D-tiny already includes a sufficiently broad pitch span.

7.3. Full Ablation Study on Training Dataset

In Table 7, we show the full ablation study on the impact of different datasets. Different training dataset, due to its different span in camera FoVs, pitch angles, image quality, etc., contribute differently on different testing data. Our DAC framework can leverage the synergy between very diverse datasets to significantly boost the overall performance to all the testing datasets.

In addition to the main content summarized in the paper,we include an ablation study on the impact of **pitch-aware ERP conversion** and **ERP-space pitch augmentation** to evaluate their effectiveness across different training datasets.

The results indicate that pitch-aware ERP conversion is crucial for DAC's generalization across almost all configurations of training and testing datasets. This remains true even when the training dataset has a limited range of camera pitch angles, such as Taskonomy [61]. Moreover, its impact becomes more pronounced as the diversity of pitch angles in the training dataset increases. In contrast, ERP-space pitch augmentation proves significant primarily

Table 5. **Zero-Shot Metric Depth Evaluation on 360°, Fisheye, and Perspective Datasets.** This table compares DAC with leading state-of-the-art metric depth models across metric depth benchmarks, upon Resnet101 [18] and SwinL [28] backbones.

Test Dataset	Methods	Train Dataset	Backbone	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	Abs Rel↓	RMSE↓	log10↓
Matterport3D [5]	UniDepth [33]	Mix 3M	ViT-L [7]	0.2576	0.5114	0.7091	0.7648	1.3827	0.2208
	Metric3Dv2 [20]	Mix 16M	Dinov2-ViT-L [31]	0.4381	0.7311	0.8735	0.2924	0.8842	0.1546
	Metric3Dv2 [20]	Indoor 670K	Dinov2-ViT-L [31]	0.4287	0.7854	0.9333	0.2788	0.8961	0.1352
	iDisc [32]	Indoor 670K	Resnet101 [18]	0.5287	0.8260	0.9398	0.2757	0.7771	0.1147
	iDisc [32]	Indoor 670K	SwinL [28]	0.5865	0.8722	0.9599	0.2272	0.6612	0.1021
	DAC (Ours)	Indoor 670K	Resnet101 [18]	**0.7727**	**0.9562**	**0.9822**	**0.156**	**0.6185**	**0.0707**
	DAC (Ours)	Indoor 670K	SwinL [28]	0.7231	0.949	0.9866	0.1789	0.5911	0.0741
Pano3D-GV2 [2]	UniDepth [33]	Mix 3M	ViT-L [7]	0.2469	0.4977	0.7084	0.7892	1.2681	0.2231
	Metric3Dv2 [20]	16M	Dinov2-ViT-L [31]	0.4040	0.6929	0.8499	0.3070	0.8549	0.1664
	Metric3Dv2 [20]	Indoor 670K	Dinov2-ViT-L [31]	0.5060	0.8176	0.9360	0.2608	0.7248	0.1201
	iDisc [32]	Indoor 670K	Resnet101 [18]	0.5629	0.8222	0.9332	0.2657	0.6446	0.1122
	iDisc [32]	Indoor 670K	SwinL [28]	0.6022	0.8528	0.9447	0.2272	0.5680	0.1035
	DAC (Ours)	Indoor 670K	Resnet101 [18]	**0.8115**	**0.9549**	**0.9860**	**0.1387**	**0.4780**	**0.0623**
	DAC (Ours)	Indoor 670K	SwinL [28]	0.7287	0.9307	0.9793	0.1836	0.4833	0.077
ScanNet++ [56]	UniDepth [33]	Mix 3M	ViT-L [7]	0.3638	0.6461	0.8358	0.4971	1.1659	0.1648
	Metric3Dv2 [20]	Mix 16M	Dinov2-ViT-L [31]	0.5360	0.8218	0.9350	0.2229	0.8950	0.1177
	Metric3Dv2 [20]	Indoor 670K	Dinov2-ViT-L [31]	0.6489	0.8920	0.9558	0.1915	0.9779	0.0938
	iDisc [32]	Indoor 670K	Resnet101 [18]	0.6150	0.8780	0.9617	0.2712	0.4835	0.0972
	iDisc [32]	Indoor 670K	Swinl [28]	0.7746	0.9439	0.9862	0.1741	0.3634	0.0680
	DAC (Ours)	Indoor 670K	Resnet101 [18]	0.8517	0.9693	0.9922	0.1323	0.3086	0.0532
	DAC (Ours)	Indoor 670K	SwinL [28]	**0.8544**	**0.9776**	**0.9939**	**0.1282**	**0.2866**	**0.0518**
KITTI360 [27]	UniDepth [33]	Mix 3M	ViT-L [7]	0.4810	0.8397	0.9406	0.2939	6.5642	0.1221
	Metric3Dv2 [20]	Mix 16M	Dinov2-ViT-L [31]	0.7159	0.9323	0.9771	0.1997	4.5769	0.0811
	Metric3Dv2 [20]	Outdoor 130K	Dinov2-ViT-L [31]	0.7675	0.9370	0.9756	0.1521	4.6610	0.0723
	iDisc [32]	Outdoor 130K	Resnet101 [18]	0.7833	0.9384	0.9753	0.1598	4.9122	0.0704
	iDisc [32]	Outdoor 130K	SwinL [28]	0.8165	0.9533	0.9829	0.1500	4.2549	0.0620
	DAC (Ours)	Outdoor 130K	Resnet101 [18]	0.7858	0.9388	0.9775	0.1559	4.3614	0.0684
	DAC (Ours)	Outdoor 130K	SwinL [28]	**0.8222**	**0.9571**	**0.9845**	**0.1487**	**3.7510**	**0.0607**
NYUv2 [30]	UniDepth [33]	Mix 3M	ViT-L [7]	**0.9875**	**0.9982**	**0.9995**	**0.052**	**0.1936**	**0.0223**
	Metric3Dv2 [20]	Mix 16M	Dinov2-ViT-L [31]	0.9718	0.9929	0.9971	0.0666	0.2621	0.0290
	Metric3Dv2 [20]	Indoor 670K	Dinov2-ViT-L [31]	0.9422	0.9885	0.9966	0.0936	0.3359	0.0388
	iDisc [32]	Indoor 670K	Resnet101 [18]	0.691	0.9028	0.9675	0.1755	0.6193	0.0838
	iDisc [32]	Indoor 670K	SwinL [28]	0.8319	0.9629	0.9891	0.1239	0.4690	0.0571
	DAC (Ours)	Indoor 670K	Resnet101 [18]	0.719	0.9324	0.985	0.1641	0.6189	0.0755
	DAC (Ours)	Indoor 670K	SwinL [28]	0.8673	0.975	0.9921	0.1187	0.4471	0.0511
KITTI [12]	UniDepth [33]	Mix 3M	ViT-L [7]	0.9643	**0.9973**	**0.9993**	0.1159	2.7881	0.047
	Metric3Dv2 [20]	Mix 16M	Dinov2-ViT-L [31]	**0.9742**	0.9954	0.9987	**0.0534**	**2.4932**	**0.0234**
	Metric3Dv2 [20]	Outdoor 130K	Dinov2-ViT-L [31]	0.9488	0.9918	0.9975	0.0848	3.1426	0.0375
	iDisc [32]	Outdoor 130K	Resnet101 [18]	0.8503	0.9626	0.9897	0.1277	4.5347	0.0528
	iDisc [32]	Outdoor 130K	SwinL [28]	0.8382	0.9682	0.993	0.1439	4.5267	0.0575
	DAC (Ours)	Outdoor 130K	Resnet101 [18]	0.8767	0.9744	0.9934	0.1155	4.3877	0.0488
	DAC (Ours)	Outdoor 130K	SwinL [28]	0.8912	0.9785	0.9947	0.1058	4.1699	0.0435

when the original training dataset lacks diversity in pitch angles. However, its contribution diminishes when the training data already encompass a wide range of pitch angles.

7.4. Zero-Shot Test of Perspective Depth Model on Distorted Images

As shown in Table 8, we evaluate Metric3D [20] on different representations of KITTI360's fisheye images including raw fisheye, the ERP conversion of fisheye, undistorted fisheye with three different FoVs. The evaluation results align with the visual examples in Figure 2, demonstrating that perspective-trained metric depth models perform poorly on fisheye data. While undistorted camera representations sacrifice significant FoV or severs interpolating artifacts, applying a virtual focal length $\frac{1}{f_{\text{virtual}}} = \tan\left(\frac{\pi}{H_{\text{erp}}}\right)$ to raw fisheye images or their ERP conversions results in even greater performance degradation. To ensure a fair comparison between DAC and pre-trained perspective models, we apply ERP conversion during fisheye testing for the perspective models as well, given that neither representation—raw fisheye nor ERP—falls within their original camera domain.

8. On Applying Camera Distortion Models

As described in Sec. 4.1, the conversion between actual image and the ERP can seamlessly handle different distortion models. In this section, we illustrate how we apply to two typical fisheye models: KB (OpenCV Fisheye) model [23] and MEI model [29].

Table 6. **Impact of Key Components and Network.** We conduct the main ablation study on indoor datasets by training with HM3D [35] and performing zero-shot testing on Pano3D-GV2 [2] and ScanNet++[56]. We compare the performance of the DAC framework with specific components removed, as well as different network architectures trained under the Metric3D[57] pipeline. Four key components of our DAC framework are included in the ablation study.

Test Datasets	Methods	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	Abs Rel$\downarrow$	RMSE$\downarrow$	log10$\downarrow$
Matterport3D [5]	Metric3Dv2 [20]	0.4879	0.8196	0.9443	0.2631	0.8556	0.1214
	iDisc-cnn [32]	0.3574	0.6355	0.8051	0.3202	1.3369	0.1854
	iDisc [32]	0.4303	0.7325	0.8777	0.3109	1.1876	0.1508
	DAC (Ours)	**0.728**	**0.9372**	**0.9761**	**0.1699**	**0.718**	**0.0774**
	w\o Pitch-Aware ERP	0.5394	0.8358	0.9442	0.2222	0.8383	0.1134
	w\o Pitch Aug 10°	0.7152	0.9379	0.9797	0.1816	0.7134	0.0789
	w\o FoV Align	0.4494	0.7962	0.9206	0.2446	1.0383	0.1331
	w\o Multi-Reso	0.5670	0.8476	0.9343	0.2219	0.9658	0.1132
Pano3D-GV2 [2]	Metric3Dv2 [20]	0.5623	0.8341	0.9396	0.2479	0.7332	0.1113
	iDisc-cnn [32]	0.3026	0.5565	0.7337	0.3548	1.2307	0.2118
	iDisc [32]	0.413	0.6844	0.8397	0.3043	1.0649	0.162
	DAC (Ours)	**0.7251**	**0.9254**	**0.9747**	**0.1729**	**0.6015**	**0.0786**
	w\o Pitch-Aware ERP	0.4911	0.7904	0.9193	0.2422	0.7521	0.1262
	w\o Pitch Aug 10°	0.6912	0.9311	0.977	0.188	0.5966	0.0819
	w\o FoV Align	0.4075	0.7585	0.9085	0.261	0.9148	0.1415
	w\o Multi-Reso	0.5128	0.7784	0.8977	0.2437	0.8867	0.1298
ScanNet++ [56]	Metric3Dv2 [20]	0.3865	0.6730	0.8229	0.3129	1.3277	0.1705
	iDisc-cnn [32]	0.4639	0.7653	0.8965	0.3045	1.3116	0.1395
	iDisc [32]	0.5301	0.8048	0.9165	0.3237	1.552	0.1251
	DAC (Ours)	**0.6539**	**0.9083**	**0.9722**	**0.1951**	**0.5926**	**0.089**
	w\o Pitch-Aware ERP	0.4711	0.8068	0.9282	0.2508	0.7925	0.127
	w\o Pitch Aug 10°	0.6741	0.9066	0.9701	0.1914	0.5966	0.0861
	w\o FoV Align	0.5428	0.8644	0.9544	0.22	0.71	0.1091
	w\o Multi-Reso	0.5504	0.8464	0.942	0.2231	0.7435	0.1116

8.1. KB Model

KB model typically includes distortion parameters k_1, k_2, k_3, k_4. Applying KB model to our Eq. 4 can start from mapping our definition in Eq. 1 and Eq. 2 to the original KB model notations to get:

$$a = x_t, \quad b = y_t \tag{7}$$

$$r = \sqrt{x_t^2 + y_t^2} \tag{8}$$

$$\theta = \arctan(r) = c \tag{9}$$

However, the direct use of (x_t, y_t) can face numerical issue when the FOV is near 180°, when the dividing of $\cos c$ approaches 0 in computing them. A more numerical stable version supporting KB at 180° is to use the numerators in Eq. 1 and Eq. 2, denoted as $(\bar{x}, \bar{y})$. Then we can rewrite:

$$a = \bar{x}, \quad b = \bar{y} \tag{10}$$

$$r = \sqrt{\bar{x}^2 + \bar{y}^2} \tag{11}$$

$$\theta = c \tag{12}$$

where we can keep the ratios $\frac{a}{r}, \frac{a}{r}$ consistent between two approaches, while avoiding numeric issues caused by dividing $\cos 0$.

The remaining process is exactly the same as the original KB model. **Fisheye distortion** is applied as:

$$\theta_d = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8) \tag{13}$$

The distorted point coordinates are $[x', y']$ where

$$x_d = \left(\frac{\theta_d}{r}\right) a \tag{14}$$

$$y_d = \left(\frac{\theta_d}{r}\right) b \tag{15}$$

Finally, given a intrinsic model including $f_x, f_y, c_x, c_y, \alpha$ as parameters, the conversion into pixel coordinates $[u, v]$ can be written as:

$$u = f_x(x_d + \alpha y_d) + c_x \tag{16}$$

$$v = f_y y_d + c_y \tag{17}$$

8.2. MEI Model

MEI model is general more complex by including parameters ξ, k_1, k_2, p_1, p_2, where an additional shift parameter ξ is applied so that the model handle even larger FOV camera, and p_1, p_2 are including tangential distortion.

Mapping our definitions to MEI model is even simpler. Note that $(\bar{x}, \bar{y}, \cos c)$ actually describe a point lying on the unit sphere, equalizing the Cartesian coordinates converted from the spherical coordinates. The projection coordinates (p_u, p_v) are computed as:

$$p_u = \frac{\bar{x}}{\cos c + \xi} \tag{18}$$

$$p_v = \frac{\bar{y}}{\cos c + \xi} \tag{19}$$

Table 7. **Ablation Study of training datasets.** Models are trained separately on each training dataset and evaluated in zero-shot tests on 360° and fisheye datasets. In addition, the ablation study on the impact of pitch-aware ERP conversion and ERP-space pitch augmentation are included to further analysis their contribution under different training distributions.

Test Datasets	Train Dataset	Methods	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	Abs Rel↓	RMSE↓	log10↓
Matterport3D [5]	HM3D-tiny [35] 310K	Metric3Dv2 [20]	0.4879	0.8196	0.9443	0.2631	0.8556	0.1214
		iDisc [32]	0.4303	0.7325	0.8777	0.3109	1.1876	0.1508
		DAC (Ours)	**0.728**	**0.9372**	**0.9761**	**0.1699**	**0.718**	**0.0774**
		w\o Pitch-Aware ERP	0.5394	0.8358	0.9442	0.2222	0.8383	0.1134
		w\o Pitch Aug 10°	0.7152	0.9379	0.9797	0.1816	0.7134	0.0789
	Taskonomy-tiny [61] 300K	Metric3Dv2 [20]	0.3244	0.6652	0.8958	0.3145	1.0727	0.1711
		iDisc [32]	0.3662	0.6538	0.8205	0.4186	2.3299	0.1787
		DAC (Ours)	**0.5363**	**0.8537**	**0.9371**	**0.232**	**0.8194**	**0.115**
		w\o Pitch-Aware ERP	0.4018	0.7576	0.894	0.2722	0.9377	0.1471
		w\o Pitch Aug 10°	0.4244	0.7633	0.9019	0.2689	0.9199	0.1428
	Hypersim [39] 60k	Metric3Dv2 [20]	0.3740	0.6746	0.8450	0.5082	1.0822	0.1637
		iDisc [32]	0.3624	0.6792	0.8757	0.315	1.0425	0.1638
		DAC (Ours)	**0.4491**	**0.8066**	**0.9438**	**0.2659**	**0.8574**	**0.1271**
		w\o Pitch-Aware ERP	0.4098	0.7526	0.9129	0.2772	0.9437	0.1431
		w\o Pitch Aug 10°	0.4577	0.834	0.9524	0.2513	0.8926	0.1206
Pano3D-GV2 [2]	HM3D-tiny [35] 310K	Metric3Dv2 [20]	0.5623	0.8341	0.9396	0.2479	0.7332	0.1113
		iDisc [32]	0.413	0.6844	0.8397	0.3043	1.0649	0.162
		DAC (Ours)	**0.7251**	**0.9254**	**0.9747**	**0.1729**	**0.6015**	**0.0786**
		w\o Pitch-Aware ERP	0.4911	0.7904	0.9193	0.2422	0.7521	0.1262
		w\o Pitch Aug 10°	0.6912	0.9311	0.977	0.188	0.5966	0.0819
	Taskonomy-tiny [61] 300K	Metric3Dv2 [20]	0.3785	0.7489	0.9062	0.2959	0.8945	0.1550
		iDisc [32]	0.3888	0.6816	0.8349	0.4076	2.1877	0.1683
		DAC (Ours)	**0.6411**	**0.8719**	**0.9452**	**0.1972**	**0.6148**	**0.0982**
		w\o Pitch-Aware ERP	0.4828	0.7882	0.9026	0.2465	0.7345	0.1323
		w\o Pitch Aug 10°	0.4954	0.7947	0.9077	0.2411	0.7197	0.1289
	Hypersim [39] 60k	Metric3Dv2 [20]	0.3085	0.6382	0.8147	0.5583	1.1762	0.1887
		iDisc [32]	0.3372	0.6473	0.831	0.3288	0.9098	0.177
		DAC (Ours)	**0.5208**	**0.8295**	**0.9424**	**0.1792**	**0.6873**	**0.1158**
		w\o Pitch-Aware ERP	0.4486	0.7655	0.9025	0.2707	0.7823	0.1385
		w\o Pitch Aug 10°	0.5293	0.8525	0.9504	0.2344	0.7212	0.1123
ScanNet++ [56]	HM3D-tiny [35] 310K	Metric3Dv2 [20]	0.3799	0.6310	0.7801	0.6090	1.0490	0.1899
		iDisc [32]	0.5301	0.8048	0.9165	0.3237	1.552	0.1251
		DAC (Ours)	**0.6539**	**0.9083**	**0.9722**	**0.1951**	**0.5926**	**0.089**
		w\o Pitch-Aware ERP	0.4711	0.8068	0.9282	0.2508	0.7925	0.127
		w\o Pitch Aug 10°	0.6741	0.9066	0.9701	0.1914	0.5966	0.0861
	Taskonomy-tiny [61] 300K	Metric3Dv2 [20]	0.6421	0.8377	0.9285	0.3840	2.2102	0.1075
		iDisc [32]	0.6743	0.9179	0.9809	0.1977	0.5235	0.083
		DAC (Ours)	**0.7981**	**0.9666**	**0.9898**	**0.1447**	**0.3556**	**0.0637**
		w\o Pitch-Aware ERP	0.7642	0.9561	0.9879	0.1542	0.3881	0.0705
		w\o Pitch Aug 10°	0.7673	0.9534	0.9892	0.1516	0.3861	0.0694
	Hypersim [39] 60k	Metric3Dv2 [20]	0.5684	0.8149	0.9173	0.3364	0.5289	0.1192
		iDisc [32]	0.6656	0.9004	0.9701	0.2213	0.5471	0.0872
		DAC (Ours)	**0.7478**	**0.9483**	**0.9871**	**0.1762**	**0.4124**	**0.0729**
		w\o Pitch-Aware ERP	0.7238	0.9236	0.9801	0.1959	0.4375	0.0778
		w\o Pitch Aug 10°	0.7439	0.9396	0.9844	0.1846	0.4106	0.0732

The distortion is then applied as:

$$\rho^2 = p_u^2 + p_v^2 \tag{20}$$

$$p_u \leftarrow p_u \cdot (1 + k_1\rho^2 + k_2\rho^4) \tag{21}$$

$$p_v \leftarrow p_v \cdot (1 + k_1\rho^2 + k_2\rho^4) \tag{22}$$

Tangential distortion is further applied as:

$$x_d \leftarrow p_u + 2p_1p_up_v + p_2(\rho^2 + 2p_u^2) \tag{23}$$

$$y_d \leftarrow p_v + p_1(\rho^2 + 2p_v^2) + 2p_2p_up_v \tag{24}$$

The later projection is applied the same way as KB model.

Table 8. **Pretrained model performance on various representations of KITTI 360 dataset [27]**

Representation	Methods	Train Dataset	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	Abs Rel$\downarrow$	RMSE$\downarrow$	log10$\downarrow$
KITTI 360 Raw (FOV 180)	Metric3Dv2 [20]	Mix 16M	0.7421	0.9498	0.9829	0.1679	3.0873	0.0739
	Metric3Dv2 [20]	Outdoor 130K	0.6400	0.9077	0.9763	0.1884	3.5698	0.0902
KITTI 360 ERP (FOV 180)	Metric3Dv2 [20]	Mix 16M	0.7159	0.9323	0.9770	0.1997	4.5769	0.0811
	Metric3Dv2 [20]	Outdoor 130K	0.7675	0.9370	0.9756	0.1521	4.6610	0.0723
KITTI 360 UD FoV 90	Metric3Dv2 [20]	Mix 16M	0.7581	0.9533	0.9738	0.1652	2.1454	0.0799
	Metric3Dv2 [20]	Outdoor 130K	0.8099	0.9582	0.9807	0.1469	2.1203	0.0650
KITTI 360 UD FoV 120	Metric3Dv2 [20]	Mix 16M	0.6398	0.9285	0.9717	0.1929	2.3375	0.0968
	Metric3Dv2 [20]	Outdoor 130K	0.6635	0.9019	0.9685	0.1865	2.5982	0.0929
KITTI 360 UD FoV 150	Metric3Dv2 [20]	Mix 16M	0.4840	0.8533	0.9551	0.2311	2.8692	0.1210
	Metric3Dv2 [20]	Outdoor 130K	0.4565	0.7788	0.9041	0.2498	3.2509	0.1355

9. Efficient Up-Projection from Distorted Cameras via Lookup Table Approximation

Up-projection is a crucial step to convert predicted depth maps into 3D point clouds. For perspective or ERP images, this process is straightforward, as the 3D ray associated with each pixel can be computed in closed form. However, up-projection from fisheye depth maps poses challenges due to the need to invert the distortion model, often requiring the solution of a high-order polynomial equation for each pixel based on the distortion parameters. This process is computationally expensive and impractical for real-time applications.

Fortunately, pre-computed lookup tables can address this issue efficiently. These tables store a mapping from 2D image coordinates to 3D ray directions, allowing for real-time up-projection, which can be written as:

$$\mathbf{L} : \mathbb{R}^2 \rightarrow \mathbb{R}^3, \quad \mathbf{L}(\mathbf{u}) = \mathbf{r}, \tag{25}$$

where $\mathbf{L}$ represents the lookup table, $\mathbf{u} = (u, v) \in \mathbb{R}^2$ denotes the 2D image coordinates, and $\mathbf{r} = (x, y, z) \in \mathbb{R}^3$ represents the corresponding 3D ray direction. The lookup tables can be generated using tools like OpenCV with gradient-based numerical methods or through simpler grid search approaches when tangential distortion parameters are negligible [27]. in this work, we use similar grid search approach to computed lookup tables for Scannet++ [56] based on their provided distortion and intrinsic parameters.

Notably, our DAC framework does not require approximated solutions for up-projection. In DAC, fisheye images are converted into ERP patches, which rely only on the forward distortion model. The resulting ERP depth maps can then be up-projected into 3D point clouds using each ERP coordinate's ray direction in a unit sphere, eliminating efficiency concerns. This represents a minor but valuable benefit of our approach.

Nevertheless, we identify two practical use cases for lookup tables in other contexts:

- Visualization Purposes: Lookup tables efficiently map ERP patches and predicted ERP depth maps back to the original fisheye space for visualization, as illustrated in Fig. 6. Specifically, ERP-to-image conversion for a fisheye image can also be performed efficiently using grid sampling, where each fisheye image coordinate is mapped to its floating-point location in the ERP space. The output of Eq. 25 already provides tangent plane normalized coordinates, $x_t = \frac{x}{z}$ and $y_t = \frac{y}{z}$. Using the inverse of Gnomonic Geometry [47], the mapping to spherical coordinates (λ, ϕ) is derived as follows:

$$\phi = \sin^{-1}\left(\cos c \sin \phi_c + \frac{y_t \sin c \cos \phi_c}{\rho}\right) \tag{26}$$

$$\lambda = \lambda_c + \tan^{-1}\left(\frac{x_t \sin c}{\rho \cos \phi_c \cos c - y_t \sin \phi_c \sin c}\right) \tag{27}$$

where

$$\rho = \sqrt{x_t^2 + y_t^2}$$
$$c = \tan^{-1} \rho$$

However, this step is only needed for visualization purpose, not required for downstream tasks where up-projected 3D points are the most demanding.
- Converting Z-Values to Euclidean Distances: For datasets like ScanNet++ [56], ground-truth depth maps recorded in Z-values must be converted to Euclidean distances for evaluation or inclusion in DAC training. This can be achieved efficiently using pre-computed ray directions from the fisheye's original incoming rays (not distorted by intrinsic parameters). The Euclidean distance for each pixel is calculated as: $D_{\text{Euclid}} = \frac{Z}{z}$, where Z represents the ground-truth Z-value, and z is the z-component of the ray direction $\mathbf{r}$.

10. Additional Visual Results

In this section, we provide three additional set of visual comparisons of the competing methods on each large-FoV test set, namely: Matterport3D [5], Pano3D-GV2 [2], Scannet++ [56], and KITTI360 [27], as shown in Fig. 8, 9, 10. Compared to Fig. 6, visual results of Unidepth [33] are also included for comparison.

Through visual comparisons, our DAC framework demonstrates sharper boundaries in the depth maps and more visually consistent scale in the depth visualization results. As seen in the A.Rel maps wrt. the ground-truth depth, our framework exhibits a significant advantage over each previous state-of-the-art method.

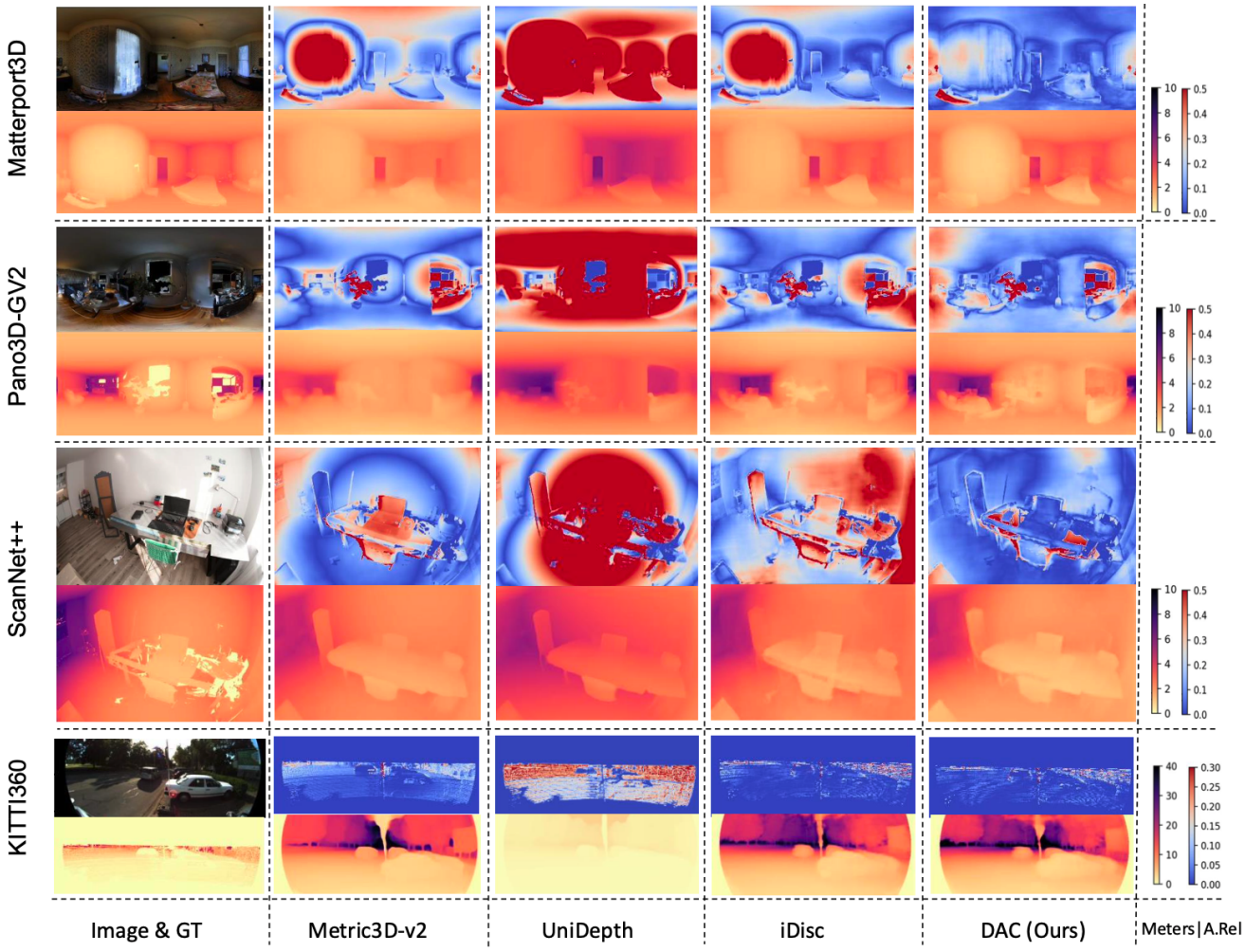

Figure 8. **Zero-Shot Qualitative Results.** For each dataset, an example is presented in two consecutive rows. The left column shows the original image and Ground-Truth depth map, followed by results from various methods. For each method, the top row displays the A.Rel map ↓ and the bottom row shows the predicted depth map. The color range for depth and A.Rel maps is indicated in the last column.

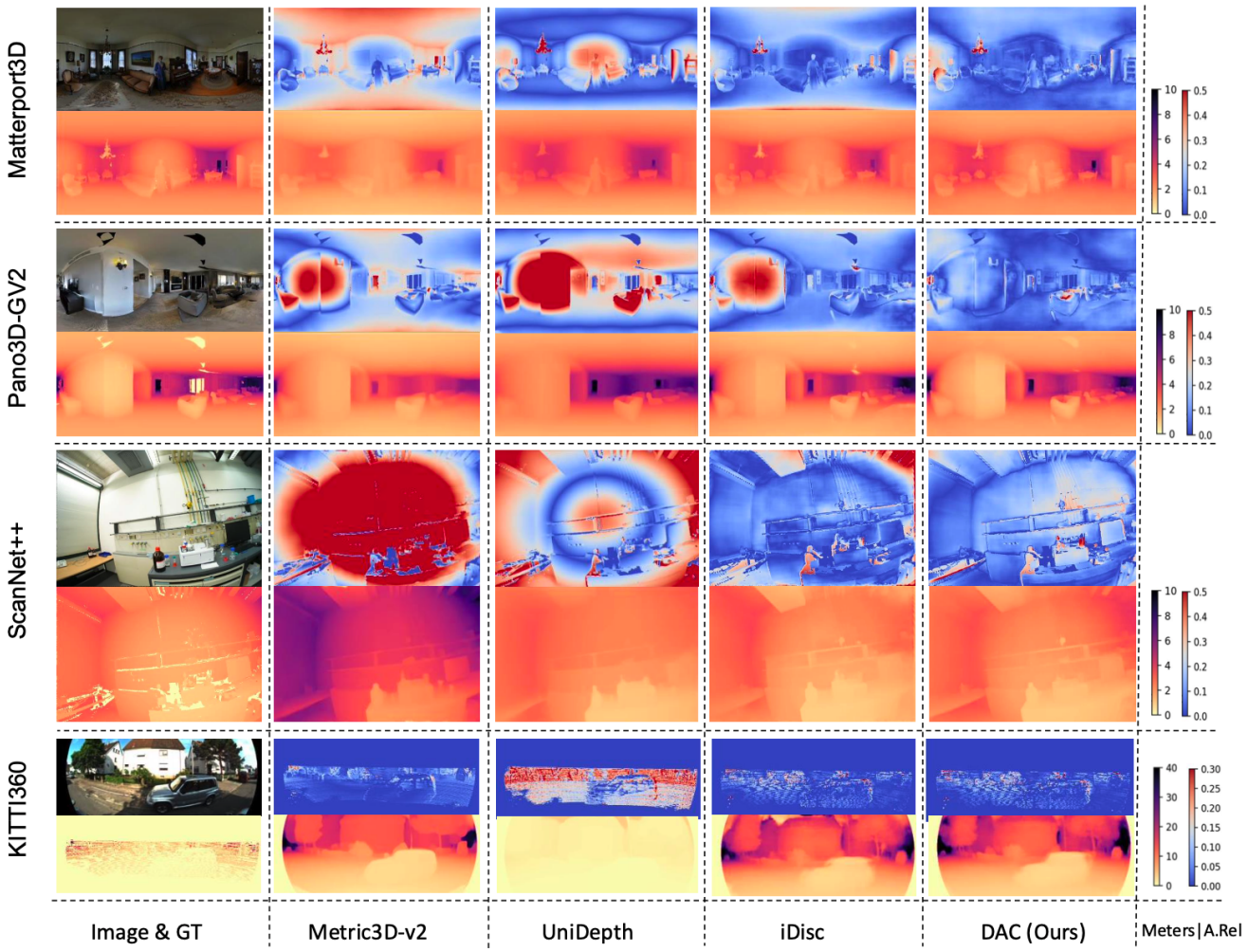

Figure 9. **Zero-Shot Qualitative Results.** For each dataset, an example is presented in two consecutive rows. The left column shows the original image and Ground-Truth depth map, followed by results from various methods. For each method, the top row displays the A.Rel map ↓ and the bottom row shows the predicted depth map. The color range for depth and A.Rel maps is indicated in the last column.

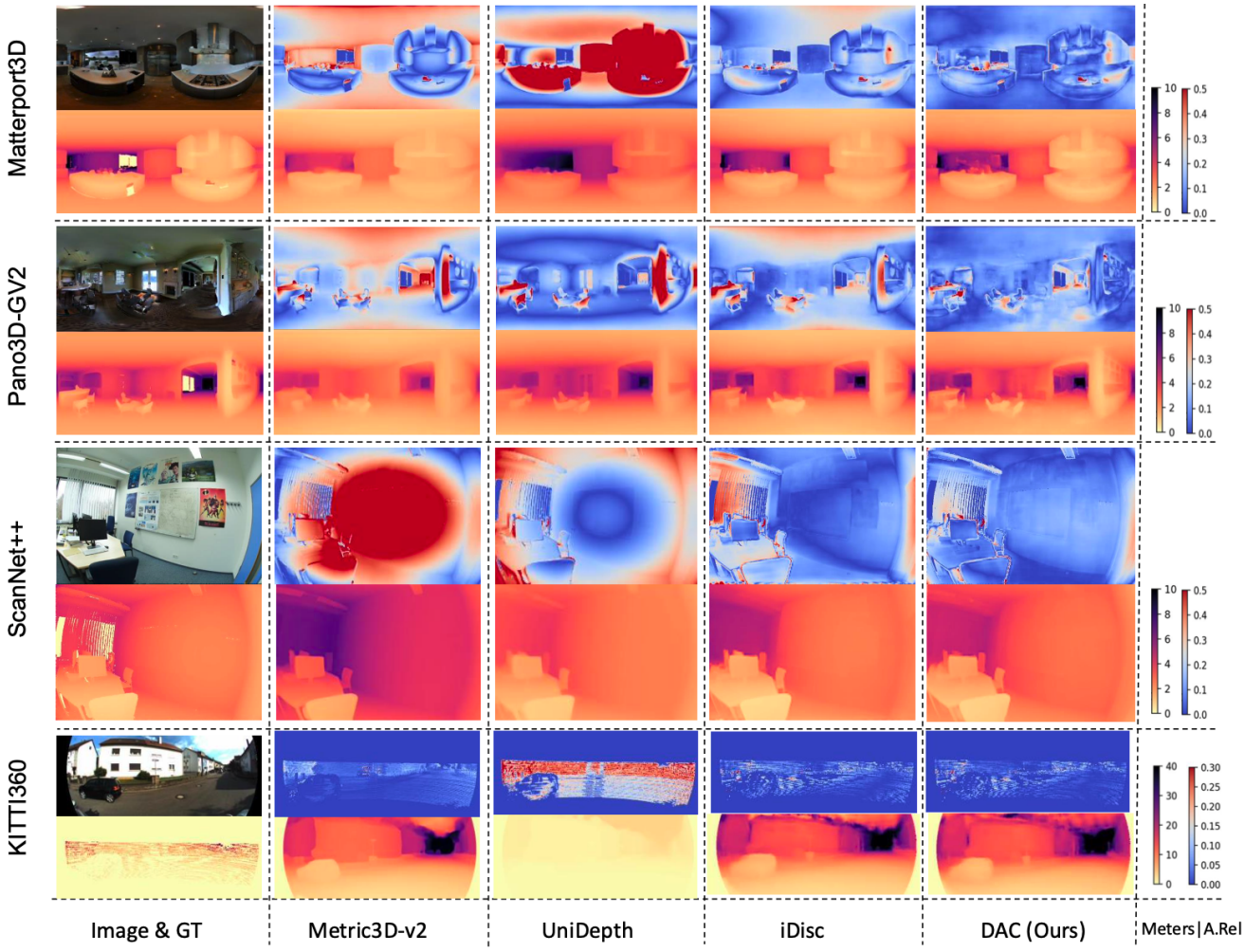

Figure 10. **Zero-Shot Qualitative Results.** For each dataset, an example is presented in two consecutive rows. The left column shows the original image and Ground-Truth depth map, followed by results from various methods. For each method, the top row displays the A.Rel map ↓ and the bottom row shows the predicted depth map. The color range for depth and A.Rel maps is indicated in the last column.