ZipDepth: Bringing Lightweight Zero-Shot Monocular Depth Anywhere, on Any Device

Supplementary Material

Fabio Tosi Luca Bartolomei Matteo Poggi Stefano Mattoccia
University of Bologna, Bologna, Italy

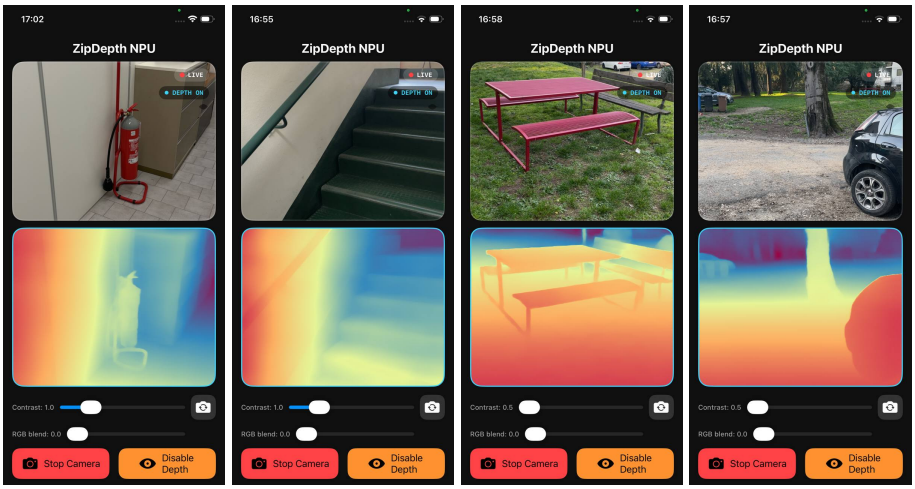

Fig. I: Qualitative results – predictions in-the-wild on iPhone 12 ANE. We report four screenshots from a prototype iOS app running ZipDepth in real-time.

This document supplements the ECCV 2026 paper *"ZipDepth: Bringing Lightweight Zero-Shot Monocular Depth Anywhere, on Any Device"*. It is organized as follows. Sec. 1 provides a layer-by-layer specification of ZipDepth. Sec. 2 details the deployment configurations used in our efficiency evaluation and reports per-backend latency and throughput across diverse platforms. Sec. 3 analyzes throughput and peak-memory scaling with input resolution for common square and non-square aspect ratios. Sec. 4 describes the full training set, spanning 17 datasets and over 14M images, together with the domain-balanced sampling strategy used to address severe domain imbalance. Sec. 5 presents qualitative results on the DA-2K benchmark, demonstrating generalization across six challenging scene categories including adverse weather, aerial views, transparent surfaces, and underwater imagery. Sec. 6 provides a qualitative comparison against MiDaS Small, DPT-Hybrid, DPT-Large, and the teacher Depth Anything v2-Large. Finally, Sec. 7 discusses failure cases by comparing ZipDepth predictions with those of Depth Anything v2-Large, highlighting the trade-off between model compactness and fine-detail reconstruction in complex scenes.

1 Detailed Architecture

Tab. I provides a layer-by-layer specification of our ZipDepth network. All convolutional layers use zero-padding to preserve spatial dimensions unless a stride is specified. *RepVGG* denotes the reparameterizable block, which maintains three parallel branches during training and is fused into a single 3×3 convolution at inference. *DWConv* denotes depthwise convolution. Channel counts after reparameterization and Conv-BN fusion are reported in the *Out Ch.* column.

Table I: Detailed ZipDepth architecture. Each row specifies one functional layer or module. S: stride; K: kernel size; G: groups; r: dilation rate. Output spatial dimensions assume an input of $H\times W$.

Stage	Layer / Module	K	S	Out Ch.	Resolution	Notes
Encoder						
Stem	ConvBN	3	2	24	$H/2$	Skip $\mathbf{s}_{1/2}$ to decoder
	ConvBN	3	2	48	$H/4$	Enters Stage 1
Stage 1	RepVGG $\times2$	3	1	48	$H/4$	
Stage 2	RepVGG (downsample)	3	2	96	$H/8$	Stride-2 transition
	RepVGG $\times2$	3	1	96	$H/8$	
	MinimalMultiScale	3	1	96	$H/8$	DWConv $r=1$ + DWConv $r=2$
	StripPoolingAttn	1	–	96	$H/8$	Sigmoid gate, DW 1×1
Stage 3	RepVGG (downsample)	3	2	192	$H/16$	Stride-2 transition
	RepVGG $\times6$	3	1	192	$H/16$	
	ChannelAttention (SE)	–	–	192	$H/16$	Reduction = 8; hidden = 24
	GlobalContextBlock	1	–	192	$H/16$	Reduction = 4; hidden = 48
Stage 4	RepVGG (downsample)	3	2	384	$H/32$	Stride-2 transition
	RepVGG $\times2$	3	1	384	$H/32$	
Post-enc.	SPPF	5	1	384	$H/32$	Hidden = 96; 3$\times$ cascaded MaxPool
	CrossScale	1	–	192, 384	$H/16$, $H/32$	Grouped 1×1; scale = 0.3
Decoder						
FPN Fusion	Proj_4 (ConvBN)	1	1	288	$H/32$	Project Stage 4
	Fuse_3	1	–	192	$H/16$	Stage 3 skip + $\uparrow\text{Proj}_4$
	Fuse_2	1	–	144	$H/8$	Stage 2 skip + $\uparrow\text{Fuse}_3$
	Fuse_1	1	–	96	$H/4$	Stage 1 skip + $\uparrow\text{Fuse}_2$
Half-res head	$\text{Fuse}_{1/2}$	1	–	32	$H/2$	$\mathbf{s}_{1/2}$ skip + $\uparrow\text{Fuse}_1$
	DepthHead	3	1	1	$H/2$	Intermediate depth $\hat{d}_{1/2}$
	ConvexUp (GPU)	3	–	1	H	2$\times$; 9-neighbour softmax
	GatedUp (NPU)	5	–	1	H	2$\times$; gate 1×1,DW5×5,1×1

2 Deployment Profiling

Tab. II reports ZipDepth latency and throughput across different hardware platforms. Measurements are performed at 384×384 resolution with batch size 1, and latency is reported as the median over 200 forward passes after 20 warm-up steps. This evaluation characterizes ZipDepth deployment efficiency across practical inference stacks, including desktop GPUs, embedded accelerators, CPUs, and mobile NPUs. We briefly describe each configuration below.

– **PyTorch Eager.** The default execution mode with no graph-level optimization. It serves as the unoptimized baseline for all speedup ratios.

– **PyTorch Fused.** Applies two lossless algebraic transformations: (i) Conv-BN fusion, which absorbs batch normalization parameters into the preceding convolution weights and biases, eliminating the normalization pass entirely; and (ii) reparameterization fusion, which collapses the three parallel branches of each RepVGG block (3×3, 1×1, and identity) into a single 3×3 convolution. Both transformations are exact and introduce no numerical error.

– `torch.compile`**.** Applies PyTorch 2's graph-mode compilation with the `inductor` backend, enabling operator fusion, memory planning, and kernel auto-tuning beyond what manual fusion provides.

– **TensorRT.** Builds a statically shaped FP16 inference engine from an exported ONNX graph. TensorRT performs layer fusion, kernel selection, and precision calibration specific to the target GPU architecture. Since ZipDepth uses only natively supported operators, export requires no graph surgery or custom plugins.

– **PyTorch JIT Frozen.** Applies `torch.jit.trace` followed by `torch.jit.freeze`, which inlines constants, folds batch normalization, and removes training-only subgraphs. This is the strongest PyTorch-native optimization available on CPU platforms where `torch.compile` support is limited.

– **ONNX Runtime (ORT).** Executes an exported ONNX graph with backend-specific optimizations. On CPU, ORT applies graph-level passes (constant folding, node fusion) and dispatches to optimized kernels. Results are reported for the best-performing thread count. The NPU/mobile path variant (◊) replaces the `unfold`-based convex upsampling with the Conv2D-only alternative (Sec. 3.2 of the main paper), which avoids operators that lack efficient CPU implementations.

– **CoreML.** Targets Apple hardware via the Core ML framework, with execution on the Apple Neural Engine (ANE).

– **TFLite GPU.** Targets mobile GPUs via TensorFlow Lite's GPU delegate, operating in FP16 precision. This backend is representative of Android deployment without dedicated NPU support.

Table II: Deployment profiling of ZipDepth at 384×384 resolution. Latency is the median over 200 forward passes (20 warm-up). ∗ Conv-BN and reparameterizable branch fusion applied via `fuse_for_inference()`. ‡ TensorRT FP16 engine with static shape. † `torch.compile` with `mode="max-autotune"`. ▲ ONNX Runtime; best thread count reported. ♢ NPU/mobile path. ♣ CoreML with Neural Engine.

Device	Backend	Prec.	Latency (ms)	FPS	Speedup (vs. Eager)
NVIDIA RTX 3090 (24 GB, 350 W)	PyTorch Eager	FP32	3.9	255	1.0×
	PyTorch Fused*	FP32	2.5	389	1.5×
	PyTorch Fused*	FP16	3.2	307	1.2×
	`torch.compile`†	FP16	0.9	1117	4.4×
	TensorRT‡	FP16	**0.8**	**1317**	**5.1**×
NVIDIA RTX 3070 Laptop (140 W)	PyTorch Eager	FP32	3.9	249	1.0×
	PyTorch Fused*	FP32	2.7	351	1.4×
	PyTorch Fused*	FP16	2.9	341	1.4×
	`torch.compile`†	FP16	2.0	492	2.0×
	TensorRT‡	FP16	**1.3**	**773**	**3.1**×
NVIDIA Jetson Orin NX 16 GB (MAXN, 50 W)	PyTorch Eager	FP32	13.9	72	1.0×
	PyTorch Fused*	FP32	8.5	118	1.6×
	PyTorch Fused*	FP16	8.8	114	1.6×
	TensorRT‡	FP16	**3.2**	**196**	**4.3**×
NVIDIA Jetson Orin NX 16 GB (15 W)	PyTorch Eager	FP32	32.0	31	1.0×
	PyTorch Fused*	FP32	29.0	34	1.4×
	PyTorch Fused*	FP16	23.7	42	1.4×
	TensorRT‡	FP16	**13.1**	**77**	**2.9**×
AMD EPYC 7443 (24-core, 200 W)	PyTorch Fused (CPU)*	FP32	19.8	50	1.0×
	PyTorch JIT Frozen (CPU)	FP32	18.9	53	1.1×
	ORT CPU▲	FP32	32.5	30	0.6×
	ORT CPU▲♢	FP32	**13.4**	**75**	**1.5**×
Intel Core i7-11800H (8-core, 45 W)	PyTorch Fused (CPU)*	FP32	25.14	39.9	1.0×
	PyTorch JIT Frozen (CPU)	FP32	20.5	48.9	1.2×
	ORT CPU▲	FP32	38.7	25.6	0.6×
	ORT CPU▲♢	FP32	**22.2**	**44.6**	**1.1**×
Apple iPad Pro M4 (∼10 W)	CoreML ANE♣	FP32	5.6	180	1.0×
	CoreML ANE♣	FP16	2.8	360	2.0×
	CoreML ANE♣♢	FP16	**1.4**	**715**	**4.0**×
Apple iPhone 12 (∼5 W)	CoreML ANE♣	FP32	25	40	1.0×
	CoreML ANE♣	FP16	4.2	240	6.0×
	CoreML ANE♣♢	FP16	**2.7**	**375**	**9.4**×
Xiaomi Poco X3 NFC (∼5 W)	TFLite GPU	FP32	96.1	10	1.0×
	TFLite GPU	FP16	**62.5**	**16**	**1.5**×

3 Resolution Scaling Analysis

The main paper reports all efficiency metrics at a fixed 384×384 input resolution. In practice, deployment scenarios often require processing images at their native resolution or at non-square aspect ratios (*e.g.* 4:3 or 16:9). Tab. III evaluates how ZipDepth throughput scales across a range of resolutions on different platforms. All measurements use PyTorch with Conv-BN and reparameterization fusion (no graph-level or backend-specific optimization), FP32 precision, batch size 1. Latency is the median over 200 forward passes after 50 warm-up iterations.

Table III: Throughput vs. input resolution. PyTorch Fused, FP32. ♣ NPU/mobile path. Resolutions include common square and non-square aspect ratios.

Device	Input Resolution (FPS)						
	256^2	384^2	384×512	512^2	384×672	512×768	768×1024
RTX 3090 - GPU	401	389	380	365	346	302	192
RTX 3070 - GPU	476	368	298	222	219	156	79
Orin NX - GPU (50 W)	141	118	116	108	110	94	50
Orin NX - GPU (15 W)	56	34	30	24	24	17	9
Intel i7-11800H - CPU	72	41	27	20	22	15	7
iPhone 12 A14	44	40	37	30	28	20	10
iPhone 12 A14 (FP16)	490	240	195	150	145	95	43
iPhone 12 A14 (FP16)♣	675	375	315	245	235	160	75

High-resolution stress test. Extending the analysis in Tab. III beyond 768×1024, Tab. IV reports throughput up to 4K across five platforms: the RTX 3090 and RTX 3070 as desktop GPU references, the Jetson Orin NX at both 50 W and 15 W power modes as embedded targets, and the iPhone 12 via CoreML on the Apple Neural Engine as a representative mobile device. Measurements use PyTorch Fused FP32; iPhone measurements use CoreML FP16 with the NPU/mobile upsampling path. Note that these experiments measure *throughput scaling only* — the model is trained at low resolution, and no claim is made about depth estimation quality at the resolutions reported below.

Peak memory across resolutions. Tab. V complements this analysis with peak GPU memory on the RTX 3090, across Eager and Conv-BN/reparameterization-fused models in FP32 and FP16. Weight memory is constant across resolutions, as expected, while activation memory grows with input size but *sub-linearly* relative to pixel count: the activation-to-megapixel ratio decreases from ∼545 MB/Mpx at 384×384 to ∼308 MB/Mpx at 4K, suggesting the presence of a resolution-independent memory component (*e.g.* fixed-size buffers and intermediate tensors whose footprint does not scale proportionally with input area) which becomes increasingly amortized at larger resolutions. This contrasts with the near-linear GMACs scaling observed in Tab. IV, confirming that compute

and memory scale differently with resolution and should be considered jointly when assessing deployment feasibility. Fusion has negligible effect on peak memory (within ∼2% of the unfused model at every resolution), while FP16 consistently reduces peak memory by ∼25–32% relative to FP32, regardless of resolution. Although FP16 halves parameter storage, the reduction in peak memory is substantially smaller, indicating that a significant fraction of the memory footprint arises from runtime workspaces and other allocations that do not scale proportionally with tensor precision.

Table IV: High-resolution throughput. ♣ CoreML ANE, NPU/mobile path.

Resolution	Mpx	GMACs	RTX 3090	RTX 3070	Orin NX (50 W)	Orin NX (15 W)	iPhone 12♣
768×1024	0.79	16.06	192	79	50	9	75
1024^2	1.05	21.41	149	61	38	7	50
1080×1920	2.07	42.53	78	31	18	3	20
1440×2560	3.69	75.28	47	17	10	2	12
2160×3840	8.29	169.61	21	6	5	0.8	OOM

Table V: Peak memory across resolutions (NVIDIA RTX 3090, PyTorch backend). *Weights* are constant across resolutions; *Activations* denotes peak intermediate-tensor memory during the forward pass; *Peak* is total memory allocated (weights + activations). TensorRT and CoreML manage memory independently via their own runtimes and are not directly comparable to these figures.

Resolution	Eager FP32			Fused FP32			Eager FP16			Fused FP16		
	Wt.	Act.	Peak	Wt.	Act.	Peak	Wt.	Act.	Peak	Wt.	Act.	Peak
384×384	25.9	80.1	106.0	23.4	80.9	104.3	12.9	66.5	79.4	11.7	65.8	77.5
768×1024	25.9	275.2	301.1	23.4	275.4	298.8	12.9	166.0	179.0	11.7	165.5	177.2
1024^2	25.9	352.9	378.8	23.4	353.8	377.2	12.9	256.8	269.7	11.7	256.7	268.4
1080×1920	25.9	666.6	692.5	23.4	667.1	690.5	12.9	474.6	487.6	11.7	475.2	487.0
1440×2560	25.9	1153.8	1179.7	23.4	1154.5	1178.0	12.9	817.7	830.6	11.7	817.3	829.1
2160×3840	25.9	2550.9	2576.8	23.4	2552.2	2575.6	12.9	1791.3	1804.3	11.7	1790.2	1801.9

4 Training Data

Tab. VI provides a per-domain breakdown of the training set used in all experiments. The full collection spans 17 datasets and approximately 14.1M images, covering indoor scenes, outdoor driving, street-level imagery, web-crawled photos, landmarks, and synthetic/transparent objects. For some of the largest sources (*e.g.* SA-1B [4], OpenImages v7 [5]), we use subsets to keep the overall pool tractable while preserving domain diversity.

Depth pseudo-labels for all images are generated offline using Depth Anything v2-Large [18]. To mitigate the heavy imbalance in per-domain sizes, we apply a temperature-scaled domain-balanced sampler: sampling probabilities are proportional to inverse-frequency weights raised to a temperature $T = 0.2$, then bidirectionally clipped to enforce a minimum coverage of 25% and a maximum repeat factor of 30× per domain within each epoch.

Table VI: Training datasets. Per-domain image counts and scene categories.

Dataset	Domain / Scene Type	Images
SA-1B [4]	Web-crawled, diverse	6 387 613
Google Landmarks [14]	Landmarks, architecture	3 358 709
OpenImages v7 [5]	Web-crawled, object-centric	1 743 042
Object365 [12]	Object-centric, indoor/outdoor	1 693 306
COCO [7]	Indoor/outdoor, multi-category	532 855
DrivingStereo [17]	Driving, stereo-captured	169 894
MegaDepth [6]	SfM reconstructions, tourism	128 315
BDD100K [19]	Driving, diverse weather	100 000
HoloPix50K [3]	Stereo, user-generated	49 124
ADE20K [20]	Indoor/outdoor, semantic scenes	27 574
HRWSI [15]	Stereo, high-resolution web	20 778
Mapillary [8]	Street-level, global	20 000
Gated2Depth [2]	Night/adverse driving	14 277
Trans10K [16]	Transparent objects	10 428
ACDC [11]	Adverse conditions driving	8 012
Cityscapes [1]	Urban driving	5 000
Flickr1024 [13]	Stereo, diverse	1 024
Total		**14 269 951**

5 Qualitative Results on DA-2K

We present qualitative depth predictions of ZipDepth on representative images from the DA-2K benchmark [18], covering six challenging scene categories. For each category, the first row shows the input RGB images and the second row shows the corresponding predicted depth maps. No ground-truth depth is available for DA-2K. The results highlight the ability of ZipDepth to generalize well beyond standard scenes: despite its compact design, the model produces coherent depth estimates even under challenging conditions such as adverse weather, aerial viewpoints, transparent and reflective surfaces, and underwater imagery. We attribute this robustness to two complementary factors: the representational capacity of the knowledge distillation teacher, Depth Anything v2-Large [18], and the diversity of our training set, which explicitly covers adverse-condition driving, transparent objects, and a broad spectrum of web-crawled and domain-specific imagery spanning over 14M images across 17 domains.

6 Qualitative Comparison with Competing Methods

Fig. III provides a qualitative comparison of ZipDepth against representative monocular depth estimation models of varying capacity: MiDaS Small [10], DPT-Hybrid and DPT-Large [9], and the knowledge distillation teacher Depth Anything v2-Large [18]. The comparison spans four diverse scenes and illustrates the trade-off between model size and depth quality. Despite its significantly smaller footprint, ZipDepth produces depth maps that are qualitatively closer to the teacher and to the large-capacity DPT models than to MiDaS Small, retaining coherent scene structure and reasonable boundary sharpness.

7 Limitations

Despite its strong generalization, ZipDepth inevitably trades accuracy for efficiency. Its compact architecture, while enabling real-time inference on embedded and mobile hardware, limits the representational capacity available to handle highly complex or ambiguous scenes: fine-grained depth boundaries, highly repetitive textures, thin structures, and strong reflections all require a level of detail that a 6.1M-parameter model may fail to capture. Fig. IV illustrates representative failure cases, showing the predictions of ZipDepth alongside those of its teacher, Depth Anything v2-Large [18]. The gap highlights where model compactness becomes the binding constraint: regions that the teacher resolves correctly are over-smoothed or incorrectly ordered by ZipDepth, reflecting the inherent cost of aggressive model compression. Note that, as a distillation-based approach, ZipDepth is additionally bounded by the quality of the pseudo-labels — though in practice this represents a secondary limitation compared to the capacity gap.

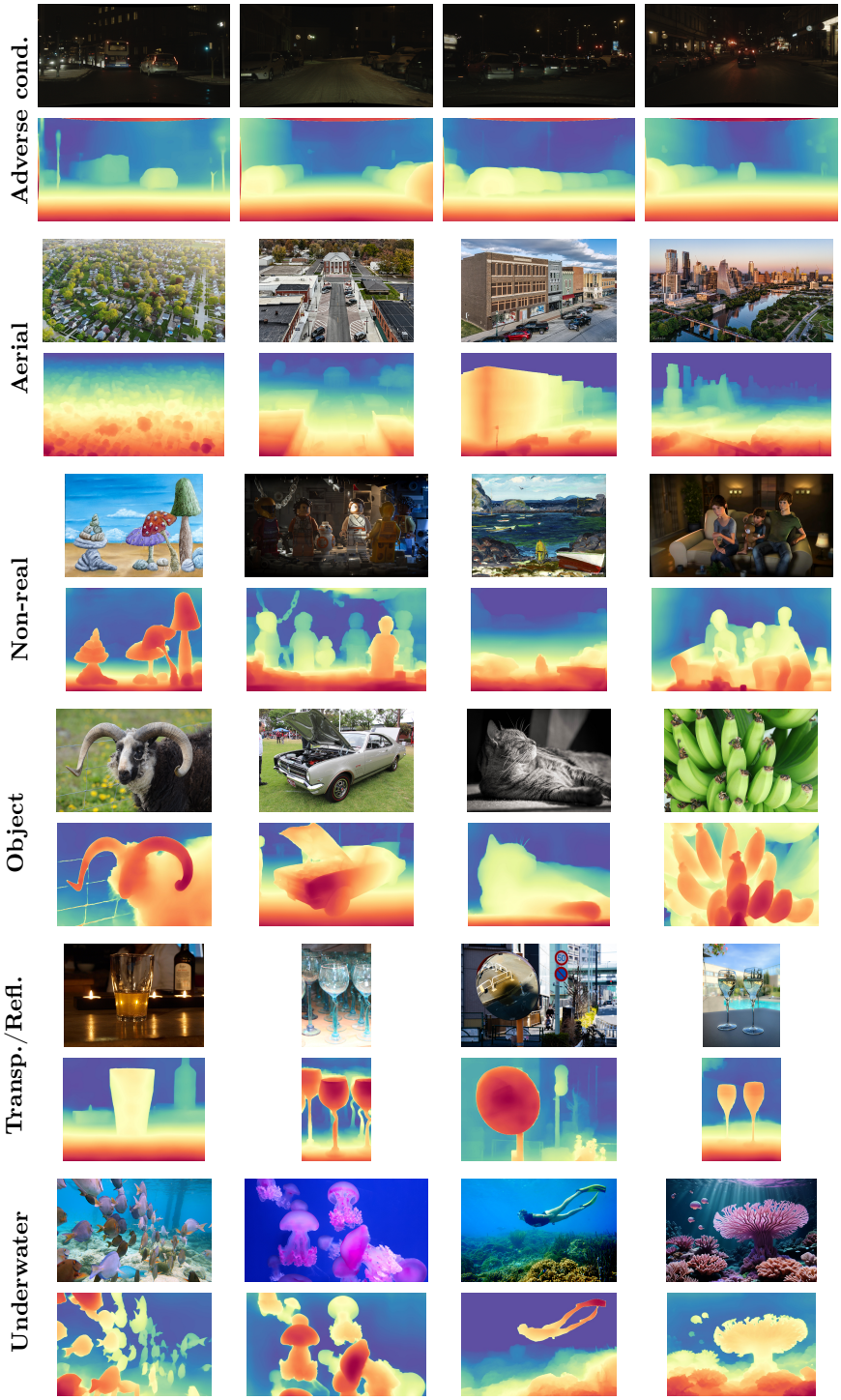

Fig. II: Qualitative predictions of ZipDepth on DA-2K [18]. Each category occupies two rows: input RGB images (*top*) and the corresponding ZipDepth depth maps (*bottom*). No ground-truth depth is available for DA-2K. Despite being a compact 6.1M-parameter model trained via knowledge distillation from Depth Anything v2-Large [18] (335M parameters), ZipDepth produces spatially coherent and semantically aware depth estimates across all six categories, including challenging cases such as transparent/reflective surfaces, non-photorealistic imagery, adverse weather, and underwater scenes.

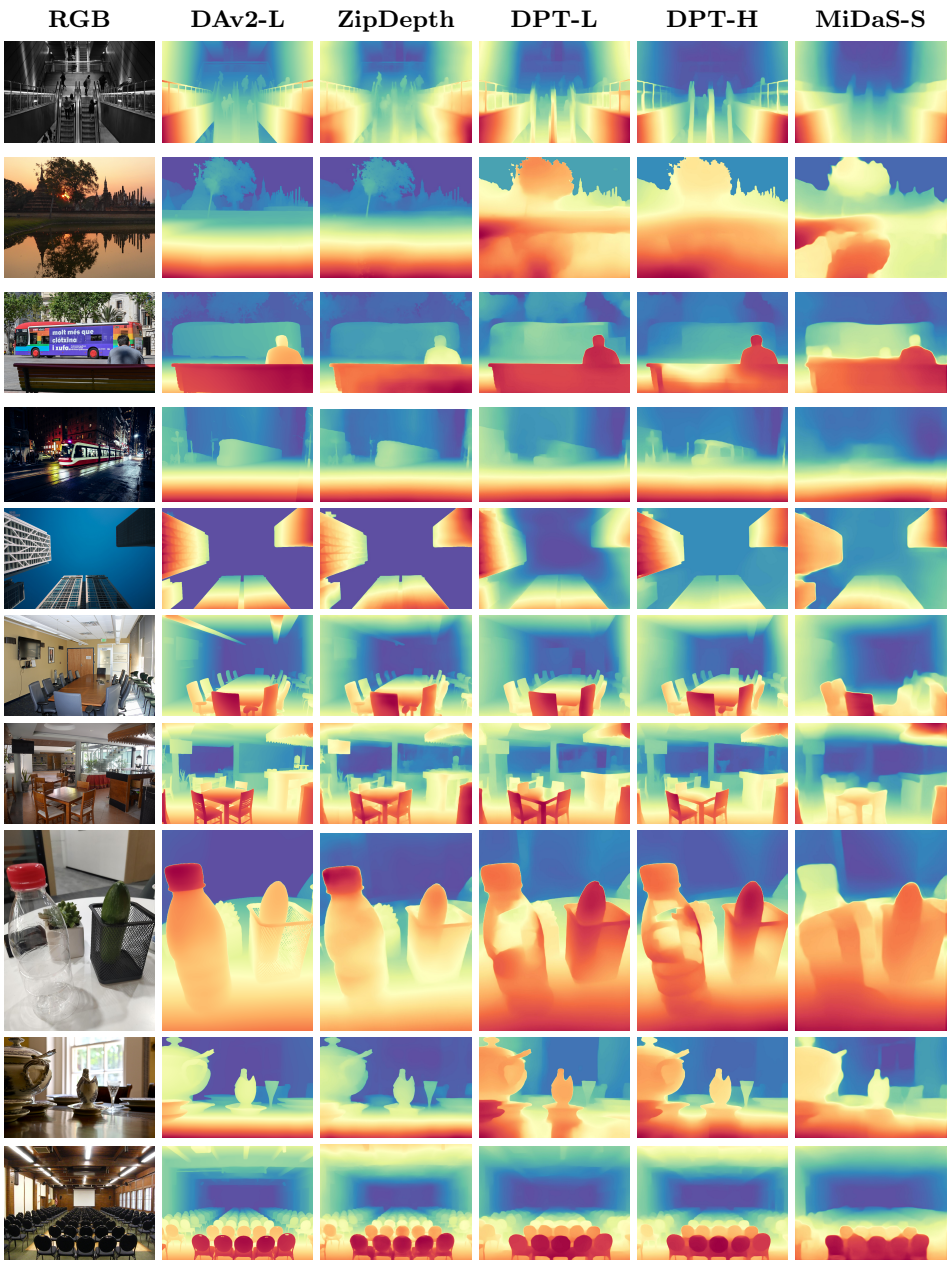

Fig. III: Qualitative comparison of monocular depth estimation methods. From left to right: input RGB, Depth Anything v2-Large [18] (teacher), ZipDepth (ours), DPT-Large [9], DPT-Hybrid [9], and MiDaS Small [10]. Despite its compact 6.1M-parameter design, ZipDepth produces depth maps that closely follow the teacher and compare favourably against larger models, while running at real-time speed on a wide range of devices.

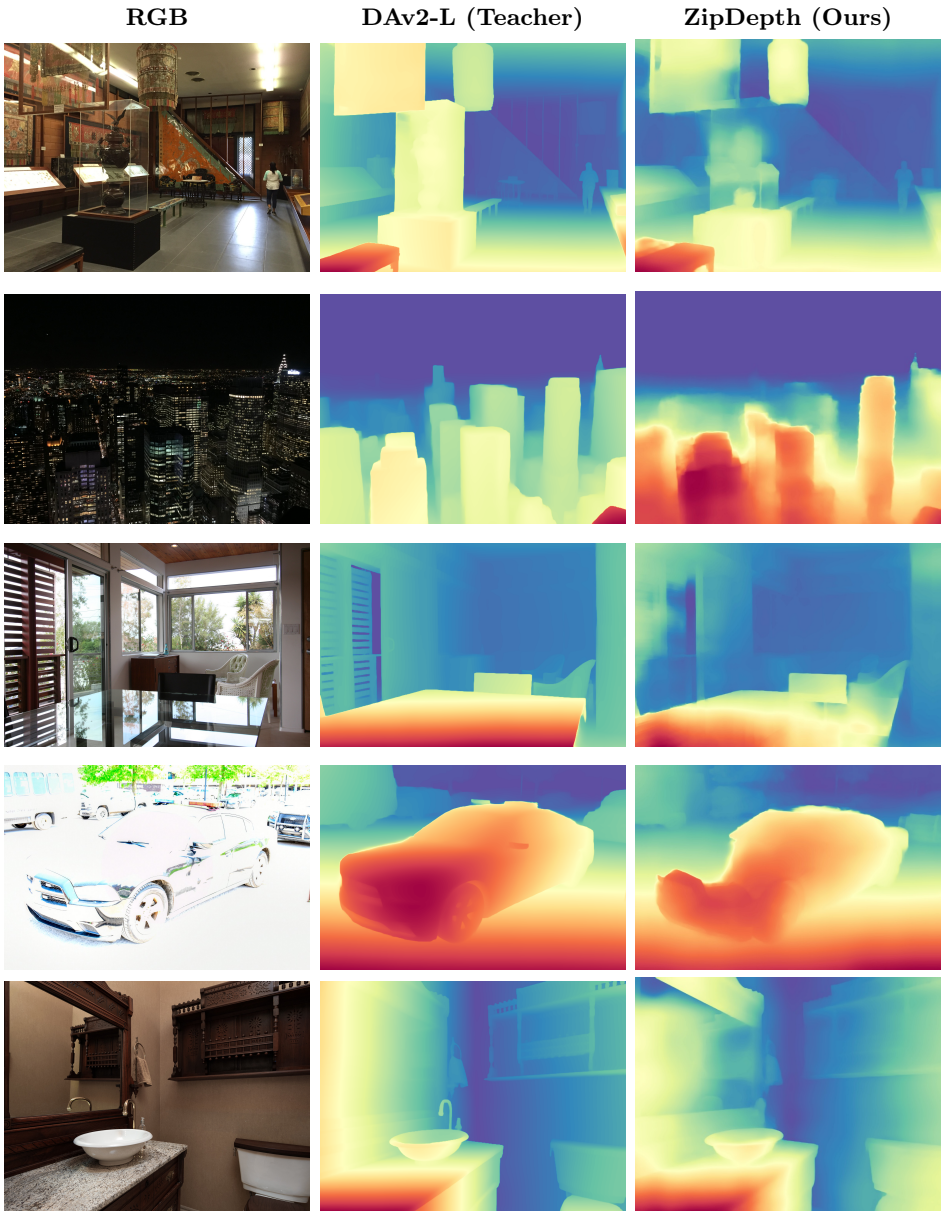

Fig. IV: Failure cases of ZipDepth compared to its teacher. Each row shows an input RGB image, the depth maps produced by Depth Anything v2-Large [18] (teacher), and the corresponding ZipDepth prediction. ZipDepth tends to over-smooth fine depth boundaries and lose structural detail in textureless or reflective regions, reflecting the capacity gap with respect to the teacher.

References

1. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016) 7
2. Gruber, T., Julca-Aguilar, F., Bijelic, M., Heide, F.: Gated2depth: Real-time dense lidar from gated images. In: The IEEE International Conference on Computer Vision (ICCV) (2019) 7
3. Hua, Y., Kohli, P., Uplavikar, P., Ravi, A., Gunaseelan, S., Orozco, J., Li, E.: Holopix50k: A large-scale in-the-wild stereo image dataset. arXiv preprint arXiv:2003.11172 (2020) 7
4. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4015–4026 (2023) 7
5. Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Kolesnikov, A., Duerig, T., Ferrari, V.: The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. IJCV (2020) 7
6. Li, Z., Snavely, N.: Megadepth: Learning single-view depth prediction from internet photos. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2041–2050 (2018) 7
7. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014) 7
8. Neuhold, G., Ollmann, T., Rota Bulo, S., Kontschieder, P.: The mapillary vistas dataset for semantic understanding of street scenes. In: Proceedings of the IEEE international conference on computer vision. pp. 4990–4999 (2017) 7
9. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 12179–12188 (2021) 8, 10
10. Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., Koltun, V.: Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. IEEE transactions on pattern analysis and machine intelligence **44**(3), 1623–1637 (2020) 8, 10
11. Sakaridis, C., Dai, D., Van Gool, L.: Acdc: The adverse conditions dataset with correspondences for semantic driving scene understanding. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 10765–10775 (2021) 7
12. Shao, S., Li, Z., Zhang, T., Peng, C., Yu, G., Zhang, X., Li, J., Sun, J.: Objects365: A large-scale, high-quality dataset for object detection. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 8430–8439 (2019) 7
13. Wang, Y., Wang, L., Yang, J., An, W., Guo, Y.: Flickr1024: A large-scale dataset for stereo image super-resolution. In: Proceedings of the IEEE/CVF international conference on computer vision workshops. pp. 0–0 (2019) 7
14. Weyand, T., Araujo, A., Cao, B., Sim, J.: Google Landmarks Dataset v2 - A Large-Scale Benchmark for Instance-Level Recognition and Retrieval. In: Proc. CVPR (2020) 7

15. Xian, K., Zhang, J., Wang, O., Mai, L., Lin, Z., Cao, Z.: Structure-guided ranking loss for single image depth prediction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 611–620 (2020) 7
16. Xie, E., Wang, W., Wang, W., Ding, M., Shen, C., Luo, P.: Segmenting transparent objects in the wild. arXiv preprint arXiv:2003.13948 (2020) 7
17. Yang, G., Song, X., Huang, C., Deng, Z., Shi, J., Zhou, B.: Drivingstereo: A large-scale dataset for stereo matching in autonomous driving scenarios. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 899–908 (2019) 7
18. Yang, L., Kang, B., Huang, Z., Zhao, Z., Xu, X., Feng, J., Zhao, H.: Depth anything v2. Advances in Neural Information Processing Systems **37**, 21875–21911 (2024) 7, 8, 9, 10, 11
19. Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., Darrell, T.: Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2636–2645 (2020) 7
20. Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., Torralba, A.: Semantic understanding of scenes through the ade20k dataset. International Journal of Computer Vision **127**(3), 302–321 (2019) 7