Towards Zero-Shot Scale-Aware Monocular Depth Estimation – Supplementary Material –

Vitor Guizilini Igor Vasiljevic Dian Chen Rareș Ambruș Adrien Gaidon
Toyota Research Institute (TRI), Los Altos, CA

1. Training Details

We implemented our models using PyTorch [8], with distributed training across 8 A100 GPUs and TensorFloat-32 precision format. We use the AdamW optimizer [7], with standard parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, a weight decay of $w = 10^{-4}$, batch size of $b = 16$, and an initial learning rate of $lr_1 = 10^{-4}$. During the first epoch, we linearly warm the learning rate up from $lr_0 = 10^{-5}$. Afterwards, we decay the learning rate by a factor of $\gamma = 0.8$ after every 5 epochs for outdoor experiments, and 2 epochs for indoor experiments, such that $lr_{n+1} = \gamma lr_n$. In addition to our proposed encoder-level data augmentation techniques, we also apply random horizontal flipping with 50% probability, and color jittering of $(0.5, 0.5, 0.5, 0.1)$ respectively for brightness, contrast, saturation and hue.

For resolution jittering, we randomly resize input images to resolutions between 25% and 150% of the original $H \times W$, independently for the height and width dimensions. Due to network architecture restrictions, we round up our sampled resolutions to be multiples of 32. For embedding dropout, we randomly select a number of encoder embeddings between 0% and 50% to remove at each training iteration. During evaluation we do not perform any sort of data augmentation. For the loss calculation, we multiply the surface normal regularization term by $\alpha_N = 0.2$, and the KL-divergence term by $\alpha_{KL} = 0.1$. To decrease memory requirements and computational complexity, during training we use strided ray sampling [6] to downsample the decoded image to $1/8$ the original resolution.

2. Network Architecture

We use a ResNet18 [4] backbone as the encoder to generate 960-dimensional image embeddings. Our geometric embeddings are calculated using $F = 16$ frequency bands and $\mu = 64$ as the maximum resolution, resulting in 51-dimensional vectors. Our latent representation is of dimensionality 1024×1024, with 8 self-attention heads and 8 self-attention layers for conditioning, including GeLU activations [5] and dropout of 0.1. We use a single cross-attention layer for conditioning, and another single cross-attention layer for decoding, followed by an MLP that projects the output to a 1-dimensional depth estimate. For uncertainty estimation, we decode 10 depth maps, from different sampled latent representations, and calculate the pixel-level mean μ_{ij} and standard deviations σ_{ij}. In total, ZeroDepth has $232,591,380$ parameters.

3. Extended Depth Estimation Tables

For completeness, in Tables 1 and 2 we provide depth estimation results for each individual camera of the *DDAD* and *nuScenes* datasets. These results are obtained using the outdoor variant of ZeroDepth, and were averaged to generate our entries in Tables 1 and 2 of the main paper. Moreover, in Table 3 we report the full depth estimation results of our ablation regarding the use of different training datasets (see Figure 6 of the main paper, where due to space constraints we only report *KITTI* results). In these results we observe a similar trend: performance consistently degrades across all evaluation datasets as we consider fewer training datasets, and the degradation is similar between metric and median-scaled predictions.

In particular, improvements seem to be correlated with the number of training tokens available on each dataset: considering 384×640 resolution images, and an encoding downsample ratio of 4 (Section 3.3, main paper), each image contains a total of 15360 tokens. Therefore, the *PD* dataset has roughly 8.5B tokens, followed by *TartanAir* with 9.4B, *Waymo* with 1.5T, and *LSD* with 1.6T tokens. Note that this is without considering our proposed encoder-level data augmentation techniques (Section 3.5, main paper), that further increases training token diversity by (i) modifying the CNN features used as image embeddings; and (ii) perturbing the geometric embeddings to cover the entire camera field of view. Increasing the number of training tokens by ingesting additional datasets, as well as increasing network complexity to enable proper learning from such diverse data, are straightforward ways to further increase performance within our framework.

Method	Camera	Med.Scale	*Lower is better*				*Higher is better*		
			AbsRel	SqRel	RMSE	RMSE_{log}	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
ZeroDepth	Front	✗	0.100	1.916	11.214	0.188	0.895	0.962	0.983
	Front-Left		0.148	2.245	10.011	0.249	0.833	0.932	0.965
	Front-Right		0.182	2.934	10.397	0.286	0.771	0.908	0.951
	Back-Left		0.165	2.642	10.648	0.269	0.806	0.918	0.957
	Back-Right		0.205	3.268	10.484	0.309	0.748	0.893	0.969
	Back		0.157	2.656	12.135	0.248	0.813	0.933	0.969
ZeroDepth	Front	✓	0.100	1.950	11.318	0.191	0.889	0.961	0.982
	Front-Left		0.151	2.325	10.067	0.254	0.818	0.931	0.965
	Front-Right		0.179	3.113	10.874	0.308	0.760	0.893	0.941
	Back-Left		0.170	2.555	10.728	0.279	0.782	0.912	0.955
	Back-Right		0.206	3.053	10.591	0.332	0.714	0.875	0.930
	Back		0.159	2.806	12.627	0.265	0.808	0.917	0.962

Table 1: **Per-camera ZeroDepth depth estimation results** on the DDAD [2] dataset.

Method	Camera	Med.Scale	*Lower is better*				*Higher is better*		
			AbsRel	SqRel	RMSE	RMSE_{log}	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
ZeroDepth	Front	✗	0.150	2.101	7.484	0.240	0.839	0.939	0.969
	Front-Left		0.287	4.931	7.300	0.363	0.711	0.862	0.920
	Front-Right		0.420	12.247	7.545	0.391	0.690	0.853	0.913
	Back-Left		0.193	3.615	7.818	0.291	0.796	0.910	0.952
	Back-Right		0.252	2.970	6.411	0.340	0.709	0.866	0.924
	Back		0.226	2.516	6.669	0.331	0.732	0.881	0.932
ZeroDepth	Front	✓	0.157	2.154	7.612	0.239	0.822	0.941	0.971
	Front-Left		0.259	3.913	7.063	0.341	0.716	0.876	0.929
	Front-Right		0.354	6.899	7.043	0.365	0.690	0.851	0.920
	Back-Left		0.192	3.095	7.639	0.281	0.789	0.917	0.958
	Back-Right		0.230	2.728	6.275	0.321	0.735	0.878	0.930
	Back		0.223	2.609	6.693	0.317	0.731	0.883	0.935

Table 2: **Per-camera ZeroDepth depth estimation results** on the nuScenes [1] dataset.

4. Variational Uncertainty Sampling

In Figure 1 we show an example of predicted variational uncertainty, and how it can be used to improve depth estimation by selecting pixels with higher confidence levels. As expected (Figure 1a), uncertainty increases with longer ranges, and is also larger in areas with sudden depth discontinuities (i.e., object boundaries), that are usually smoothed out to generate a characteristic "bleeding" effect across modes. By removing as few as 10% of the valid depth pixels, we already observe a significant improvement of 30% in Root Mean Squared Error (RMSE), from 4.044 to 2.859, mostly due to the removal of areas with bleeding artifacts. In fact, the overall pointcloud structure (i.e., observed cars, ground plane and walls) is preserved even when we remove as much as 50% of valid depth pixels, leading to an RMSE improvement of 63% relative to the full pointcloud.

5. Full Surround Pointclouds

The *DDAD* and *nuScenes* datasets have multiple cameras in each sample, which enables the reconstruction of full surround pointclouds by combining reconstructions from each individual camera. This property has been explored in several works [3, 9], as a way to generate scale-aware depth maps by exploiting cross-camera extrinsics as a source of metric information. In Fig 2 we show examples of ZeroDepth pointclouds for each of these datasets, obtained by overlaying individual pointclouds from the 6 cameras in a single sample. We emphasize that these are direct transfer results, generated by evaluating ZeroDepth with-

Evaluation	Dataset	Med. Scale	*Lower is better*				*Higher is better*		
			AbsRel	SqRel	RMSE	RMSE_{log}	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
KITTI	– TA	✓	0.104	0.651	4.011	0.174	0.905	0.978	0.995
		✗	0.103	0.670	4.171	0.187	0.891	0.970	0.991
	– PD	✓	0.109	0.697	4.227	0.179	0.899	0.977	0.995
		✗	0.105	0.720	4.400	0.192	0.886	0.968	0.990
	– W	✓	0.118	0.858	4.579	0.188	0.881	0.974	0.993
		✗	0.110	0.831	4.552	0.199	0.876	0.962	0.988
	– LSD	✓	0.121	0.753	4.536	0.198	0.872	0.968	0.991
		✗	0.133	0.830	4.562	0.207	0.861	0.963	0.988
	All	✓	0.102	0.627	4.044	0.172	0.910	0.980	0.996
		✗	0.100	0.662	4.213	0.181	0.899	0.973	0.992
DDAD	– TA	✓	0.166	2.889	11.576	0.284	0.808	0.908	0.953
		✗	0.168	2.927	11.744	0.294	0.791	0.901	0.950
	– PD	✓	0.181	2.954	11.988	0.283	0.784	0.902	0.951
		✗	0.183	3.025	12.238	0.295	0.774	0.893	0.957
	– W	✓	0.198	3.470	12.767	0.328	0.772	0.886	0.949
		✗	0.202	3.657	12.928	0.338	0.765	0.879	0.942
	– LSD	✓	0.212	4.101	13.809	0.319	0.748	0.852	0.936
		✗	0.224	4.231	14.771	0.335	0.726	0.838	0.923
	All	✓	0.160	2.610	10.814	0.258	0.811	0.924	0.961
		✗	0.161	2.633	11.034	0.272	0.813	0.915	0.956
nuScenes	– TA	✓	0.250	3.912	7.258	0.330	0.741	0.881	0.931
		✗	0.266	4.161	7.494	0.341	0.738	0.879	0.928
	– PD	✓	0.255	3.812	7.468	0.342	0.727	0.865	0.919
		✗	0.266	4.239	7.629	0.354	0.712	0.853	0.907
	– W	✓	0.266	4.323	7.925	0.375	0.708	0.846	0.904
		✗	0.281	5.779	8.206	0.418	0.688	0.825	0.883
	– LSD	✓	0.278	4.411	8.328	0.409	0.671	0.827	0.888
		✗	0.303	6.462	8.858	0.421	0.655	0.806	0.861
	All	✓	0.236	3.566	7.054	0.311	0.747	0.891	0.941
		✗	0.255	4.730	7.205	0.326	0.746	0.885	0.935

Table 3: **ZeroDepth outdoor depth estimation results using different training datasets.** *All* refers to the use of all 4 considered datasets, and each additional entry indicates the removal of a specific dataset: TA for *TartanAir*, PD for *Parallel Domain*, W for *Waymo*, and LSD for *Large-Scale Driving*. We observe a consistent decrease in performance when fewer training datasets are considered, and this decrease is similar between metric and median-scaled predictions.

out fine-tuning, and these are *single-frame* results, meaning that each image was processed independently, and the reconstructed pointclouds were combined without any post-processing or alignment procedure. As we can see, these individual pointclouds seamlessly blend in overlapping areas, which indicates that our *learned scale is consistent across multi-cameras*, including across cameras with different intrinsics, resolutions, and relative vehicle orientation. Furthermore, as shown by the LiDAR pointclouds overlaid with the pointclouds, our learned scale is not only consistent across cameras, but *it is also metric*, i.e. it aligns with the "ground-truth" LiDAR information without any required post-processing.

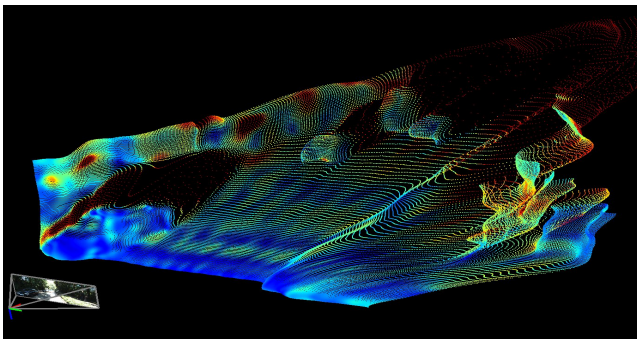
References

[1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 2

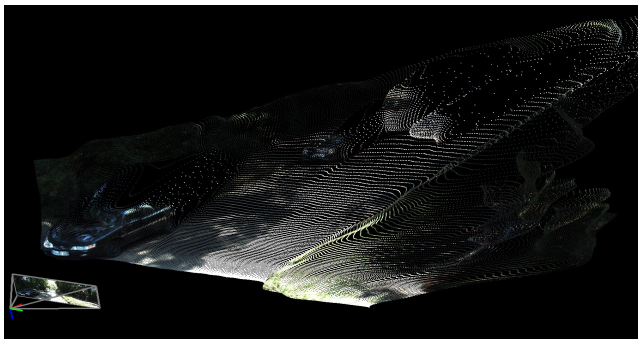
[2] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *CVPR*, 2020. 2

[3] Vitor Guizilini, Igor Vasiljevic, Rares Ambrus, Greg Shakhnarovich, and Adrien Gaidon. Full surround monodepth from multiple cameras. *arXiv:2104.00152*, 2021. 2

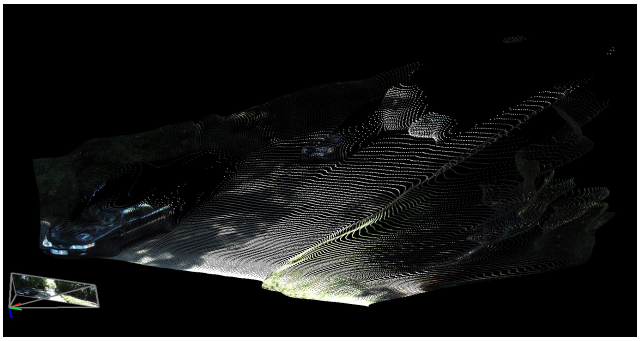
[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceed-*



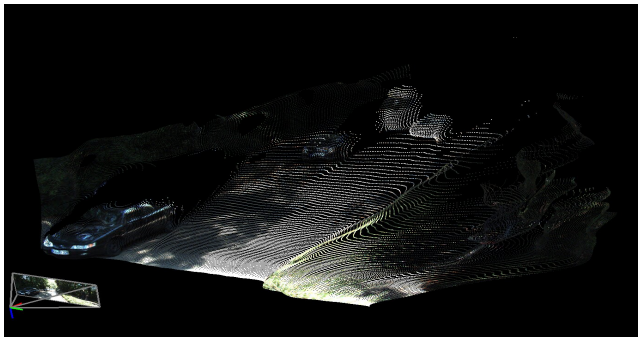
(a) Standard deviation



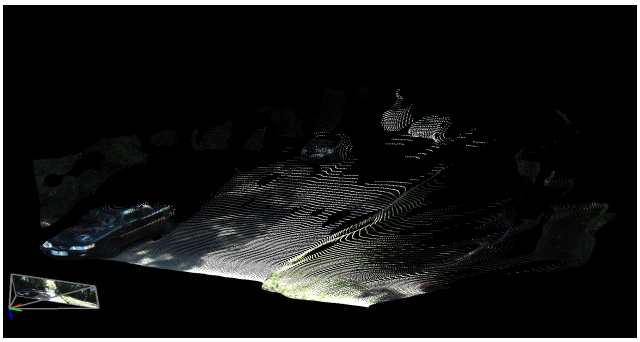
(b) 100% (RMSE 4.044)



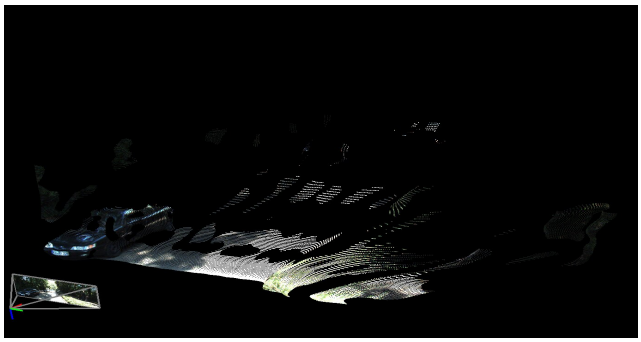
(c) 90% (RMSE 2.859)



(d) 75% (RMSE 2.132)

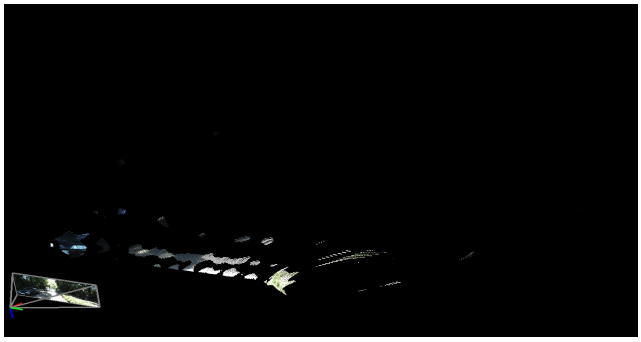


(e) 50% (RMSE 1.489)



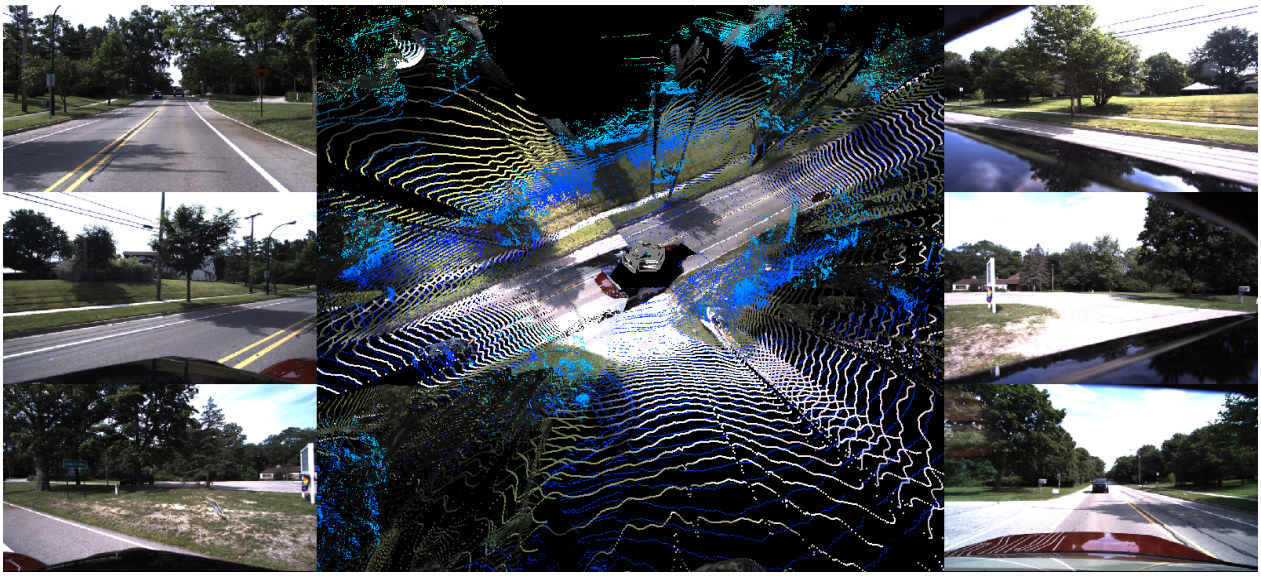
(f) 25% (RMSE 1.174)

(g) 10% (RMSE 0.723)

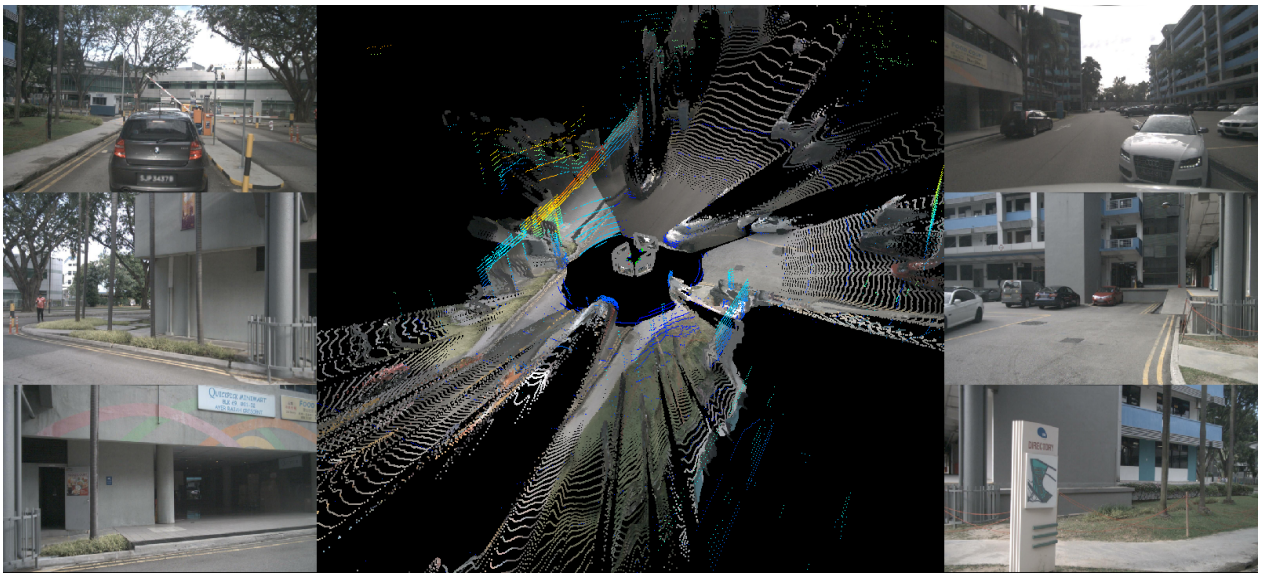


(h) 5% (RMSE 0.529)

Figure 1: **ZeroDepth pointcloud filtering based on variational uncertainty.** In (a) we show the predicted monocular pointcloud colored based on the standard deviation calculated from 10 samples. Afterwards, we show the same pointcloud filtered according to standard deviation (lowest to highest), and also report the corresponding RMSE from the filtered depth map. Even with minimal filtering (e.g., 10%) we already observe significant improvements (30%) in accuracy, mostly by removing areas with "bleeding" artifacts due to object discontinuities.



(a) DDAD



(b) nuScenes

Figure 2: **ZeroDepth full surround metric pointclouds**, obtained by overlaying predicted monocular pointclouds from the six available cameras on the (a) *DDAD* and (b) *nuScenes* datasets. LiDAR pointclouds are shown as height maps for comparison purposes only. No post-processing, scaling, or alignment of any kind was performed. More examples are shown in our supplementary video.

ings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2016. 1

[5] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 1

[6] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5885–5894, October 2021. 1

[7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. 1

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*. 2019.

1

[9] Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Yongming Rao, Guan Huang, Jiwen Lu, and Jie Zhou. Surrounddepth: Entangling surrounding views for self-supervised multi-camera depth estimation. *arXiv preprint arXiv:2204.03636*, 2022. 2