ACE-G: Improving Generalization of Scene Coordinate Regression Through Query Pre-Training

Supplementary Material

7. Hyperparameters

Architecture We use DINOv2-L with registers [23, 41] as our image encoder and use $N = 12$ cross-attention-only transformer blocks (*cf*. Fig. 4).

Pre-Training We train our network for 4.4M map code iterations (resulting in 440 000 mapping and 440 000 query iterations for the head, because we are only updating the head in every 10th iteration, *cf*. Sec. 4.2) on 8 A100 GPUs with a scene batch size $N_{\text{spb}} = 200$ and patch batch size $N_{\text{pps}} = 512$. Each map code is optimized between 6000 and 10000 iterations with $N_{\text{qstandby}} = 5000$. To focus optimization on solvable patches we found it beneficial to only use the lowest 30% losses in each batch. We use AdamW for map codes and network weights with a learning rate of 0.0001 without learning rate scheduler. During pre-training we use a map code size of $N_{\mathcal{C}} = 1024$.

Novel Scene Mapping We use slightly varying parameters for our 5 minute and 25 minute configuration following manual tuning on validation scenes. In the 5 minute setup a maximum buffer size of 4M patches, 1000 iterations, and a batch size of 40960 is used. In the 25 minute setup we spend more time budget on the buffer creation using 8M patches, increase the number of iterations to 4000, and increase the batch size to 51200. In both cases, AdamW with a one cycle learning rate schedule with maximum learning rate 0.002 is used. We use $N_{\mathcal{C}} = 4096$ resulting in 12 MB maps (full precision). During optimization we apply dropout on the image features with a dropout probability of 10%.

8. Datasets

Every combination of mapping images yields a unique map code after optimization and every other image in a sequence can potentially aid in improving the generalization performance of the coordinate regressor. Therefore, we randomly generate multiple mapping-query configurations per scene taking into account specific dataset characteristics. For most datasets, sequences are ordered in time which gives a strong clue for which images are likely covisible. Therefore, we follow an interval-based configuration scheme where the sorted image sequence is split into disjoint subsets serving as the mapping and query portion. For unsorted datasets we adjust parameters such that empirically most query views should still be solvable while also including challenging views with little visual overlap.

We follow two sampling schemes: an interspersed one, in which mapping and query intervals of varying length alternate throughout the sequence; and a query-mapping-query scheme, where a mapping interval of varying length is surrounded by two query intervals of varying length.

Table 6. **Image encoder analysis**. Accuracy, in terms of median position error (in cm), on 7Scenes, 12Scenes, Indoor-6 and RIO10 (top) with per-scene results for 7Scenes (bottom) for different image encoders. **Best** and second best highlighted.

	Static		Dynamic	
	7S	12S	I6	R10
ACE w/ ACE enc.	**1.1**	**0.7**	11.0	358.4
ACE w/ DINOv2 enc. (“DINO-ACE”)	7.2	1.9	5.6	83.8
ACE-G w/ ACE enc. (Ours)	1.3	**0.7**	11.3	144.5
ACE-G w/ DINOv2 enc. (Ours)	4.6	1.2	**4.5**	**41.1**

	Chess	Fire	Heads	Off.	Pump.	RK	Stairs
ACE w/ ACE enc.	**0.6**	**0.8**	**0.6**	1.1	**1.2**	**0.8**	**2.8**
ACE w/ DINOv2 enc. (“DINO-ACE”)	0.9	1.4	0.8	1.4	1.8	1.1	43.9
ACE-G w/ ACE enc. (Ours)	**0.6**	**0.8**	**0.6**	**1.0**	1.4	**0.8**	3.8
ACE-G w/ DINOv2 enc. (Ours)	1.0	1.5	0.9	1.4	1.8	1.1	24.5

For ARKitScenes and ScanNet++, we first sample a mapping interval, then find covisible image pairs given the image pair information published by [63] and sample a short interval around the image known to be covisible.

Beyond this interval-based sampling, we randomly switch mapping and query sequences for the MapFree dataset and randomly mirror scenes. Finally, a random rotation is applied every time a new mapping-query configuration is added to the active scenes.

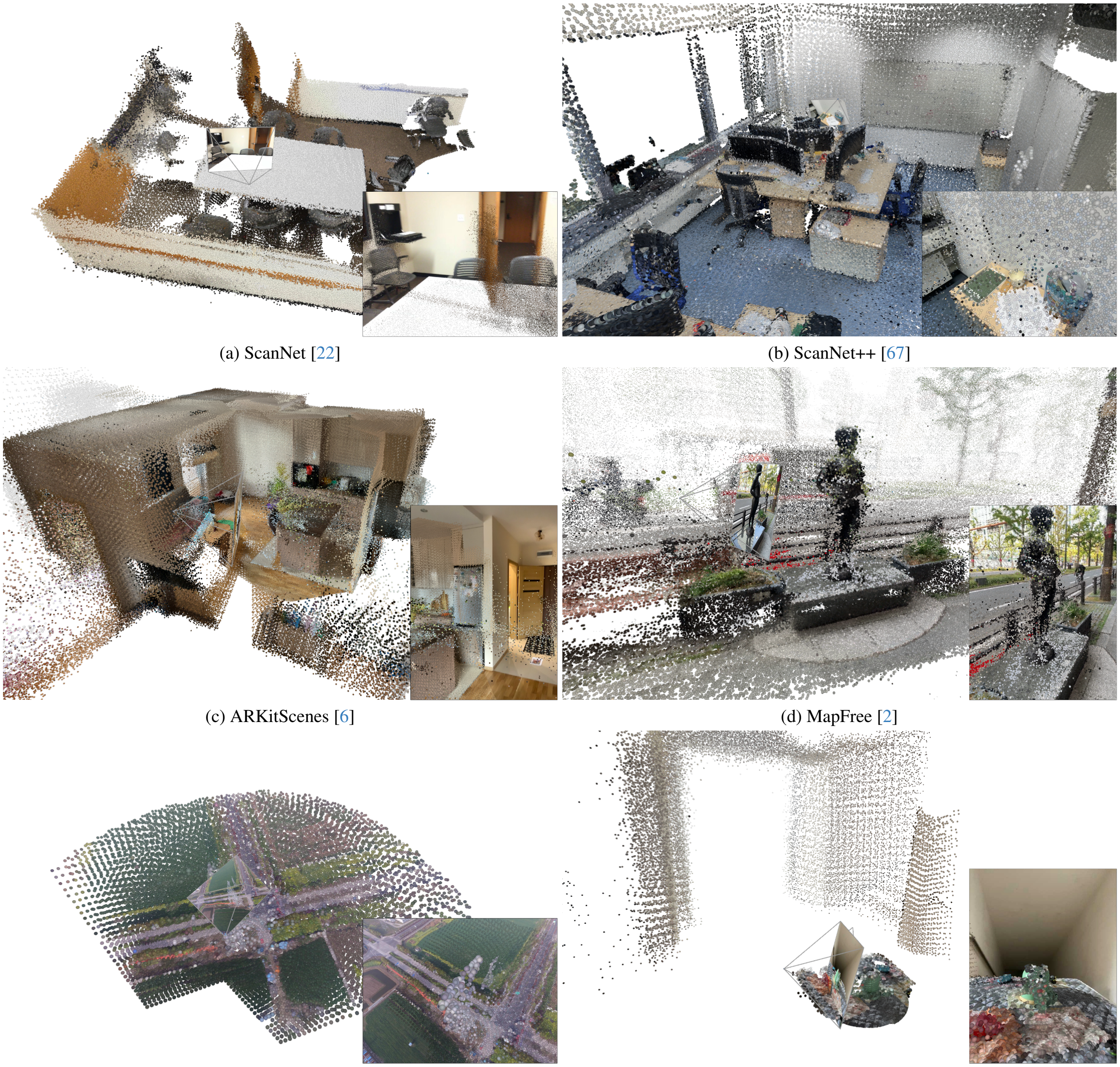
Figure 9 visualizes a mapping-query split for each included dataset.

9. Additional Experiments

9.1. Image Encoder

To better understand the interplay of our pre-training and the image encoder, we report additional results of ACE and ACE-G paired with ACE’s fully-convolutional image encoder and DINOv2 in Tab. 6. In addition to the datasets reported in the main paper, we include 7Scenes [53] and 12Scenes [58]. Datasets can be grouped into *static* (7Scenes, 12Scenes) and *dynamic* (Indoor-6, RIO10), depending on whether there are environment and lighting changes between mapping and query images.

In summary, ACE-G with DINOv2 achieves the most balanced results across datasets. The accuracy of ACE and



(a) ScanNet [22]

(b) ScanNet++ [67]

(c) ARKitScenes [6]

(d) MapFree [2]

(e) BlendedMVG [66]

(f) WildRGBD [64]

Figure 9. **Mapping-query splits used for training.** The 3D points show the accumulated scene coordinates from all mapping views. The inset shows projected ground-truth points in a query view. Note that overlap between mapping points and query view varies significantly.

ACE-G depends to some extent on the image features being used. The ACE features works well on static scenes that require little generalization but performs poorly in dynamic scenes. DINOv2 features are less precise compared to ACE features on static scenes, but generalize much better in dynamic scenes.

To further understand the differences on 7Scenes, we also include per-scene results on that dataset (Tab. 6). The performance drop is caused by one scene (Stairs), and can be attributed to DINOv2 features, not to ACE-G's architecture or pre-training.

Notably, ACE-G's architecture and pre-training consistently improves when building on-top of DINOv2 features. The strong performance of ACE on static scenes, comes at the cost of worse performance on dynamic datasets. We believe that the strong performance of ACE-G in dynamic conditions is highly relevant in practice when query images are taken long after an environment has been mapped.

Table 7. **Fine-tuning results.** Accuracy under $(20^\circ, 20\,\mathrm{cm})$ error threshold on the validation splits of the training datasets. Fine-tuning on different dataset combinations can specialize the model for specific conditions.

(%)	SN	SN++	ARK	MF	BMVG	WR
Baseline	63.5	55.5	56.0	47.1	40.6	36.7
Indoor	+1.9	+2.6	+1.2	-3.4	-7.5	-9.2
Outdoor	-8.7	-10.6	-9.6	+0.4	+3.4	-9.0
MapFree	-8.5	-10.1	-9.1	+0.2	-7.1	-8.6

9.2. Fine-Tuning

In Tab. 7 we further show validation results for three specialized models fine-tuned on a subset of datasets for 1M iterations after 4M iterations of pre-training on all 6 datasets: in one case we fine-tune only on indoor datasets, in a second case only on outdoor datasets, and in the final case only on the MapFree dataset. Interestingly, the latter models benefit less from the fine-tuning, which might suggest that the two outdoor training datasets (MapFree and BlendedMVG) are not sufficient for the variety of scenes present in these datasets.