Learning to Produce Semi-dense Correspondences for Visual Localization

Supplementary Material

1. Method Details

1.1. Attention Layers

We implemented a standard transformer block [4] for the self-/cross-attention layer. However, we omitted the conventional positional encoding step because our proposed network explicitly learned keypoint embeddings to guide depth estimation. We utilized one cross-attention layer and three pairs of self and cross-attention layers to perform geometric (Eq. 7) and visual guidance (Eq. 9), respectively.

1.2. Normalization Techniques

During our experiments, we observed that applying normalization techniques to depth and loss enhances the training network's convergence speed and increases overall generalizability. Specifically, we extracted the minimum and maximum depth values, d_{min} and d_{max}, from the observed depth D^o. We normalized D^o before applying Eq. 4. Subsequently, we converted the output depth D^r in Eq. 10a back into the original depth range using d_{min} and d_{max}.

To normalize the loss in Eq. 14, we computed the standard deviation (σ) of ground-truth depths and used it as a scaling factor.

1.3. Implementation of CPA

While the PIN module (Section 3.2) is naturally differentiable when using standard neural network layers in Pytorch, it is required to make the CPA module (Section 3.3) differentiable with respect to the predicted 3D points and confidences of PIN. This ensures that our network can be trained end-to-end across multiple reference views.

The quantization function Q_s, which approximates keypoint k_i within cell size s, is simply designed as $Q_s(k_i) = \texttt{round}(\frac{k_i}{s}) * s$, where $\texttt{round}(x)$ converts x into the closest integer number.

To efficiently implement the point aggregation step in Eqs. 12&13, we extracted a set of unique quantized keypoints and then used the `index_reduce` function in Pytorch, as follows:

```
 uniq_kpts2d, indices = torch.unique(Q_s(kpts2d))
# aggregate confidence values (Eq. 13)
 agg_conf.index_reduce_(0, indices, conf, "mean")
# aggregate 2D keypoints
 agg_kpts2d.index_reduce_(0, indices, \
                          kpts2d*conf, "mean")
 agg_kpts2d = agg_kpts2d / (agg_conf + 1e-6)
# aggregate 3D points (Eq. 12)
 agg_pts3d.index_reduce_(0, indices, \
                         pts3d*conf, "mean")
 agg_pts3d = agg_pts3d / (agg_conf + 1e-6)
```

2. Detailed Experiments

2.1. Training

The proposed method was developed using PyTorch and optimized with AdamW. We conducted training on MegaDepth [7], comprising 196 scenes from various global locations. For each scene, we randomly selected the top 300 query images and the top 3 reference images for each query. During evaluation, the number of reference images can be increased to enhance performance. We directly utilized pre-trained weights for the 2D feature-matching model and updated weights exclusively for the proposed PIN and CPA (Sections 3.2&3.3). The training process utilized two NVIDIA GeForce RTX 3090 GPUs, each equipped with 24GB of memory. The initial learning rate and batch size were set to 0.001 and 4, respectively. The training spanned 40 epochs and was completed within two days. The model, trained on MegaDepth, was directly used for evaluation across all datasets, eliminating the need for finetuning or retraining.

2.2. Analysis for 7Scenes and Cambridge

7scenes. The 7scenes dataset, primarily designed for indoor scenarios, introduces several challenges, including textureless surfaces and repetitive patterns. Therefore, the SfM pipeline usually produces noisy 3D models. As illustrated in Fig. 1, the generated point clouds for multiple scenes exhibit substantial noise, failing to accurately represent the scene structures. In our evaluations of the dataset, we employed an image resolution of 640×480 and configured the quantization size (s) in the CPA module to 2. Similar to the setups of HLoc [10, 11], we employed DenseVLad [19] to select the top 10 reference images for each query image.

The quantitative results using median errors were reported in Table 1 of the main paper. Here, we further provide a qualitative evaluation in Fig. 1. We visualized the point cloud inputs and drew trajectories of the estimated camera positions. As shown in Fig. 1, the point clouds of DeViLoc contain a considerable amount of noisy points compared to HLoc. However, DeViLoc can predict more accurate camera positions, as depicted by the color codes. We also calculated the percentage of test images in which their camera positions and orientations are smaller than 3cm and 3^o respectively. The results indicate that our method is significantly better than the robust pipeline HLoc[SP+SG].

Cambridge landmarks. The Cambridge dataset includes five outdoor scenes, featuring query and reference images taken from various trajectories. We utilized

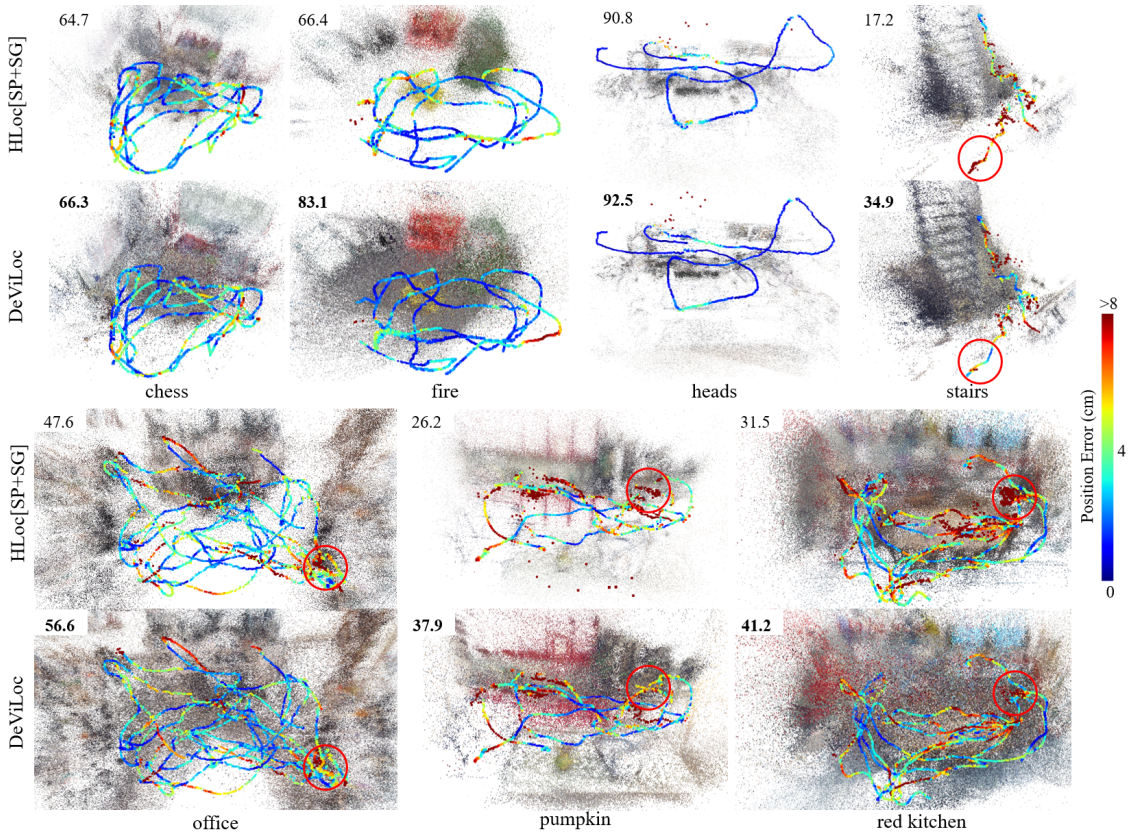

Figure 1. Qualitative results on 7scenes. We visualize the estimated camera positions for all query images with highlighted position errors using a color map. The point cloud inputs for each method are displayed in the background. Additionally, we provide the percentage of query images with a camera pose error below (3cm, 3^o). Despite having to deal with very noisy 3D inputs, DeViLoc consistently outperforms HLoc[SP+SG] across most scenes.

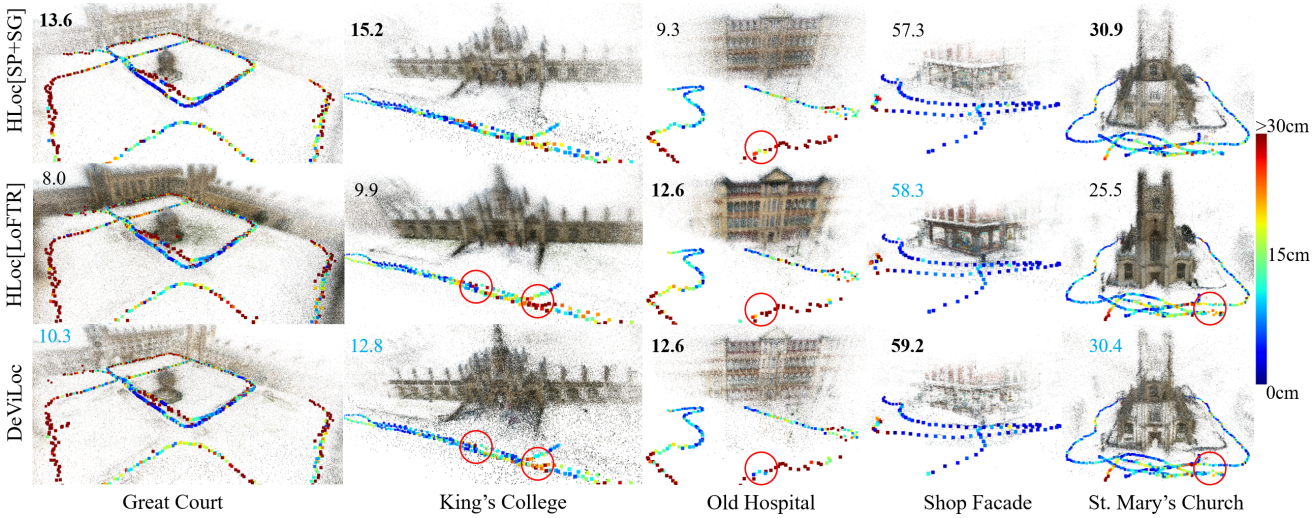

Figure 2. Qualitative evaluation for the Cambridge dataset. We compare with robust FM-based methods, HLoc[SP+SG] [3, 11] and HLoc[LoFTR] [16]. The position errors are color-coded, and the percentage of queries with the pose threshold of (5cm, 5^o) is reported. DeViLoc outperforms HLoc[LoFTR] and achieves competitive performance compared to HLoc[SP+SG], even when handling more sparse and noisy inputs.

Method	urban	suburban	park	overcast	sunny	foliage	mixed fol.	no foliage	low sun	cloudy	snow
AS [14]	81.0/87.3/92.4	62.6/70.9/81.0	45.5/51.6/62.0	64.1/70.8/78.6	55.2/62.3/71.3	58.8/65.3/73.9	59.2/67.5/77.4	83.3/88.9/94.6	65.8/73.4/82.8	71.6/77.6/84.2	73.0/81.0/90.5
D2Net [5]	94.0/97.7/99.1	93.0/95.7/98.3	89.2/93.2/95.0	92.5/95.6/97.5	86.2/91.8/95.2	88.0/92.8/95.9	94.3/96.9/98.4	98.0/99.4/**99.8**	95.1/97.5/98.7	95.5/97.5/98.9	97.2/98.9/**99.6**
HLoc[SP] [3]	89.5/94.2/97.9	76.5/82.7/92.7	57.4/64.4/80.4	77.1/82.8/91.8	65.1/72.3/86.8	69.2/75.5/88.3	75.2/81.7/90.8	88.7/92.8/96.4	78.0/83.9/91.8	83.4/87.7/94.0	80.7/86.6/93.2
R2D2 [9]	89.7/96.6/98.3	76.1/83.8/89.0	64.4/72.1/76.5	79.9/87.0/90.6	70.3/78.3/83.2	74.1/81.2/85.6	75.7/84.1/87.9	86.6/93.3/95.3	77.8/85.7/89.3	84.1/90.0/92.5	79.8/87.6/91.1
PixLoc [12]	88.3/90.4/93.7	79.6/81.1/85.2	61.0/62.5/69.4	78.5/80.0/84.8	72.4/75.6/81.8	74.3/76.8/82.7	76.6/77.7/81.8	84.1/84.8/87.7	77.5/78.4/82.2	82.0/82.9/87.0	75.7/76.4/80.4
HLoc[SP+SG]	95.5/98.6/**99.3**	90.9/94.2/97.1	85.7/89.0/91.6	92.3/95.3/96.9	86.1/91.3/94.6	88.3/92.5/95.3	91.6/94.5/96.2	95.4/97.1/98.3	91.8/94.4/96.3	95.2/97.0/98.0	92.3/94.6/96.6
HLoc+PixLoc	**96.9/98.9/99.3**	93.3/95.4/97.1	87.0/89.5/91.6	93.6/95.5/96.9	88.4/92.4/94.6	90.3/93.4/95.3	93.3/95.0/96.2	96.6/97.5/98.3	93.6/95.1/96.3	96.2/97.3/98.0	94.3/95.3/96.6
SFD2 [20]	95.0/97.5/98.6	90.5/92.7/95.3	86.4/89.1/91.2	92.1/94.0/95.8	86.3/90.3/93.4	87.9/91.0/93.9	91.9/94.0/95.5	95.3/96.6/97.6	92.4/94.4/95.8	93.3/94.7/96.3	92.9/94.6/96.0
Ours (top-10)	95.7/98.4/99.2	**97.1/98.3/99.4**	**92.1/95.1/96.3**	**96.2/97.9/98.7**	**90.4/94.8/96.8**	**92.2/95.5/97.2**	**97.1/98.6/99.2**	**98.5**/99.3/99.6	**97.4/98.7/99.2**	**97.4/98.3/98.9**	**97.7**/98.6/99.1
Ours (top-15)	96.5/**98.9/99.3**	**97.7/98.8/99.8**	**93.1/96.1/97.1**	**96.7/98.2/98.9**	**91.9/96.3/98.0**	**93.4/96.7/98.2**	**97.5/98.8/99.2**	**98.9/99.6**/99.7	**97.8/98.9/99.2**	**98.0/98.9/99.3**	**98.4/99.1/99.4**

Table 1. Detailed breakdown of the results on the extended CMU dataset. The results are categorized into different scenarios such as scene types (urban, suburban, park), weather conditions (overcast, sunny, low sun, cloudy, snow), and foliage appearance (foliage, no foliage, mixed foliage). The metrics are calculated with pose error thresholds of $\{(25\text{cm}, 2^o), (50\text{cm}, 5^o), (5\text{m}, 10^o)\}$. We additionally include the other FM-based methods, R2D2 [9] and SFD2 [20], for comparison. As depicted in the table, DeViLoc demonstrates state-of-the-art performance.

NetVLad [1] to retrieve 20 reference images for each query image. In the testing phase, the longest dimension of an image was resized to 864, and the quantization size (s) was configured to 4. For both datasets, a RANSAC threshold of 20 pixels was employed in the PnP solver during the camera pose estimation step.

The qualitative results on Cambridge are shown in Fig. 2. We compared with two FM-based methods, HLoc[SP+SG] and HLoc[LoFTR [16]]. We found that although the detector-free method LoFTR can produce dense point cloud inputs, its performance is not significantly improved in comparison to the detector-based method, SP+SG. The main reason is that the dense point clouds produced by HLoc[LoFTR] also increase the number of imprecise 3D points. Meanwhile, the transformer-based model, SuperGlue (SG), is very effective in eliminating noisy points. In contrast to both models, our point clouds built from SIFT-based COLMAP are both noisy and sparse. However, our method is still better than HLoc[LoFTR] and achieved competitive performance compared to HLoc[SP+SG].

2.3. More Details of Large-Scale Evaluation

We conducted additional evaluations of DeViLoc on an extensive visual localization benchmark that includes long-term and large-scale datasets [15]. Our evaluations were performed on three datasets, Aachen Day-Night [13, 15], RobotCar-Seasons [8, 15], and Extended CMU-Seasons [2, 18]. The predicted camera poses were submitted to the benchmark website (https://www.visuallocalization.net) to obtain recall metrics at thresholds of $(25\text{cm}, 2^o)$,$(50\text{cm}, 5^o)$, and $(5\text{m}, 10^o)$.

Aachen. The Aachen Day-Night dataset includes 4328 database images from various locations in Aachen city, along with 922 query images captured under both day and night conditions. In this evaluation, we selected the top-50 reference images per query using NetVLad [1]. We resized the longest dimension of each image to 864 and applied the quantization size of 4. We then used a RANSAC threshold of 15 pixels to estimate camera poses.

	Day-all	Night-all
DeViLoc w/o CPA ($\tau = 0$)	**57.0** / 81.7 / 97.4	27.1 / 67.3 / **92.8**
DeViLoc w/ CPA ($\tau = 0.5$)	56.9 / 81.8 / **98.0**	31.3 / 68.9 / 92.4
DeViLoc w/ CPA ($\tau = 0.8$)	56.2 / **82.0** / **98.0**	**32.5** / **69.8** / 92.5

Table 2. Effectiveness of CPA module on the RobotCar dataset.

Although our method experienced a slightly inferior performance compared to HLoc[SP+SG] as shown in Table 2 of the main paper, it still demonstrated the effectiveness in in handling low-quality point cloud input and challenging night-time conditions.

RobotCar. The RobotCar-Seasons dataset is notably challenging, featuring 26121 database images and 11934 query images taken in different seasons (winter, summer), various weather conditions (sun, rain, snow), and different times of the day (dusk, dawn, night). We employed the top-20 reference images, maintaining similar settings for image size, quantization size, and RANSAC threshold as in the Aachen evaluation.

We showcased the effectiveness of our method in producing semi-dense 2D-3D matches and filtering noise on the RobotCar dataset. Fig. 3 presents the 2D-3D matching results under various localization conditions. We observed that our method demonstrates proficiency in estimating highly confident matches, particularly in areas associated with buildings. This capability stems from our model being trained on the MegaDepth dataset. Leveraging the confidence estimation module, our approach adeptly eliminates inaccurate matches, as depicted in the orange colors. Additionally, we conducted a quantitative experiment with different setups of confidence threshold τ. Table 2 demonstrates the effectiveness of noise filtering using the CPA module.

However, there exist several cases in which our CPA module can not successfully eliminate noisy 2D-3D matches. For instance, as depicted in Fig. 3, some failure cases include *(overcast-winter, illustration 1&2)* and

(rain, illustration 1), characterized by depth predictions at extended distances. This led to a subpar performance of our method on the RobotCar dataset compared to HLoc in one metric (Table 2).

CMU. The extended CMU dataset provides a more challenging localization scenario with a high number of query images (56613) captured from different locations and conditions. The dataset contains 60937 database images. Building a point cloud input from these images requires days to finish. For FM-based methods that utilize the HLoc pipeline, it is required to build the point clouds based on the detected features. Our method did not need to implement this step because the SIFT-based point cloud is available in the dataset. Due to the large number of query images, we selected only 10 reference images per query. We then resized the width of each image to 864 and set the quantization size and the RANSAC threshold to 4 and 20, respectively.

Table 1 provides a detailed breakdown of the results from the CMU benchmark, categorizing them based on different challenging conditions to assess the performance of localization methods. As demonstrated in Table 1, DeViLoc consistently achieves over 90% accuracy across all scenarios, surpassing other methods.

Furthermore, Fig. 4 offers a qualitative comparison between our method and HLoc[SP+SG]. The comparison reveals that DeViLoc generates numerous matches that effectively capture specific structures in the scenes. While our results may contain some noisy points, their overall impact is insignificant. In contrast, HLoc heavily relies on predetermined 3D points, leading to fewer 2D-3D matches in challenging conditions, as illustrated in Slices 16 and 18.

3. Discussion

Flexibility of DeViLoc. While we utilized the detector-free image-matching model [6] to predict a substantial number of 2D-2D matches in the experiments, it is also possible to employ detector-based models for this step. In this situation, the runtime required for localization would be significantly reduced since the 2D-2D matching step is much more time-consuming than the proposed PIN and CPA modules (as demonstrated in Table 4 of the main paper). Due to the length and density of our paper, we have decided to leave this experiment to the audience after releasing the source code.

Storage Demand. Our database stores images and the 3D coordinates of point clouds, without the need of saving local features for all database images as done in HLoc. Moreover, during the experiments, we observed that our method maintains robust performance even when processing low-resolution images with a width of 864 pixels. Consequently, our localization system can reduce the storage demands by only saving low-resolution database images. However, the storage size of DeViLoc is still relatively large compared to methods that do not store local features or database images, such as NeuMap [17] or GoMatch [21]. Nevertheless, these methods involve a tradeoff in localization accuracy.

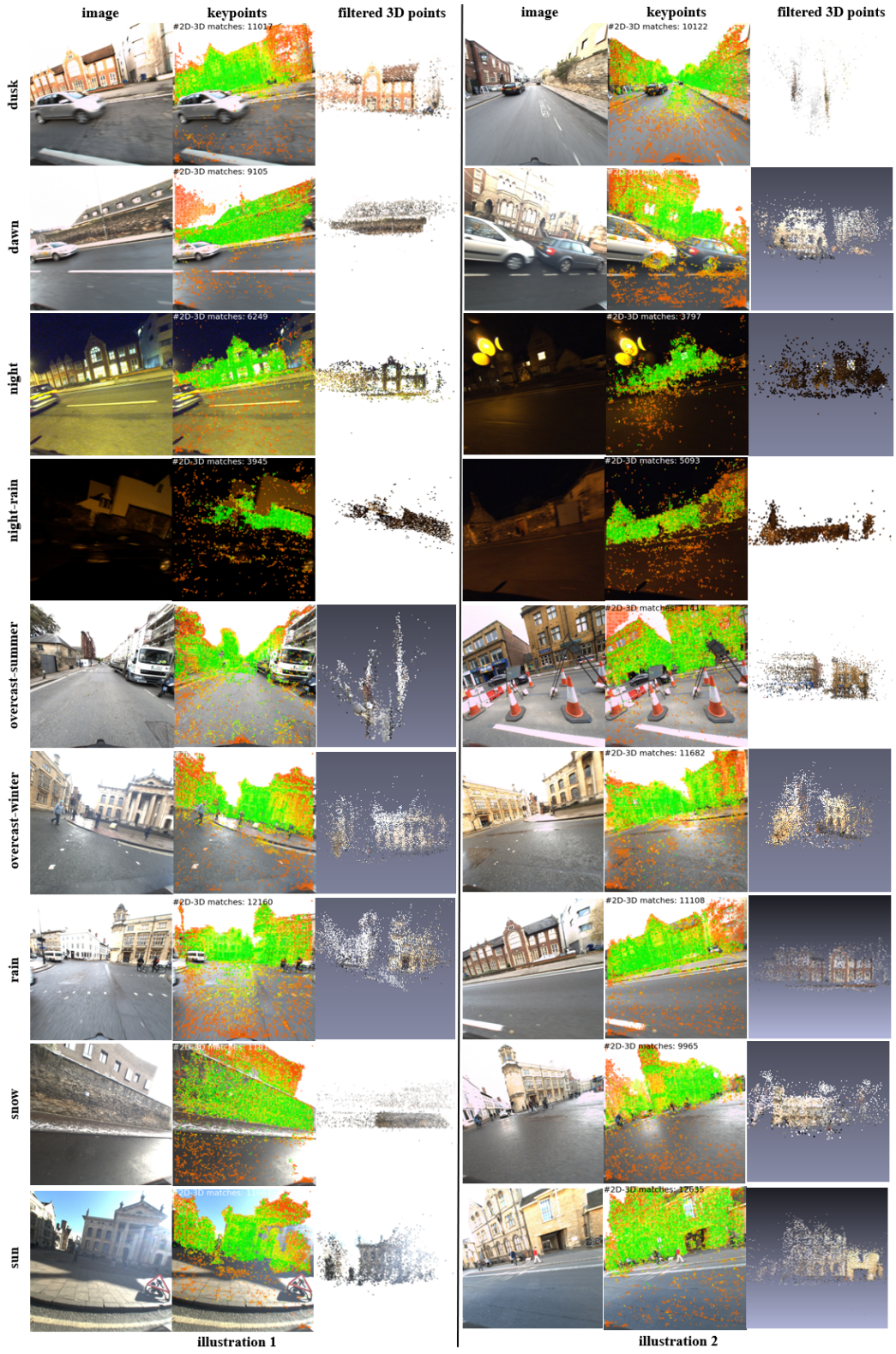

Figure 3. Illustrations of our 2D-3D matching results on the RobotCar dataset under challenging conditions, including varying times (dusk, dawn, night), weather conditions (rain, snow, sun), and seasons (summer, winter). DeViLoc successfully predicts numerous matches with uncertainties, effectively filtering out low-confidence matches (depicted in orange) and enhancing overall matching results.

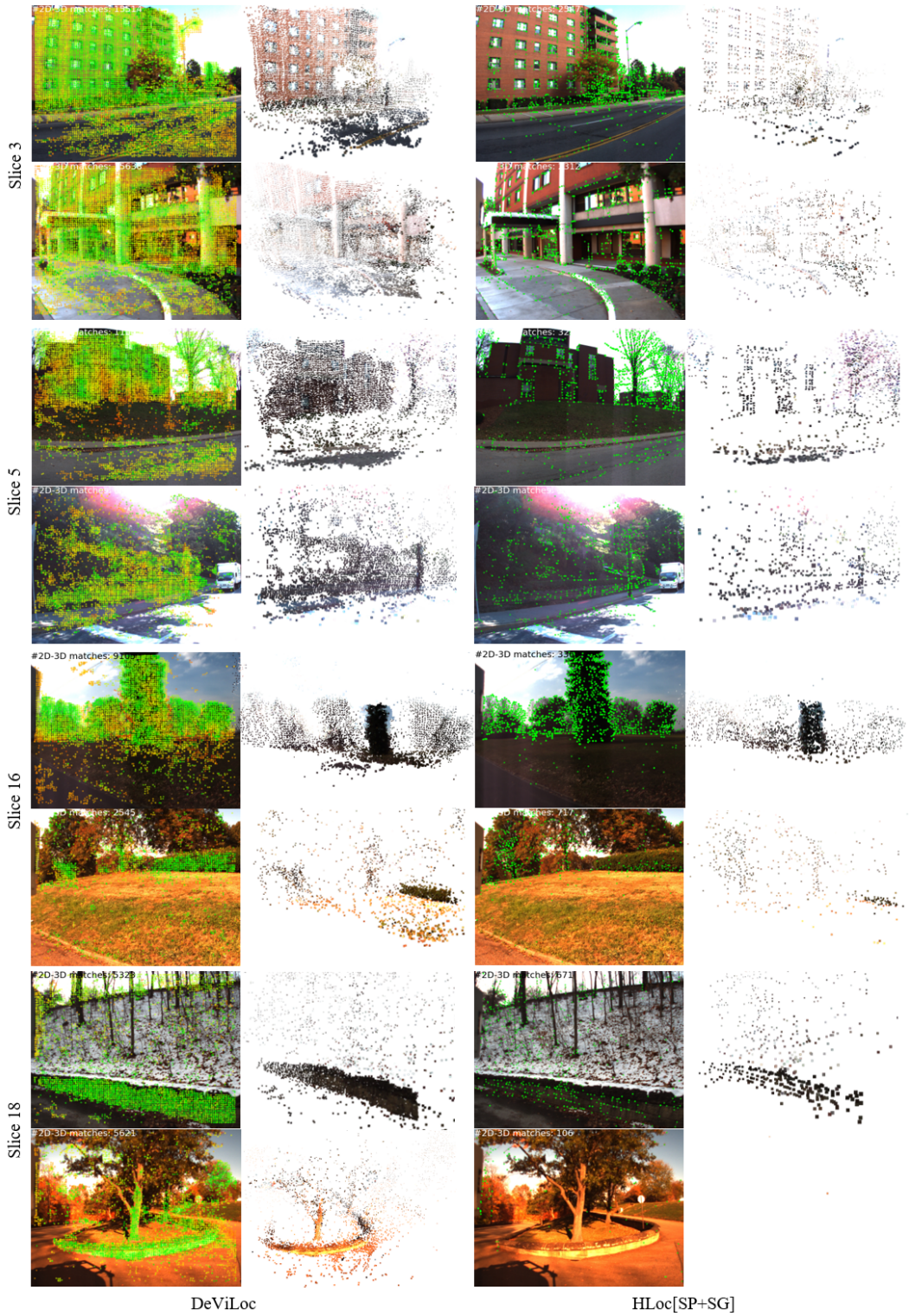

Figure 4. Qualitative comparison on the extended CMU seasons dataset. Compared to HLoc[SuperPoint+SuperGlue], DeViLoc can produce a higher number of accurate 2D-3D matches. Our method also estimates the matching confidence indicated by a color scale, from red (lowest confidence) to green (highest confidence).

References

[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016. 3

[2] Hernán Badino, Daniel Huber, and Takeo Kanade. Visual topometric localization. In *2011 IEEE Intelligent vehicles symposium (IV)*, pages 794–799. IEEE, 2011. 3

[3] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. 2, 3

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1

[5] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint detection and description of local features. *arXiv preprint arXiv:1905.03561*, 2019. 3

[6] Khang Truong Giang, Soohwan Song, and Sungho Jo. Topicfm: Robust and interpretable topic-assisted feature matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2447–2455, 2023. 4

[7] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2041–2050, 2018. 1

[8] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The oxford robotcar dataset. *The International Journal of Robotics Research*, 36(1):3–15, 2017. 3

[9] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2d2: Reliable and repeatable detector and descriptor. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 3

[10] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12716–12725, 2019. 1

[11] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 1, 2

[12] Paul-Edouard Sarlin, Ajaykumar Unagar, Mans Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, et al. Back to the feature: Learning robust camera localization from pixels to pose. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3247–3257, 2021. 3

[13] Torsten Sattler, Tobias Weyand, Bastian Leibe, and Leif Kobbelt. Image retrieval for image-based localization revisited. In *BMVC*, page 4, 2012. 3

[14] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1744–1756, 2016. 3

[15] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, et al. Benchmarking 6dof outdoor visual localization in changing conditions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8601–8610, 2018. 3

[16] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8922–8931, 2021. 2, 3

[17] Shitao Tang, Sicong Tang, Andrea Tagliasacchi, Ping Tan, and Yasutaka Furukawa. Neumap: Neural coordinate mapping by auto-transdecoder for camera localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 929–939, 2023. 4

[18] Carl Toft, Will Maddern, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, et al. Long-term visual localization revisited. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(4):2074–2088, 2020. 3

[19] Akihiko Torii, Relja Arandjelovic, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1808–1817, 2015. 1

[20] Fei Xue, Ignas Budvytis, and Roberto Cipolla. Sfd2: Semantic-guided feature detection and description. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5206–5216, 2023. 3

[21] Qunjie Zhou, Sérgio Agostinho, Aljoša Ošep, and Laura Leal-Taixé. Is geometry enough for matching in visual localization? In *European Conference on Computer Vision*, pages 407–425. Springer, 2022. 4