LandscapeAR: Large Scale Outdoor Augmented Reality by Matching Photographs with Terrain Models Using Learned Descriptors – Supplementary material

Jan Brejcha[1,2][0000−0002−2091−6185], Michal Lukáč[2][0000−0002−9664−7786], Yannick Hold-Geoffroy[2][0000−0002−1060−6941], Oliver Wang[2][0000−0003−2839−7153], and Martin Čadík[1][0000−0001−7058−9912]

[1] Brno University of Technology, Faculty of Information Technology, CPhoto@FIT, Božetěchova 2, 61200 Brno, Czech Republic, http://cphoto.fit.vutbr.cz
[2] Adobe Inc., 345 Park Ave, San Jose, CA 95110-2704, USA

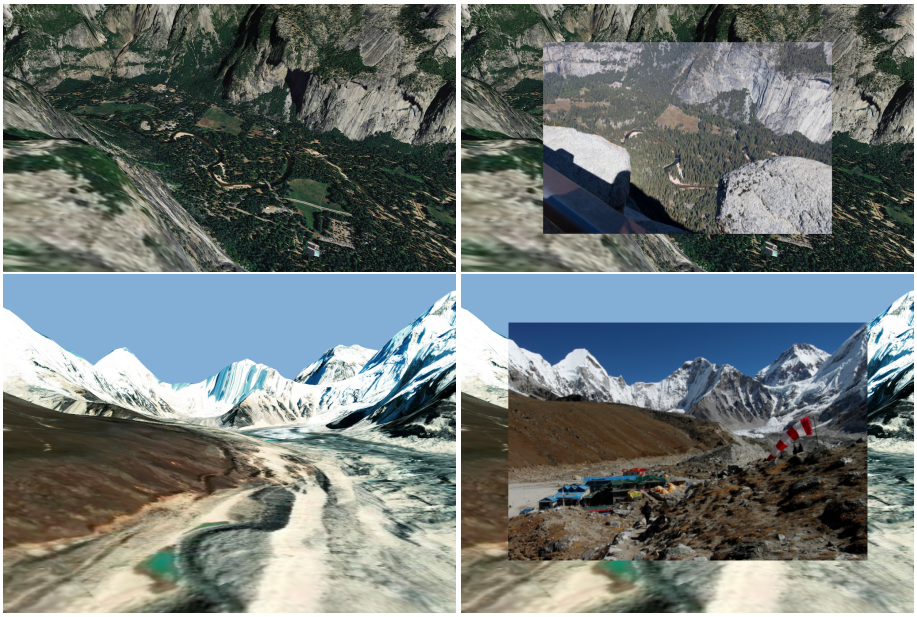

Fig. 1. Illustration of successful results obtained by pose estimation using our cross-domain matching method. Left: terrain rendered with the estimated camera postion and rotation, right: the rendered image overlaid by the photograph. First line: Yosemite Valley (image credit Kirk Northrop, https://flic.kr/p/22MAjoC), second line: Nepal – view from Gorakshep.

1 Training parameters

For training, we use a curriculum learning strategy, as described in the Sec. 3.4 of the main paper. To allow the gradual increase of the difficulty in the hard negative mining

throughout the minibatch, we introduced the tunable parameter λ in Eq. (3). We start training with the $\lambda = 0$ and increase λ by 0.05 with each 10 k steps up to a maximum hardness. Once maximum hardness is reached, we keep it constant until the end of the training. We experimentally found that a maximum of $\lambda = 0.23$ worked well for our data, with the margin set to $\alpha = 0.2$, and minimum distance in 3D was set to $m = 50$ m. We used minibatch size of 300 patches, learning rate 10^{-5}, and ADAM optimizer. To prevent overfitting we used early stopping using validation set; the network was trained for 21 epochs using 1.2M training steps.

2 Rendering Digital Elevation Models and Satellite Imagery

For rendering digital elevation models with satellite imagery overlay we use publicly available OSGEarth[3] library. In the region of European Alps we use DEM at 1 arcsecond resolution, in the region of South America and Asia we use DEM at 3 arcsecond resolution, both publicly available at http://viewfinderpanoramas.org/. In the region of USA we use DEM at 1 arcsecond resolution publicly available from USGS[4]. We use satellite imagery provided by European Space Agency (ESA), at resolution of 1-2 m in the Europe, USA and Asia, and at 5 m resolution in the region of Southern America.

3 Qualitative evaluation

We illustrate several qualitative results of our method in Fig. 1. In the top row, we see that our keypoint-based approach, unlike the horizon line based methods [3, 2], is able to precisely estimate camera pose even for images where no horizon line is visible. Additionally, our approach is expected to work well if around 100 of inliers distributed all over the photograph are available.

We found that our approach is most likely to fail on images fully covered by snow (see the top row of Fig. 2), containing a lot of high-frequency noise in the foreground (usually caused by foliage or trees), or when the photograph contains mostly flat terrain (see the bottom row of Fig. 2), where the amount of overlapping keypoints with the rendered image is low.

4 Comparison with State-of-the-Art

In addition to the comparison with state-of-the-art methods based on positional error presented in the main paper in Sec. 4.3, we also add a comparison with respect to the number of inliers given its month in the year, shown in Fig. 3. We may observe that more photographs across all datasets is usually captured during the summer and early autumn months (June – October), see dashed blue line. This correlates with the counts of inliers of all methods in the comparison – higher amounts of inliers are more likely in the summer photographs. On GeoPose3K and Nepal, our method trained with auxiliary loss

[3] http://osgearth.org
[4] https://www.usgs.gov

Fig. 2. Illustration of inaccurate results. Left: terrain rendered with the estimated camera postion and rotation, right: the rendered image overlaid by the photograph. First line: Mount Everest and Nuptse, second line: view from Alexandrovka – an observation tower near Adamov, Czech Republic.

functions (see red line) typically produces more inliers than D2Net and HardNet++. On Andes Huascaran and Yosemite, D2Net and HardNet++ are generally able to find more inliers than our method. This illustrates that images from the Nepal dataset are likely to have features similar to the training set, the Alps, which is less the case for Yosemite and Andes Huascaran. Moreover, Andes Huascaran is rendered using a different ortho-photo texture (RapidEye satellite) and was created using D2Net matches, giving an advantage to this method.

Mean and median counts of inliers for each method and dataset are illustrated in Tab. 1. Similarly to per-month number of inliers, we see that our method retrieves the most inliers on GeoPose3K and Nepal datasets, while D2Net and HardNet++ are able to retrieve more inliers on Andes Huascaran and Yosemite, respectively. However, it seems that the inlier increase of HardNet++ on Yosemite dataset is caused by few images with a large number of inliers, since the mean of HardNet++ is the largest, but the median is not—in fact, our method was able to get the largest median in number of inliers on this dataset.

5 Auxiliary loss functions in single-domain scenario

According to our experiment, auxiliary loss function defined in Eq. (4) brings further improvement over the basic variant of the cross-domain triplet loss defined in Eq. (1). To illustrate this, we evaluated each branch of our network on the single domain HPatches dataset [1] and compared with HardNet++ and D2Net in Tab. 2 on three tasks – patch

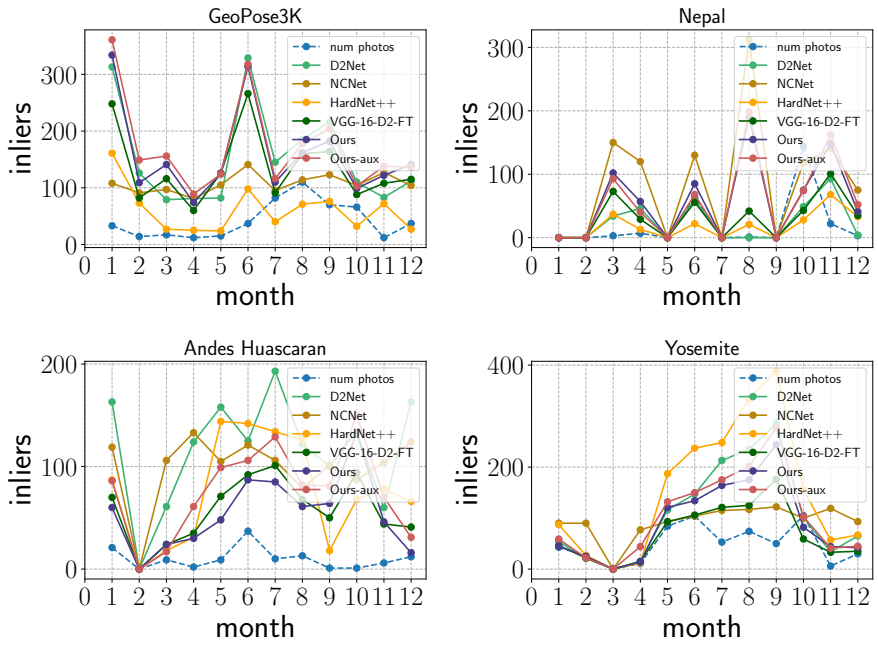

Fig. 3. Comparison of our method with state-of-the-art with respect to number of inliers given the month in four different locations across the Earth. Higher is better.

verification, matching and instance retrieval. The symbols $*$ and ◆ denote DiffSeq (negative pairs are formed by patches from different sequences) and SameSeq (negative pairs are formed by patches from the same sequence) respectively—for its exact definition, please see the HPatches paper [1]. Please note that the HPatches benchmark is a single domain dataset containing only photographs, which is not compatible with the design of our architecture; moreover, our architecture was trained for much more specific task than the competitors. Therefore, we needed to evaluate our network twice—once for each branch. Clearly HardNet++ exhibits superior performance over other methods on HPatches (see the first line in bold in Tab. 2), while on our cross-domain scenario it exhibits worse performance compared to our method (see Fig. 8 in the main paper). This illustrates that our cross-domain scenario is different from the single-domain one. On HPatches, the variant of our network trained with auxiliary loss function outperforms the variant trained with basic triplet loss, which is consistent with the comparison on our cross domain datasets. Interestingly, the best performing variant of our method is the render branch trained with auxiliary loss functions (see the last line of Tab. 2 in bold). This is most probably caused by the fact that in our train dataset the rendered images are always aligned perfectly, unlike the photographs, which eventually can contain outliers.

	GeoPose3K		Nepal		Andes Huascaran		Yosemite	
	Mean	Median	Mean	Median	Mean	Median	Mean	Median
D2Net	175.72	69.50	55.41	33.00	**134.01**	**101.00**	142.84	21.00
~~NCNet~~	~~110.81~~	~~94.50~~	~~124.71~~	~~103.00~~	~~113.17~~	~~94.00~~	~~103.13~~	~~83.00~~
HardNet++	64.27	0.00	29.97	16.50	107.56	54.00	**207.66**	18.00
VGG-16-D2-FT	144.73	69.50	51.95	22.00	71.89	63.50	91.66	19.00
Ours	166.53	83.00	**84.52**	**49.50**	63.33	48.00	121.34	22.50
Ours-aux	**178.85**	**86.50**	80.14	43.50	86.79	80.00	137.70	**24.00**

Table 1. Comparison of our method with state-of-the-art with respect to number of inliers in four different locations across the Earth. The larger the better, best performing algorithms are in bold. Although NCNet is able to get many inliers compared to other algorithms, we measured low amount of correctly localized images (see Fig. 8, NCNet in the main paper), and therefore we removed it from this comparison.

Method	**Verification**						**Matching**			**Retrieval 20k**		
	*	◆	*	◆	*	◆						
	Easy		Hard		Tough		Easy	Hard	Tough	Easy	Hard	Tough
HardNet++	**0.986**	**0.979**	**0.974**	**0.962**	**0.939**	**0.919**	**0.730**	**0.582**	**0.401**	**0.792**	**0.677**	**0.492**
D2Net	0.810	0.788	0.721	0.700	0.666	0.646	0.387	0.172	0.075	0.545	0.312	0.179
VGG-16-D2-FT	0.866	0.834	0.770	0.734	0.706	0.671	0.168	0.459	0.017	0.292	0.122	0.062
Ours-photo	0.906	0.877	0.812	0.776	0.738	0.701	0.278	0.094	0.037	0.421	0.193	0.101
Ours-render	0.899	0.868	0.811	0.774	0.740	0.703	0.222	0.070	0.026	0.375	0.173	0.089
Ours-aux-photo	0.925	0.902	0.828	0.796	0.747	0.713	0.382	0.152	0.065	0.508	0.255	0.135
Ours-aux-render	0.956	0.942	0.915	0.892	0.857	0.830	0.453	0.231	0.112	0.556	0.326	0.181

Table 2. Comparison of variants of our network to HardNet++ and D2Net on the full HPatches dataset [1] (single domain). For D2Net, we used the dense feature extractor which results in 15x15 descriptors per 65px^2 patch, from which only the central descriptor was used. Higher is better in all tasks. HardNet++ perform the best, from our methods the render branch trained with auxiliary loss gives second best result (see bottom line in bold). ∗ DiffSeq; ◆ SameSeq [1].

6 Algorithm runtimes

We implemented the pose estimation algorithm presented in Sec. 3.5 in Python for PC and a simplified version (see Sec. 5 of the main paper) for the iPhone in C++. The complete pose estimation algorithm takes around 40 seconds on PC; the simplified version runs about 1 minute on the iPhone XS. Description of 5 k patches takes approximately 0.1 seconds on PC with NVIDIA GeForce GTX 1080 and around 7 seconds on the iPhone XS.

References

1. Balntas, V., Lenc, K., Vedaldi, A., Mikolajczyk, K.: HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 (2017). https://doi.org/10.1109/CVPR.2017.410
2. Brejcha, J., Čadík, M.: Camera orientation estimation in natural scenes using semantic cues. In: 2018 International Conference on 3D Vision (3DV). pp. 208–217 (Sep 2018). https://doi.org/10.1109/3DV.2018.00033
3. Saurer, O., Baatz, G., Köser, K., Ladický, L., Pollefeys, M.: Image Based Geo-localization in the Alps. International Journal of Computer Vision pp. 1–13 (2015). https://doi.org/10.1007/s11263-015-0830-0, http://dx.doi.org/10.1007/s11263-015-0830-0