Back to the Feature: Learning Robust Camera Localization from Pixels to Pose

Paul-Edouard Sarlin[1*] Ajaykumar Unagar[2*] Måns Larsson[3,4] Hugo Germain[5] Carl Toft[3]
Viktor Larsson[1] Marc Pollefeys[1,6] Vincent Lepetit[5] Lars Hammarstrand[3] Fredrik Kahl[3] Torsten Sattler[3,7]

[1] Department of Computer Science, ETH Zurich [2] ETH Zurich [3] Chalmers University of Technology
[4] Eigenvision [5] Ecole des Ponts [6] Microsoft [7] Czech Technical University in Prague

Supplemental

In the following pages, we provide additional details on the experiments conducted in the main paper. Section A analyzes the convergence of the alignment depending on the accuracy of the initial pose. In Section B, we evaluate the benefits of learning environment-specific priors. In Section C, we assess the impact of the accuracy of the 3D model. Section D analyzes the ground truth poses of the RobotCar dataset, supporting the results of the evaluation performed in Section 5.3. In Section F, we explain the computation of the convergence basin shown in Figure 5 and provide additional examples. In Section E, we provide qualitative examples for the localization experiment of Section 5.2. Lastly, in Section G, we provide details on the implementation of PixLoc, its training, and the ablation study presented in Section 5.4.

A. Convergence and initial pose

Convergence: The pose optimization in PixLoc tends to converge to spurious local minima if the initial pose is too coarse, such as on the Aachen dataset, in which reference images are sparse. Since the receptive field of the CNN is limited, the convergence mostly depends on the initial 2D reprojection error, which accounts for the rotation and translation errors and for the distance to the 3D structure. The exact density of reference images required for high success thus depends on the distance to the scene.

We report in Figure 1 the success rate for different initial reprojection errors and their distribution for the oracle retrieval, with hloc as pseudo ground truth. Convergence within 1 meter is observed for 80% of the cases only when the initial error is smaller than 200 pixels and is significantly reduced for larger errors.

Initial pose: The 7Scenes and Cambridge datasets have reference poses with a high density. In driving scenarios like in the RobotCar and CMU datasets, there are no rotation changes between reference and query poses. In all these scenarios, initializing PixLoc with the pose of the first retrieved image is therefore sufficient.

To improve the performance on the Aachen dataset, the

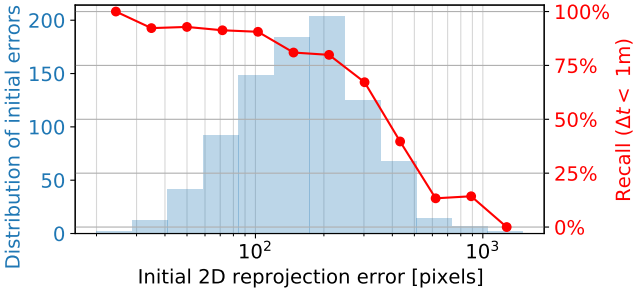

Figure 1. **Impact of the initial pose on the Aachen dataset.** The success of the pose optimization decreases with larger initial reprojection errors, which vary significantly across the 922 queries.

Initial pose	Aachen Day-Night	
	Day	Night
top-1	61.7 / 67.6 / 74.8	46.9 / 53.1 / 64.3
top-3 averaging	64.3 / 69.3 / 77.4	51.0 / 55.1 / 67.3
oracle prior	68.0 / 74.6 / 80.8	57.1 / 69.4 / 76.5

Table 1. **Selection of the initial pose.** Averaging the poses of the top retrieved images improves the convergence of PixLoc compared to simply selecting the pose of the first image.

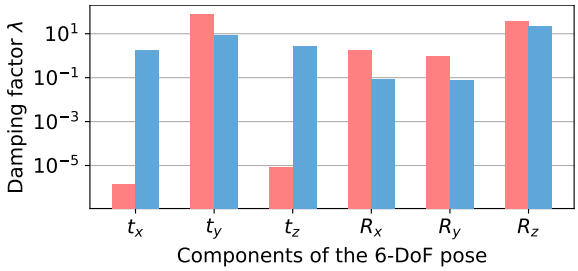

Figure 2. **Learned motion prior.** Training on data recorded with 3-DoF car-mounted cameras (CMU, in red) or with 6-DoF handheld devices (MegaDepth, in blue) results in different motion priors learned by the damping factor λ. Larger relative values indicate smaller expected motion in the corresponding direction.

results in Table 2 rely on additional filtering steps. We first cluster the top-3 retrieved reference images based on their covisibility [7,9] and only retain the images that belong to the largest cluster. We then perform a weighted average of the reference poses [4], where the weights are computed from the similarity of the global descriptors [6]. We compare in

Table 1 the results obtained with this pose averaging and with the top-1 retrieval. To further improve the convergence, one could also rerank based on featuremetric error or initialize with poses randomly sampled around top-retrieved poses.

B. Benefits of training on different datasets

The training datasets CMU and MegaDepth reflect different scenarios, autonomous driving and tourism landmark photography, respectively. Training on each one separately allows to learn task-specific priors and demonstrates the ability of PixLoc to adapt to the environment.

Each dataset depicts scenes with different semantic elements (street-level landscapes and urban landmarks, respectively) and different changes of conditions (weather and season for CMU, cameras, occluders, and viewpoints for MegaDepth). Figure 6 mentions that the models learn to ignore different unreliable elements depending on the training dataset. For example, tree silhouettes are reliable on CMU due to the small viewpoint changes, but are ignored by the model trained on MegaDepth.

Cameras also exhibit different motions, as they are either car-mounted or hand-held. Such priors are learned by the model through the damping factors, which we visualize in Figure 2. On CMU, the motion across query and reference images is mostly a translation along the x and z axis of the camera, and never along the y axis (fixed height above the ground plane) or a rotation around the z axis (fixed roll). Differently, the motion on MegaDepth is more uniformly distributed among the 6 DoF, resulting in similar factors. The relative scale between the two sets of factors is irrelevant.

These learned priors have a noticeable impact on the performance, as shown in Table 2. Training on CMU performs better than training on MegaDepth when evaluating on a driving dataset like RobotCar. When evaluating on a totally different environment like Aachen, it however still performs better than a scene-specific approach like ESAC (shown in Table 2). PixLoc thus generalizes well across scenarios but can also learn and exploit their specificities.

C. Accuracy of the 3D model

When localizing on the Cambridge Landmarks dataset, PixLoc relies on SfM models triangulated by hloc [7, 8]. For indoor scenes of the 7Scenes dataset, we found that the 3D SfM points are less accurate than the dense depth provided with the dataset. The results in the main paper (Table 1) are thus based on this dense depth.

More specifically, we rely on the depth maps rendered by Brachmann *et al*. [1], which are aligned to the color images and are less noisy than the original depth maps. We simply replace each 3D SfM point by back-projecting one of its 2D observations using the interpolated depth and the image pose. This 3D model has the same sparsity as the SfM point cloud

Training dataset	Aachen (urban scenes like MD)		CMU (natural scenes)	
	Day	Night	Urban	Park
MD	**68.0 / 74.6 / 80.8**	**57.1 / 69.4 / 76.5**	78.3 / 81.8 / 94.6	72.5 / 75.5 / 90.3
CMU	54.4 / 62.6 / 74.3	46.9 / 54.1 / 68.4	**91.9 / 93.4 / 95.8**	**84.0 / 85.8 / 90.9**

Table 2. **Cross-dataset evaluation with oracle prior.** Training and testing in different environments does not perform as well as training for the target distribution. Task-specific priors learned by PixLoc, like semantics and motion, are thus largely beneficial.

3D from	median error in translation/rotation (cm/°) ↓							R↑
	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	
SfM	3/0.90	**2**/0.87	**1/0.79**	**3**/0.96	5/1.42	4/1.44	6/1.38	69.5
RGB-D	**2/0.80**	**2/0.73**	**1**/0.82	**3/0.82**	**4/1.21**	**3/1.20**	**5/1.30**	**75.7**

Table 3. **Depth sensor fusion vs. SfM point cloud.** For the 7Scenes indoor environment, localizing with 3D points obtained from depth maps fused across multiple view (RGB-D SLAM) is more accurate than with point clouds triangulated with SfM.

but is more accurate. This process is fair as it relies on the same data as all other learning-based approaches, which use the full dense 3D model for training.

We show in Table 3 the impact on the performance of PixLoc. Using this corrected 3D model results in more accurate localization than the triangulated SfM model.

D. Inaccuracy of ground truth poses

The RobotCar v2 dataset has publicly available ground truth poses for a subset of the queries. We project 3D SfM points into the query images using ground truth poses and those estimated by hloc. We observe in many instances a large reprojection error, where hloc poses look qualitatively more accurate. Some examples are shown in Figure 3. This might explain why no method localizes more than 58% of the daytime images at the finest threshold according to the public leaderboard [1]. This might also explain why refining poses with PixLoc does not show improvements for the day-time queries of RobotCar, as observed in Section 5.3.

Similar artifacts were found in training sequences of the Extended CMU Seasons dataset, making the training supervision noisier. We however do not know if this also applies to the poses of the test sequences because such poses are not publicly available.

E. Qualitative examples

We show examples of successful localization on the Extended CMU Seasons dataset in Figure 4. We show failure cases in Figure 5. Similarly, we show successful and failed examples on the Aachen Day-Night dataset in Figures 6 and 7, respectively. Videos and animations of the uncertainties and the optimization are available along with the code and trained weights at `github.com/cvg/pixloc`.

[1] `https://www.visuallocalization.net/benchmark/`

F. Attraction basin

Computation: We compute the basin of attraction of a given point by backtracking feature gradients throughout the levels and scales. For each pixel, we consider the 2 neighbors, in an 8-connected neighborhood, that are in the direction opposite to the feature gradient $^{\partial \mathbf{F}_q}/_{\partial \mathbf{p}_q^i}{}^\top \mathbf{r}_k^i$. A pixel is in the basin of attraction if any of those two are themselves in the basin. The voting is performed in a soft manner using the gradient angle, resulting in a basin score for each pixel. We first label the point of interest as in the basin and then iteratively run the algorithm at each level, from the finest to the coarsest level, moving to the next one when the scores stop changing. Note that the total convergence basin of the pose, which corresponds to the aggregation of all the points, might be smaller or larger.

Visualization: We show one example in Figure 5 in the main paper, where we color pixels that belong to the basin by changing their hue according to the angle of the total gradient. We show additional examples in Figure 8, but showing the gradient field as arrows only.

G. Experimental details

We now provide more details about the implementation of PixLoc and the experiments.

Implementation: The CNN and the optimizer are implemented in PyTorch [5]. The linear system of the Levenberg-Marquardt step (Equation 4) is solved using the Cholesky decomposition. The lookup of features and uncertainty is computed via bilinear interpolation. We use the Cauchy robust cost function with scale 0.1. When computing the residuals or the Jacobian matrix, we ignore points that project outside the image or within 2 pixels of the image borders. We set $\lambda_{\min}{=}{-}6$ and $\lambda_{\max}{=}5$.

Training: We train PixLoc with image pairs composed of a query image and a single reference image. For each pair, we sample 512 3D points visible in the reference image according to the SfM covisibility information. We apply gradient checkpointing to each block of the encoder and of the decoder to minimize the GPU memory consumption. The network is trained for 50k iteration with a constant learning rate of 10^{-5} and the Adam optimizer [3]. To stabilize the training, the average loss per pair is clamped to 50 pixels and the per-parameter gradients are clipped to $[-1, 1]$.

When training on the CMU dataset, we use slices 8-12 and 22-25 for training and slices 6, 13, 21 for validation. We train with batches of 3 image pairs. The images are resized such that their smallest dimension is 720 pixels and we sample square crops of 720 pixels The query pose is initialized with the pose of the reference image.

When training on the MegaDepth dataset, we use the same split of scenes as Dusmanu *et al*. [2] and sample image pairs with an overlap score in $[0.3, 1]$. In addition, we rotate images that are not upright using the gravity direction of each scene. All images are resized such that their smallest dimension is 512 pixels, and we sample square crops of 512 pixels. PixLoc is then trained with batches of 6 image pairs.

Figure 3. **Inaccurate RobotCar ground truth poses.** We plot the projection of 3D SfM points in the query images according to the ground truth (in blue) and hloc (in red) poses. We project the same points in the reference images using the reference poses (in blue). Query points using hloc are better aligned to the reference points, indicating that the ground truth query poses are inaccurate.

Inference: In order to keep the runtime reasonable, we use 5

or 3 reference images when initializing from hloc or retrieved reference poses, respectively. The optimization runs at each level for at most 100 iterations, but stops when either the gradient or the step are small enough. When refining hloc poses, we only optimize over the medium and fine levels as the initial estimate is always sufficiently good. All images are resized such that their longest dimension is equal to 1024 pixels. For the multiscale inference, the resized images are successively aligned at scale 1/4 and 1.

Ablation study: We sample 2000 query images from slices 6, 7, 13, and 21 of the CMU dataset. To generate challenging pairs, we select the closest reference image that is at least 4 meters away. For the baseline based on a fixed damping factor λ, we use $\lambda{=}10^{-2}$. As GN-Net [10] has no publicly-available implementation, we reimplemented it and trained it with our settings on the CMU dataset. The GN-Net loss has several hyperparameters: the Gauss-Newton sampling vicinity, the weight of the contrastive loss, and the margin of its negative term. We performed an extensive hyperparameter search and report the best results obtained. Our training data is significantly more difficult than the one used in the original paper [10], with significantly larger baselines and appearance changes. This explains the large performance gap observed in Table 3 compared to the results originally reported.

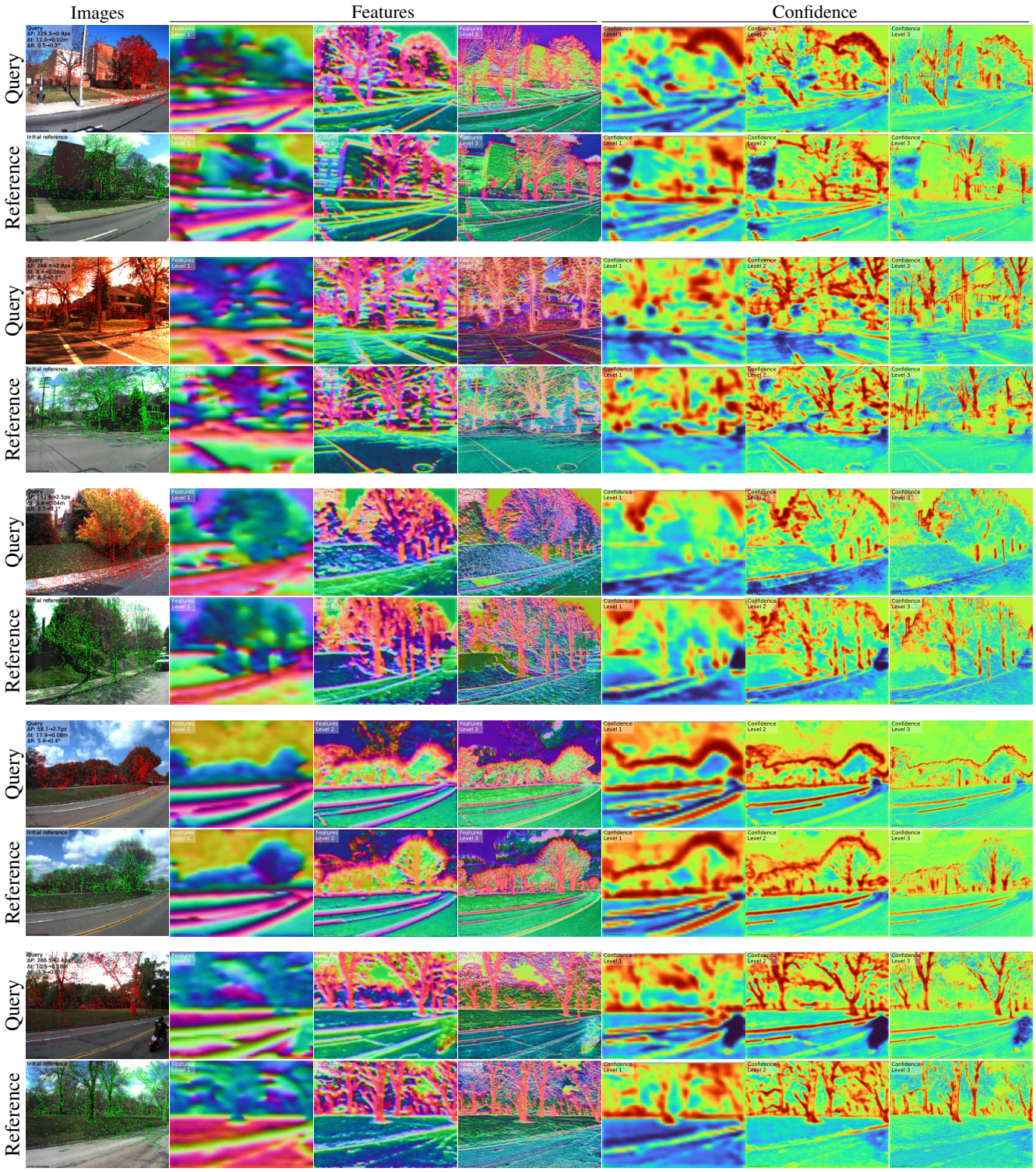

Figure 4. **Successful localization on the CMU dataset.** We show 5 challenging queries with large initial errors and large cross-season appearance changes that are successfully localized by PixLoc. We project 3D SfM points into the initial reference image (in green) and into the query image using the estimated pose (in red). We show the features at the 3 different levels, mapping them to RGB using PCA. We also show the confidence maps, where blue pixels are ignored while red ones are more important for the optimization. Features useful for localization are invariant across seasons and thus appear in similar colors.

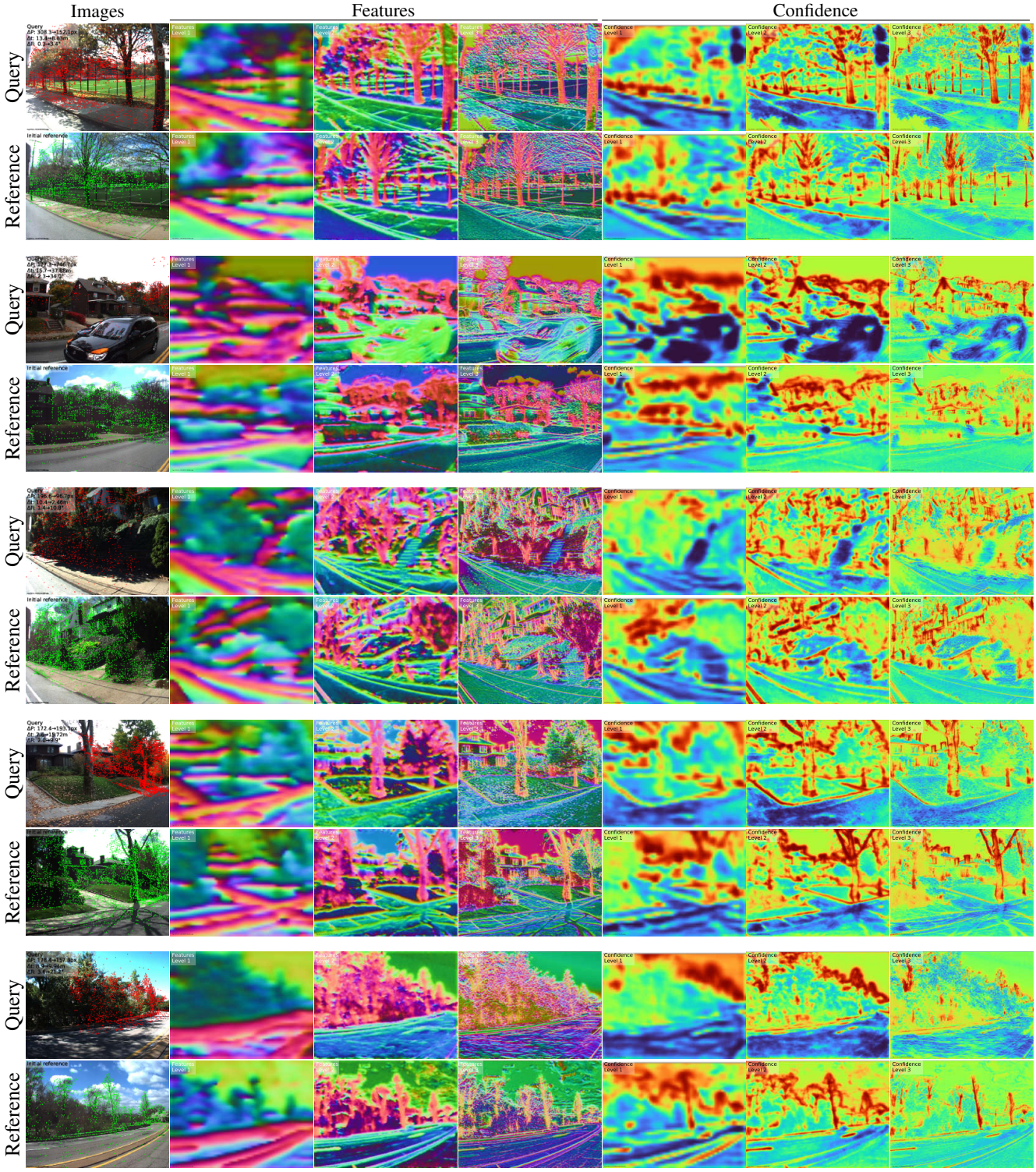

Figure 5. **Failure cases on the CMU dataset.** We show examples for which the optimization results in a large final error. This is often due to repeated elements or to a lack of spatial context of the coarse features or a lack of distinctive elements. Natural scenes are be particularly challenging when tree trunks and vegetation cannot be easily distinguished.

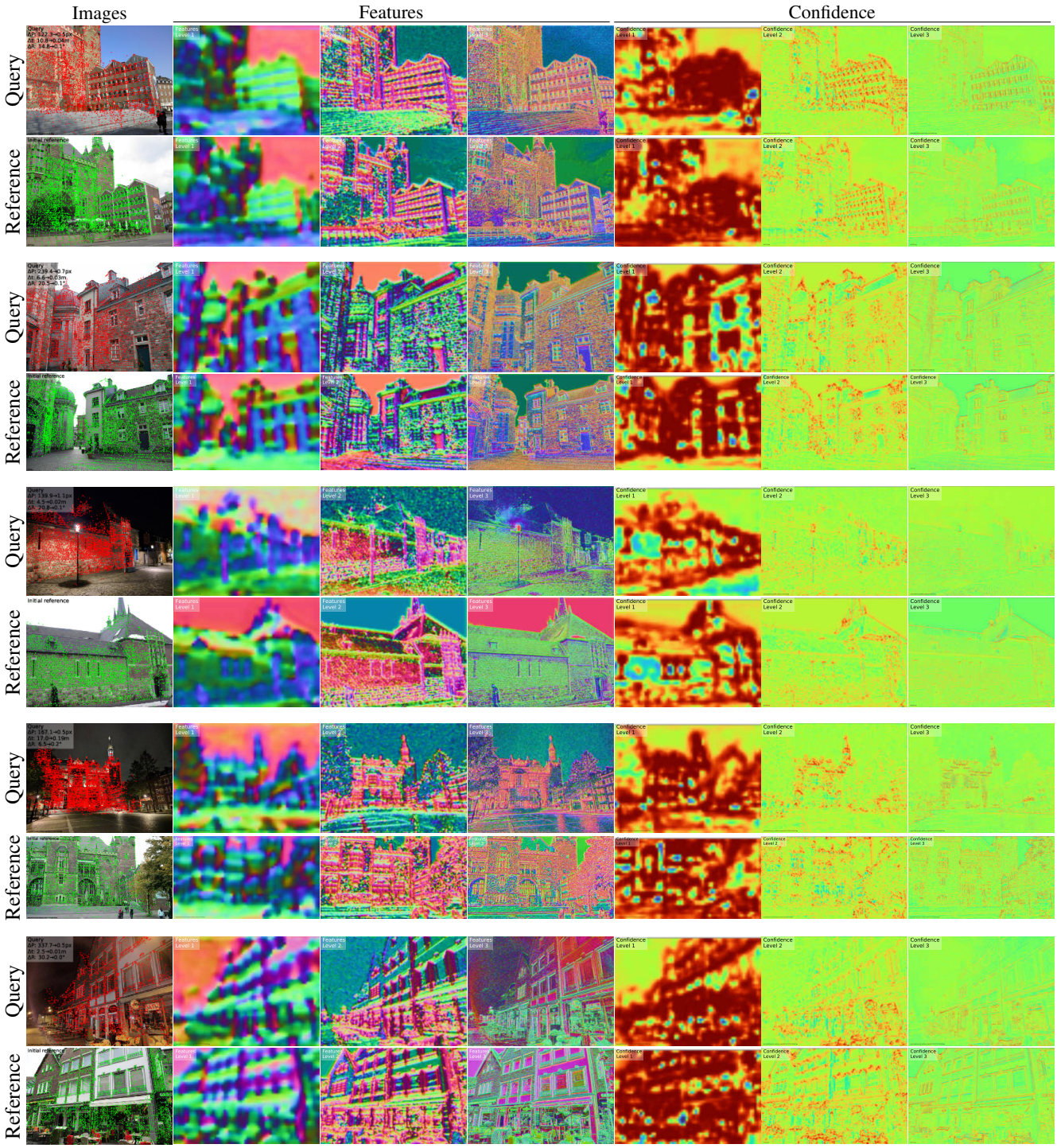

Figure 6. **Successful localization on the Aachen dataset.** We show 5 challenging queries with large initial errors and large day-night appearance changes that are successfully localized by PixLoc. The reprojection and pose errors are computed with respect to the pose estimated by hloc.

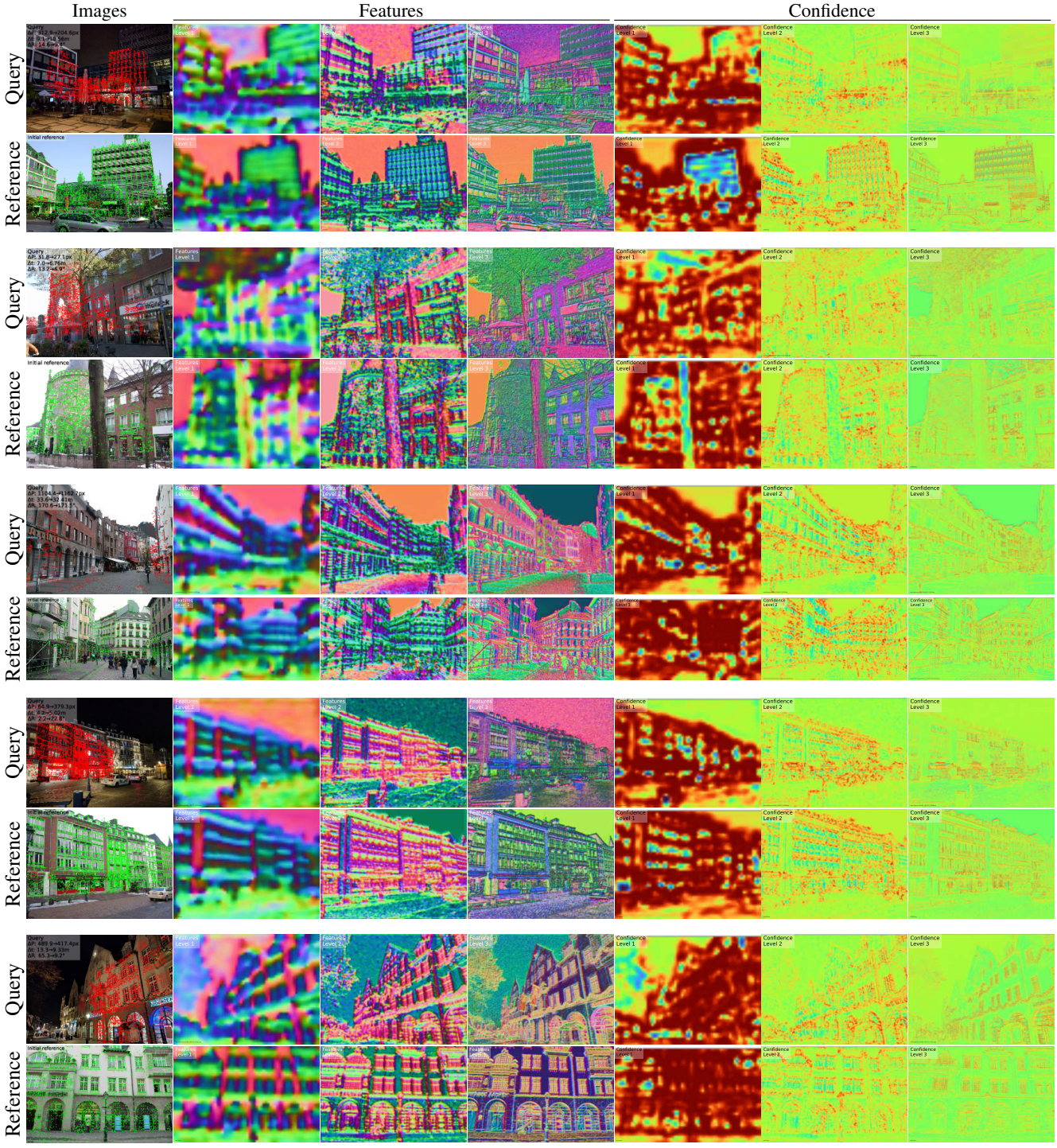

Figure 7. **Failure cases on the Aachen dataset.** Convergence to a local and incorrect minima can be due to large appearance changes (row 1), occlusion (row 2), large viewpoint change (row 3) or repeated structures on facades (rows 4 and 5).

Figure 8. **Convergence basin.** We show the convergence basins of individual selected points given cross-season query and reference images from the CMU dataset. The last row shows smaller basins due to repeated patterns like poles or tree silhouettes.

References

[1] Eric Brachmann and Carsten Rother. Visual camera relocalization from RGB and RGB-D images using DSAC. *TPAMI*, 2021. 2

[2] 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. In *CVPR*, 2019. 3

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014. 3

[4] F Landis Markley, Yang Cheng, John L Crassidis, and Yaakov Oshman. Averaging quaternions. *Journal of Guidance, Control, and Dynamics*, 30(4):1193–1197, 2007. 1

[5] 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 *NeurIPS*, 2019. 3

[6] Noé Pion, Martin Humenberger, Gabriela Csurka, Yohann Cabon, and Torsten Sattler. Benchmarking image retrieval for visual localization. In *3DV*, 2020. 1

[7] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, 2019. 1, 2

[8] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 2

[9] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *TPAMI*, 39(9):1744–1756, 2016. 1

[10] Lukas Von Stumberg, Patrick Wenzel, Qadeer Khan, and Daniel Cremers. GN-Net: The Gauss-Newton loss for multi-weather relocalization. *RA-L*, 5(2):890–897, 2020. 4