Leveraging Pixel Correspondences for Sparse-to-Dense Feature-Metric Localization

Lixin Xue
D-INFK
ETH Zurich
lixxue@ethz.ch

Zimeng Jiang
D-ITET
ETH Zurich
zjiang@ethz.ch

Le Chen
D-ITET
ETH Zurich
lechen@ethz.ch

Abstract

Visual localization is a key component to many robotics systems. However, it is very challenging in changing conditions such as day-night or summer-winter. Based on Sparse-to-Dense Hypercolumn Matching [12], we improve the localization accuracy by (1) performing Feature-metric PnP given an initial estimation of the pose and (2) training on the supervision of pixel correspondences using double margin contrastive loss and Gauss-Newton loss to generate better feature maps. Experimental results show that Feature-metric PnP refines pose estimation and we achieve our best accuracy when combine it with features trained on correspondences.

1. Introduction

In recent years, an increasing number of mobile robot navigation systems, such as autonomous driving cars and micro drones, have been deployed in large, uncontrolled, and GPS-denied environments for disaster relief, industrial plant inspection, cargo delivery, etc. As the key component of those tasks, visual localization has been a significant subject for researchers. However, in changing conditions such as day-night or summer-winter, accurately predicting the 6 DoF camera pose of a visual query with respect to a reference frame can be very challenging and remains an unsolved problem[28].

To tackle the visual localization problem, a hierarchical approach called HFNet[25] first uses an image retrieval module to obtain localization candidates. Then it establishes 2D-2D correspondences between the query image and the reference candidate based on deep image features. This approach incurs significant runtime savings and achieves remarkable localization robustness. Germain et al. [12] later extended it with Sparse-to-Dense Hypercolumn Matching (S2DHM) to improve the image matching results.

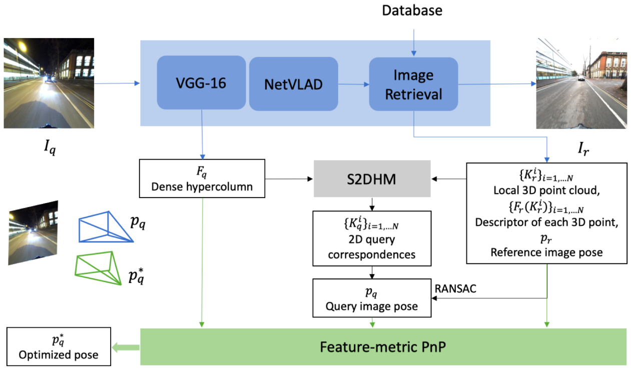

Figure 1. **S2DHM + Feature-PnP pipeline.** Based on S2DHM, the RANSAC-PnP pose is further fed into our Feature-metric PnP module, together with 2D-2D correspondences between query and reference image, 2D-3D correspondences between reference image and 3D world points, sparse reference hypercolumn of each 3D point, and the dense hypercolumns of query image. The output is the optimized pose. Different from S2DHM, where hypercolumns are extracted from the VGG-16 backbone specifically trained for image retrieval, we fine-tune VGG-16 used in S2DHM with supervision on cross-season correspondences and extract hypercolumns from it.

However, inaccurate matches produced by S2DHM still negatively affect the localization accuracy as offsets of a few pixels can lead to localization errors of several meters[13]. To produce more accurate correspondences, we propose to match keypoints' features in two images to further refine the estimated pose through our Feature-PnP module. With the recently released correspondences data between images taken in different conditions, we further fine-tune the CNN in the S2DHM pipeline to produce deep features more suitable for image matching and Feature-PnP. Experiment results indicate the effectiveness in improving localization accuracy of our Feature-PnP module and the fine-tuned features. Figure 2 and Figure 4 demonstrate the qualitative results of our improvements.

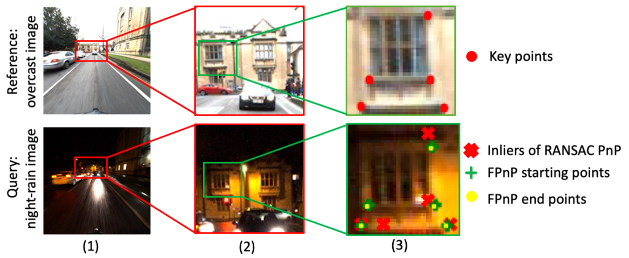

Figure 2. **Feature-PnP optimizes correspondences.** This is one example of finding accurate correspondences under challenging conditions. The night-rain image has poor illumination and strong reflection. As illustrated in (3), inliers of RANSAC-PnP are points of maximum correlation to their corresponding keypoints in the reference image, but they can be very inaccurate. Feature-PnP starting points are those reprojected after RANSAC-PnP. They are already remarkably accurate. But Feature-PnP end points are even better, although points only move at most 1-2 pixels. In fact, very few pixels' moving can lead to significant changes of the pose.

2. Related Works

2.1. Visual localization methods

Many high-level robotic tasks require precise 6-DoF poses, thus motivating the development of methods dealing with visual localization problem under strong visual changes. Visual localization methods are traditionally classified into two categories: structure-based or image-based. [33, 27, 1, 6]

Hierarchical localization combines the two approaches and constructs a pipeline [17, 21]. Sarlin et al. [26] proposed to first localize at the map level using learned image-wide global descriptors, and then estimate a precise pose from 2D-3D correspondences computed in the 3D point cloud subset provided by the returned top-ranked images from the database in the image retrieval step. In this way, competitive results could be obtained at low computational costs. Since these are poorly repeatable in extreme conditions, Germain et al. [12] later extended it with Sparse-to-Dense Hypercolumn Matching (S2DHM), using an off-the-shelf retrieval network. They perform an exhaustive search in the counterpart image, which is implemented efficiently by running convolutional operations on dense feature maps with a sparse set of local hypercolumn descriptors.

2.2. Metric learning for feature matching

Metric learning aims to learn a distance metric for effective similarity measurements between input samples by pulling similar samples closer in embedding space and pushing dissimilar ones apart [10]. The robotics community has developed many methods utilizing deep metric learning to learn feature descriptors for visual localization. Methods like [8] used deep metric learning to directly learn the feature mapping that preserves either geometric or semantic similarity for generic correspondences.

The contrastive loss [14] is widely used in metric learning. It encourages all positive samples to be close, while all negative samples should be separated by a certain fixed distance. For example, von Stumberg et al. [34] used contrastive loss together with Gauss-Newton loss to learn deep features. However, forcing all samples to the same fixed distance can be quite restrictive. Schroff et al. [30] introduced triplet loss, which only requires negative samples to be farther away than any positive samples. Methods like LFNet [23] and D2Net [11] applied triplet loss to tackle localization tasks. Hao et al. [15] pointed out that single margin contrastive loss will bias the network toward positive image pairs during training, so they proposed double margin contrastive loss, in which a margin for positive samples is introduced.

In deep metric learning, the sample selection strategies play an equal or more important role than the loss, but they are relatively less studied [36]. For the contrastive loss it is common to select from all positive samples randomly [7]. For the triplet loss, semi-hard negative mining, first introduced by FaceNet [30], is widely adopted [22, 24]. Shrivastava et al. [31] proposed an online hard example mining algorithm for training region-based ConvNet detectors. LFNet [23] employed a progressive mining strategy to obtain the most informative patches possible, in which the selected negative samples will become harder and harder as training goes.

Estimating the camera pose could be difficult under lighting and weather changes if the features are not robust to visual changes. For visual feature descriptors, Schmidt et al. [29] trained a deep neural network with the contrastive loss to produce viewpoint- and lighting-invariant descriptors for localization. Wohlhart and Lepetit [35] used CNN to compute descriptors by enforcing simple similarity and dissimilarity constraints between the descriptors. Von Stumberg et al. [34] proposed a Gauss-Newton loss to learn weather invariant deep features suitable for relocalization tracking. Dense pixel-wise features across the whole image provide powerful representation for localization. In practice, dense features have shown to lead to better matching results than sparse feature matching [9].

3. Method

3.1. The Sparse-to-Dense Localization

Given a query image I_q, S2DHM[12] first uses an image retrieval module to find a reference image I_r in the dataset which covers the same area as the query image. Specifically, a VGG-16[32] together with NetVLAD layer[2] pretrained for the image retrieval task is used to encode the image and then find nearest neighbor images in the feature space.

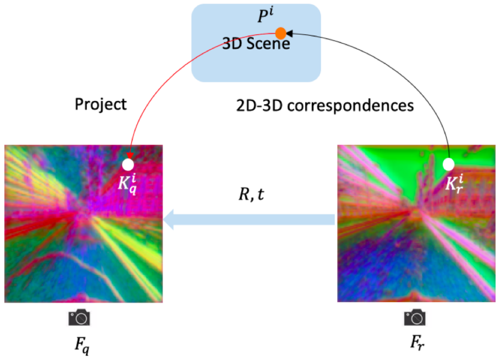

Figure 3. **Illustration of Feature-PnP.** The goal is to refine the query image pose in a unsupervised fashion by minimizing feature distance between keypoints in two images. Suppose we have an initial pose estimation for a query image and its corresponding pose-annotated reference image. The initial estimation can be absolute pose or relative pose $[R, t]$ with respect to the reference image. Given 2D-3D correspondences K_r^i and P^i, P^i can be projected into the query image based on the initial pose. We compare keypoints features $F_q(K_i^q)$ and $F_r(K_i^r)$ using a suitable loss function . Minimizing such loss forces matches to be adjusted so that they have the same features, therefore refine the pose.

With high-resolution feature maps (hypercolumns) got by interpolating and concatenating VGG features at different layers, they obtain 2D correspondences by matching local features in a sparse-to-dense manner. With such 2D-2D correspondences between the query image I_q and the reference image I_r, and the given 2D-3D correspondences between the I_r and the world points, they get 2D-3D correspondences for I_q and use the PnP algorithm in a RANSAC scheme to get an estimation of the camera pose for the query image I_q.

3.2. Feature-metric PnP

In challenging conditions like day-vs-night, summer-vs-winter, the above-mentioned pipeline might produce poor results due to bad matches. To improve the pose estimation, we would like to utilize deep features which are invariant to lighting, weather, and season changes to some extent. To be more specific, we want to adjust the matches by forcing corresponding keypoints in two images to have the same features. In other words, we would like to minimize the following objective function:

$$L(p_q) = \sum_i \rho(e_i), \tag{1}$$

where $\rho(\cdot)$ is a suitable loss function, e_i is the feature difference:

$$e_i = F_q(K_q^i) - F_r(K_r^i). \tag{2}$$

The p_q is the pose of the query image, $K_\cdot^i$ represents the keypoint, $F_\cdot(\cdot)$ is the corresponding feature. Figure 3 illustrates the idea of Feature-PnP.

We can minimize such objective function via iterative optimization methods, such as gradient descent, Gauss-Newton, and Levenberge-Marquart algorithms. However, these methods are designed to work on flat Euclidean space $\mathbb{R}^N$. In the case of pose estimation where we use quaternion or transformation matrix to represent a pose , the pose space forms a 6D manifold embedded in a higher-dimensional Euclidean space ($\mathbb{R}^7$ for quaternion representation and $\mathbb{R}^{16}$ for transformation matrix representation). In this case, one step iteration in the embedded Euclidean space usually leads to an invalid pose. An elegant solution to this problem is to optimize the intrinsic 6D manifold $SE(3)$ using an exponential and logarithm map. For further details, please refer to [5].

The key step here is to obtain the jacobian of the feature error e_i with regard to the pose p_q so that we can use the above mentioned iterative optimization method to get an update of the pose. Using the chain rule, we can decompose the gradient into three parts:

$$\frac{\mathrm{d}e_i}{\mathrm{d}p_q} = \frac{\mathrm{d}e_i}{\mathrm{d}K_q^i}\frac{\mathrm{d}K_q^i}{\mathrm{d}P^i}\frac{\mathrm{d}P^i}{\mathrm{d}p_q} \tag{3}$$

The first term on the right-hand side can be approximated numerically by calculating the image gradient. The second term is the gradient of the projection operator, which projects 3D point P_q^i to 2D pixel coordinate K_q^i. The third term, which is the gradient of 3D point P_q^i with regard to the pose p_q, bears an analytical solution. For briefness, please refers to Appendix A.2 of [5] for further details.

With the computed Jacobian, we can further obtain the gradient and Hessian matrix of the loss function with respect to the current pose. Then we use Levenberg-Marquardt algorithm to optimize the pose so that the total loss $L(p_q)$ converges to a local minimum.

Based on the result of local validation, we found some key implementation details that are useful in improving the performance of Feature-PnP. Instead of using the widely used Sobel filter for the image gradient, we found that computing second order accurate central differences largely boosts the performance. The difference between two image gradients is that Sobel filter produces smoother gradient as it takes the gradient of neighboring pixels into account as well. Also, the bilinear interpolation for both image features and image gradients are important as we are using a relatively low resolution feature map compared to the original image resolution. Without bilinear interpolation, we might end up with the same feature for different pixels. We found that the performance of Feature-PnP is relatively stable with different loss functions. We believe this is because Feature-PnP is based on the inliers found by RANSAC-PnP, where there are few outliers, and robust loss functions are therefore unnecessary.

3.3. Train on correspondences

One can directly feed Feature-PnP with feature maps trained with supervision for the task of image retrieval, for instance, the hypercolumns extracted in S2DHM. However, these features may offer more global rather than pixel-wise information. Therefore, we fine-tune the VGG-16 backbone used in S2DHM with supervision on correspondences, such that it explicitly learns features tailored for pixel-level matching. The hypercolumns generated from fine-tuned VGG-16 should be weather invariant with pixel-wise accuracy, serving as better inputs for Feature-PnP.

A two-stream Siamese VGG-16 network is fed with a pair of images $\mathbf{I}_a$ and $\mathbf{I}_b$. We extract intermediate features $\mathbf{F}_a^l$ and $\mathbf{F}_b^l$ from the same layers as S2DHM selects for hypercolumn matching, where l denotes the layer to be extracted. The loss is performed on each selected intermediate feature layer. Every input image pair has known positive correspondences, N_+. Positive means that a keypoint $K_a \in \mathbb{R}^2$ in $\mathbf{I}_a$ and a keypoint $K_b \in \mathbb{R}^2$ in $\mathbf{I}_b$ correspond to the same point in the 3D scene, otherwise they are negative correspondences. To learn discriminative feature representation, both positive and negative correspondences have to be considered and we use double margin contrastive loss.

The positive correspondences are ground-truth data, while the negative correspondences, N_-, are sampled during training. Randomly sampling often yields easy samples that contribute no loss, while with the hardest sampling the training can quickly collapse. To balance in between, we utilize a progressive mining strategy [23] with distance constrain [37] to obtain the most informative matches. Given positive correspondence K_a and K_b, we first obtain all candidate pixels $\mathbf{u}_b \in \mathbb{R}^2$ in $\mathbf{I}_b$ that satisfy the distance constrain $||K_b - \mathbf{u}_b|| > \alpha$, where α is the distance threshold. Then we sort $\mathbf{u}_b$ by loss in increasing order and sample randomly over the smallest M, where $M = max(5, 300e^{\frac{-0.6k}{10000}})$ and k is the current iteration. As training goes, we progressively mine harder and harder samples and end up with sampling from a pool of 5 hardest ones.

Pixel-wise double margin contrastive loss [16] (DMC) attempts to minimize the distance between positive matches and maximize the distance between negative matches. It is formulated as:

$$L_c(\mathbf{F}_a, \mathbf{F}_b, l) = L_+(\mathbf{F}_a, \mathbf{F}_b, l) + L_-(\mathbf{F}_a, \mathbf{F}_b, l) \quad (4)$$

$$L_+(\mathbf{F}_a, \mathbf{F}_b, l) = \frac{1}{N_+} \sum_{N_+} max(0, D_{feat} - M_+)^2 \quad (5)$$

$$L_-(\mathbf{F}_a, \mathbf{F}_b, l) = \frac{1}{N_-} \sum_{N_-} max(0, M_- - D_{feat})^2 \quad (6)$$

,where $D_{feat}(\cdot)$ is the L_2 distance between features: $D_{feat} = ||\mathbf{F}_a^l(K_a) - \mathbf{F}_b^l(K_b)||$. M_+ and M_- denote the margin for positive matches and negative matches respectively. Conventional single margin contrastive loss induces a bias towards positive matches, since they always contribute to the final loss, while negative matches contribute only when D_{feat} is larger than the margin. In contrast, DMC loss separates margins for positive and negative matches, resulting a more balanced update for the weights of the network.

3.4. Gauss-Newton loss

While the DMC loss encourages the network to learn distinctive features for different identities, our final goal is to produce features suitable for pose estimation, specifically, Feature-PnP. Hence, Gauss-Newton (GN) loss is applied as a regularization term. It follows the same formulation as described in section 3.2. For detailed derivation and calculation, please refer to [34]. The intuition is that when applying a perturbation to the current solution, the network should maximize the probability density of correct correspondences. Therefore, optimizing over GN loss is analogous to performing Gauss-Newton update. Applying GN loss on multi-scale feature maps thereby forces the features to provide a larger convergence basin for downstream Levenberg-Marquardt optimization in Feature-PnP.

The final loss is a weighted sum of DMC loss and GN loss performed at each feature layer. It is formulated as:

$$L(\mathbf{F}_a, \mathbf{F}_b) = \sum_l L_c(\mathbf{F}_a, \mathbf{F}_b, l) + \lambda \sum_l L_g(\mathbf{F}_a, \mathbf{F}_b, l) \quad (7)$$

where L_g represents the GN loss and λ is the weighting coefficient.

4. Experiments

4.1. Datasets

We evaluate our method on two challenging datasets: the RobotCar Seasons dataset[28] and the Extended CMU-Seasons dataset[28]. The RobotCar Seasons dataset uses a subset of the images provided in the RobotCar [20] dataset. The reference and query images were captured by cameras mounted on a car. One traversal is used to define a reference condition (overcast) and the reference scene representation. Other traversals, covering different seasonal and illumination conditions, are used for the query. It contains images taken at night times with a lot of motion blur.

The Extended CMU Seasons dataset uses a subset of the images provided in the CMU Visual Localization dataset[3]. It uses images taken under a single reference condition (sunny with no foliage). For this reference condition, the dataset provides a reference 3D model reconstructed using Structure-from-Motion. In addition, query images taken under different conditions at different locations are provided.

Both two datasets represent an autonomous driving scenario, where it is necessary to localize images taken under varying seasonal conditions against a reference scene representation.

4.2. Training

For all training experiments, we use the AdamW optimizer [19] with a learning rate of 1e-7 and a weight decay of 0.05. For the DMC loss term, we set M_+ to 0.1, and M_- to 1. For the GN loss term, the maximum distance of the start point to the correct point, called the vicinity parameter, is usually small. Here we set it to 1 pixel for all experiments. The first term of GN loss is weighted by 1 while the second term is weighted by 2/7. All steps of optimization take a single image pair as input. For each pair of images, we select 1024 positive correspondences randomly and sample 1024 negative correspondence using the progressive mining strategy mentioned in Section 3.3. The final loss is the sum of DMC loss and GN loss where the weighting coefficient λ is 0.5.

4.3. Evaluation Setup

For fast iteration and hyperparameter tuning, we use reference images of the RobotCar Seasons dataset as query images where we have ground-truth poses for local evaluation. Specifically, we randomly choose 500 images from 6954 reference images and use them for the query. We find the image with the most inliers in the reference images (we skip the exact same image) and report its pose error. We use mean translation error, mean rotation error (in degrees), and percentage of images in the coarse-precision, medium-precision, and high-precision introduced in the [28] as the evaluation metric. One thing worth mentioning is that this local validation setup is not representative enough as both the query images and the reference images are captured under the same conditions. So tuning hyperparameters in this setup would make our algorithm bias towards easy examples with good initial pose estimation. For the evaluation of the real query images, we just run our algorithms over all images and submit the result to the evaluation server[1]. We use the same ratio test value for both datasets. We set the exhaustive search factor to 0.006.

4.4. Effectiveness of Feature-Metric PnP

Following the S2DHM[12] pipeline, we simply use our Feature-PnP module to optimize the initial pose estimation given by the RANSAC-PnP module as indicated in Figure 1. We evaluate our method on the RobotCar Seasons dataset and the Extended CMU Seasons dataset. The result shown in Table 1 suggests the effectiveness of our Feature-PnP module, especially in the High-Precision (pose error under 2.5m and 2°). This indicates that with a good initial starting point, our Feature-PnP module works well. However, the limited improvement in the coarse precision manifests that this Feature-PnP module barely helps when the starting point is too bad. As shown in Figure 2, Feature-PnP end points are only 0-2 pixels off from starting points. This observation agrees with the fact that typical images and corresponding feature maps are extremely non-convex. Taking derivative as stated in Section 3.2 is only valid in a vicinity of probably 1-2 pixels, which means bad initial solution can never converge to the correct one.

4.5. Effect of different deep image features

To train better features for image matching and our proposed Feature-PnP module, we utilize the recently released 2D-2D point matches between image taken under different conditions[18]. To train deep features with such supervision, we experiment with different contrastive/triplet loss together with GN loss[34]. We found the DMC loss proposed in [16] works best with GN loss so we use it throughout our experiments.

The first insight of our experiments is that training purely on correspondences produces unsatisfactory features for image matching as shown in Table 4.5. This is reasonable since correspondence only provides sparse constraints on image features where most pixels rarely appear in the loss function. Therefore, such supervision is more suitable for a fine-tuning step when the features have been trained on upstream tasks, e.g. the image retrieval task in our setting.

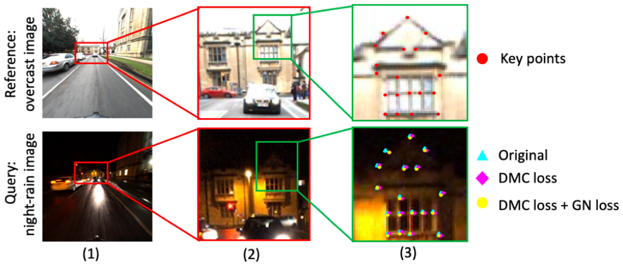

Figure 4. **Feature-PnP results using different features**. (3) shows that although there is some overlapping among correspondences obtained from different features, DMC loss + GN loss produces overall the most accurate correspondences regardless of strong visual change. DMC loss + GN loss performs better than using only DMC loss, for that GN loss ensures larger convergence basin for optimization in Feature-PnP. It is worth noting that original S2DHM features are actually very robust and even better than other two features for some keypoints.

We report localization results on 5 different features: 1) the original S2DHM[12] feature; 2) features fine-tuned with DMC loss; 3) features fine-tuned with DMC loss and GN loss (DMC+GN). To improve performance on the challenging night conditions, we also train features solely on

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

	RobotCar Seasons						*Extended CMU Seasons*								
Method	*Day-All*			*Night-All*			*Urban*			*Suburban*			*Park*		
	High	Medium	Coarse	High	Medium	Coarse	High	Medium	Coarse	High	Medium	Coarse	High	Medium	Coarse
Original + rPnP	46.1	77.5	95.1	30.1	70.2	94.5	47.2	71.5	93.3	39.3	63.8	91.5	26.0	47.5	80.2
Original + fPnP	**48.0**	**77.9**	95.1	**32.3**	**71.1**	94.5	**50.6**	**73.9**	**93.5**	**42.6**	**66.0**	**91.7**	**29.0**	**49.7**	**80.4**

Table 1. **RANSAC-PnP's and Feature-PnP's performance on the two datasets**. We report localization recalls in percent, for three translation and orientation thresholds (high, medium, and coarse) as in [28]. Here "Original" refers to using the features trained on the image retrieval task provided by S2DHM[12].

RobotCar Seasons	*Day-All*			*Night-All*		
Method	High	Medium	Coarse	High	Medium	Coarse
Train from scratch + rPnP	40.7	73.5	94.6	5.2	17.2	41.8
Fine-tune + rPnP	48.3	**79.2**	**95.1**	27.3	64.1	93.5
Train from scratch + fPnP	41.7	74.2	94.6	6.2	18.8	42.3
Fine-tune + fPnP	**49.6**	78.9	**95.1**	**30.0**	**65.6**	**93.8**

Table 2. **Importance of pretraining**. Features pretrained on the image retrieval task as in [12] result in significantly better localization results, especially in the challenging night conditions.

RobotCar Seasons	*Day-All*			*Night-All*		
Method	High	Medium	Coarse	High	Medium	Coarse
Original + rPnP	46.1	77.5	95.1	**30.1**	**70.2**	94.5
DMC + rPnP	47.7	78.9	95.1	27.3	64.5	93.1
DMC + rPnP*	47.3	78.4	95.1	28.9	69.5	**94.6**
DMC + GN + rPnP	**48.3**	**79.2**	95.1	27.3	64.1	93.5
DMC + GN + rPnP*	47.9	77.7	95.1	29.6	69.7	94.4
Original + fPnP	48.0	77.9	95.1	32.3	71.1	**94.5**
DMC + fPnP	49.1	78.8	**95.2**	30.5	66.6	93.3
DMC + fPnP*	48.4	78.3	95.1	30.9	**71.3**	94.4
DMC + GN + fPnP	**49.6**	**78.9**	95.1	30.0	65.6	93.8
DMC + GN + fPnP*	48.2	78.0	95.1	**33.8**	71.2	**94.5**

Table 3. **Comparison of learned features.** DMC refers to original S2DHM[12] features fine-tuned with DMC loss. DMC+GN refers to features trained together with GN loss. Results with * refer to features trained solely on image correspondences on overcast-night image pairs.

overcast-night correspondences in the DMC and DMC+GN setting. The recall accuracy is summarized in Table 3. Figure 4 shows correspondences of different features after Feature-PnP optimization.

From the result above, we find out both DMC loss and GN loss contribute to the improvement of the accuracy in the Day-All setting. However, when we train features on both day-night and day-day correspondences, the performance on Night-All degrades. We conjecture that the imbalance in the correspondences data is one of the reason for the inferior result: the day-day correspondences are two times more than the day-night correspondences. Also, another reason is that it is hard to learn features invariant to day-night lighting changes. We tried weighted loss for different correspondences but the result is even worse than the uniform weight version indicated in Table 3. Only the version trained solely on day-night correspondences gives a boost in the Night-All condition, while with minor degradation in the Day-All condition compared to the uniform weight version. We also find that when training with only day-night correspondences, DMC+GN boosts Feature-PnP results on Night-All setting compared with use DMC loss only, specially on high precision. This demonstrates that GN loss as a regularization term works quite well for challenging conditions.

5. Conclusion

In this paper, we introduced Feature-PnP to fine-tune the pose estimated by RANSAC-PnP. To devise features suitable for pose estimation, we also presented a way to train on pixel correspondences by combining DMC loss and GN loss. The experiments showed that the Feature-PnP gives superior performance in terms of accuracy. Furthermore, both DMC loss and GN loss contribute to the improvement of the accuracy in day conditions. Training solely on day-night image pairs with DMC+GN loss significantly boosts Feature-PnP results, especially on high precision. In the future, we would like to explore new ways of training features on correspondences, such as training with image retrieval tasks or regressing correspondence maps as done in S2DNet[13].

Work Distribution

Lixin Xue is responsible for the implementation of Feature-PnP and its integration into the S2DHM pipeline. Zimeng Jiang focuses on feature visualization and implementing the DMC and GN loss to train on correspondences. Le Chen contributes to all the experiments and part of the training on correspondences code. Our code is publicly available.[2][3].

Acknowledgments

The authors gratefully thank Mr. Paul-Edouard Sarlin, Dr. Viktor Larsson, Prof. Marc Pollefeys, and other people in the ETH Zurich Computer Vision and Geometry group for valuable discussion and suggestions.

[2] https://github.com/zimengjiang/S2DHM/tree/vgg/s2dhm

[3] https://github.com/zimengjiang/gn_net

References

[1] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. 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.

[2] R. Arandjelovic, P. Gronát, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *CVPR*, pages 5297–5307. IEEE Computer Society, 2016.

[3] H. Badino, D. . Huber, and T. Kanade. The CMU Visual Localization Data Set, 2011.

[4] J. T. Barron. A general and adaptive robust loss function. *CVPR*, 2019.

[5] J.-L. Blanco. A tutorial on SE(3) transformation parameterizations and on-manifold optimization. Technical Report 012010, University of Malaga, 2010.

[6] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother. Dsac-differentiable ransac for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6684–6692, 2017.

[7] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE, 2005.

[8] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker. Universal correspondence network. In *Advances in Neural Information Processing Systems*, pages 2414–2422, 2016.

[9] J. Czarnowski, S. Leutenegger, and A. J. Davison. Semantic texture for robust dense tracking. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 860–868, 2017.

[10] Y. Duan, L. Chen, J. Lu, and J. Zhou. Deep embedding learning with discriminative sampling policy. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4964–4973, 2019.

[11] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8092–8101, 2019.

[12] H. Germain, G. Bourmaud, and V. Lepetit. Sparse-to-dense hypercolumn matching for long-term visual localization. 2019.

[13] H. Germain, G. Bourmaud, and V. Lepetit. S2dnet: Learning accurate correspondences for sparse-to-dense feature matching. *arXiv preprint arXiv:2004.01673*, 2020.

[14] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[15] J. Hao, J. Dong, W. Wang, and T. Tan. Deepfirearm: Learning discriminative feature representation for fine-grained firearm retrieval. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 3335–3340. IEEE, 2018.

[16] J. Hao, J. Dong, W. Wang, and T. Tan. Deepfirearm: Learning discriminative feature representation for fine-grained firearm retrieval. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 3335–3340, 2018.

[17] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2599–2606. IEEE, 2009.

[18] M. Larsson, E. Stenborg, L. Hammarstrand, M. Pollefeys, T. Sattler, and F. Kahl. A cross-season correspondence dataset for robust semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[19] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[20] W. Maddern, G. Pascoe, C. Linegar, and P. Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017.

[21] S. Middelberg, T. Sattler, O. Untzelmann, and L. Kobbelt. Scalable 6-dof localization on mobile devices. In *European conference on computer vision*, pages 268–283. Springer, 2014.

[22] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4004–4012, 2016.

[23] Y. Ono, E. Trulls, P. Fua, and K. M. Yi. Lf-net: Learning local features from images. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 6234–6244. Curran Associates, Inc., 2018.

[24] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. 2015.

[25] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12716–12725, 2019.

[26] P.-E. Sarlin, F. Debraine, M. Dymczyk, R. Siegwart, and C. Cadena. Leveraging deep visual descriptors for hierarchical efficient localization. *arXiv preprint arXiv:1809.01019*, 2018.

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

[28] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. 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.

[29] T. Schmidt, R. Newcombe, and D. Fox. Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, 2016.

[30] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[31] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 761–769, 2016.

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[33] L. Svärm, O. Enqvist, F. Kahl, and M. Oskarsson. City-scale localization for cameras with known vertical direction. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1455–1461, 2016.

[34] L. von Stumberg, P. Wenzel, Q. Khan, and D. Cremers. Gn-net: The gauss-newton loss for multi-weather relocalization. *IEEE Robotics and Automation Letters*, 5(2):890–897, 2020.

[35] P. Wohlhart and V. Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3109–3118, 2015.

[36] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl. Sampling matters in deep embedding learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2840–2848, 2017.

[37] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl. Sampling matters in deep embedding learning. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

.1. Feature-PnP local evaluation results

To validate the effectiveness and improve the performance of our Feature-PnP module, we evaluate it with different image gradient operators and different loss functions in the local validation setting as stated in Section 4.3. The results are summarized in Table 4 where we use the mean number of inliers, mean rotation error (rerror), mean translation error (terror), and percentage of images in the coarse-precision, medium-precision, and high-precision introduced in the [28] as evaluation metrics. The results indicate that using second-order accurate central differences is crucial for the performance while different loss functions don't make much difference.

.2. A suspicious bug in S2DHM

There is a bug in the public S2DHM repository where the reference image's camera parameters (the intrinsic matrix and distortion coefficients) are used for RANSAC-PnP to estimate the pose of the query image. The correct implementation should be using the query image's camera parameters. This bug is confirmed by the author of S2DHM[4]. However, with this bug fixed we find the performance on two datasets degrades a lot, as shown in Table 5. This bug is still puzzling us and the authors.

.3. Detailed results for all conditions

The detailed recall accuracy under different conditions of all experiments mentioned in the paper is shown in the Table 6 and Table 7.

[4] https://github.com/germain-hug/S2DHM/issues/6

	sobel + sq	*np + cauchy*	*np + huber*	*np + sq*	*sobel + cauchy*	*sobel + huber*	*sobel + gm*	*np + gm*
mean rPnP num inliers	571.2	571.2	571.2	571.2	571.2	571.2	571.2	571.2
mean rPnP rerror	0.177	0.177	0.177	0.177	0.177	0.177	0.177	0.177
mean rPnP terror	1.180	1.180	1.180	1.180	1.180	1.180	1.180	1.180
mean fPnP num inliers	572.0	573.3	573.3	573.3	572.0	572.0	572.0	573.3
mean fPnP rerror	0.156	0.112	0.111	0.112	0.155	0.155	0.155	0.111
mean fPnP terror	1.045	0.779	0.768	0.769	1.047	1.045	1.045	0.773
rPnP high prec	12.0	12.0	12.0	12.0	12.0	12.0	12.0	12.0
fPnP high prec	13.6	24.4	25.2	25.0	13.8	13.4	13.8	24.4
rPnP medium prec	31.4	31.4	31.4	31.4	31.4	31.4	31.4	31.4
fPnP medium prec	35.0	48.2	46.8	46.8	34.6	35.0	34.6	48.0
rPnP coarse prec	99.4	99.4	99.4	99.4	99.4	99.4	99.4	99.4
fPnP coarse prec	99.6	100.0	100.0	100.0	99.6	99.6	99.6	100.0

Table 4. **Local validation of Feature-PnP**. *sobel* refers to the method using Sobel filter for image gradient while *np* refers to the method using second-order accurate central differences for image gradient. *sq*, *cauchy*, *huber*, and *gm* refers to square loss, Cauchy loss, Huber loss, and Geman-McClure loss respectively[4].

	RobotCar Seasons		*Extended CMU Seasons*		
Method	*Day-All*	*Night-All*	*Urban*	*Suburban*	*Park*
m	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5
deg	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10
Reference's cam params + rPnP	46.1 / 77.5 / 95.1	30.1 / 70.2 / 94.5	47.2 / 71.5 / 93.3	39.3 / 63.8 / 91.5	26.0 / 47.5 / 80.2
Reference's cam params + fPnP	48.0 / 77.9 / 95.1	32.3 / 71.1 / 94.5	50.6 / 73.9 / 93.5	42.6 / 66.0 / 91.7	29.0 / 49.7 / 80.4
Query's cam params + rPnP	46.9 / 77.5 / 95.1	21.5 / 61.3 / 94.5	3.3 / 11.9 / 87.0	2.0 / 9.6 / 87.6	1.8 / 8.5 / 73.9
Query's cam params + fPnP	48.1 / 78.0 / 95.1	22.9 / 63.7 / 94.5	33.0 / 53.5 / 90.0	26.7 / 46.8 / 88.7	17.1 / 32.9 / 75.8

Table 5. **Results with different camera parameters**. The result using reference images' camera parameters is significantly better than the one using query images', especially on the Extended CMU Seasons dataset where the image is distorted.

RobotCar Seasons	*night conditions*		*day conditions*						
weather	*night rain*	*night*	*overcast winter*	*sun*	*rain*	*snow*	*dawn*	*dusk*	*overcast summer*
m	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5
deg	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10
Original + fPnP	36.8 / 76.8 / 95.5	27.9 / 65.3 / 93.6	41.8 / 76.4 / 96.2	40.9 / 70.9 / 90.7	60.8 / 84.3 / 96.9	51.5 / 82.0 / 95.7	49.1 / 77.2 / 93.8	55.8 / 80.5 / 95.9	36.9 / 74.7 / 96.8
DMC + GN + fPnP (all)	35.5 / 73.2 / 94.5	24.4 / 58.0 / 93.2	45.9 / 77.7 / 96.2	43.7 / 71.7 / 90.9	61.8 / 83.1 / 96.9	54.2 / 84.3 / 95.7	49.3 / 78.9 / 94.0	55.8 / 81.2 / 95.9	38.0 / 75.4 / 96.8
DMC + GN + fPNP	41.1 / 75.9 / 95.5	26.5 / 66.4 / 93.6	44.6 / 77.7 / 96.2	43.9 / 70.2 / 90.9	60.8 / 83.4 / 96.9	53.4 / 83.2 / 95.7	47.8 / 76.2 / 93.8	53.0 / 80.2 / 95.9	34.6 / 75.6 / 96.8
DMC + fPnP (all)	37.3 / 74.3 / 94.8	23.7 / 58.9 / 91.8	46.2 / 77.2 / 96.2	41.5 / 72.4 / 90.9	61.5 / 83.1 / 96.9	53.6 / 84.3 / 95.7	48.2 / 78.3 / 94.0	55.6 / 81.2 / 95.9	38.4 / 75.4 / 97.0
Train from scratch + fPnP (all)	3.4 / 11.8 / 22.0	8.9 / 25.8 / 62.6	39.0 / 73.8 / 95.9	30.7 / 62.0 / 90.7	58.9 / 83.6 / 96.9	44.8 / 80.8 / 95.3	41.6 / 73.3 / 92.8	50.8 / 79.2 / 95.9	28.5 / 67.8 / 95.2
DMC + fPnP	39.3 / 77.7 / 95.2	22.4 / 64.8 / 93.6	45.1 / 77.4 / 96.2	44.6 / 72.0 / 90.9	60.3 / 82.9 / 96.9	53.2 / 83.4 / 95.7	45.3 / 76.4 / 94.0	54.6 / 80.7 / 95.9	37.1 / 75.4 / 96.5
Original + rPnP	35.0 / 76.1 / 95.5	25.1 / 64.2 / 93.6	41.8 / 75.6 / 96.2	39.8 / 69.8 / 90.7	59.1 / 84.1 / 96.9	50.9 / 82.0 / 95.7	44.9 / 76.8 / 93.8	53.8 / 80.7 / 95.9	33.7 / 74.1 / 96.8
DMC + GN + rPnP (all)	33.4 / 71.1 / 94.5	21.2 / 57.1 / 92.5	45.4 / 78.5 / 96.2	40.2 / 72.4 / 90.9	60.1 / 84.1 / 96.9	51.5 / 83.8 / 95.7	48.7 / 79.1 / 94.0	55.8 / 82.0 / 95.9	37.8 / 74.7 / 96.8
DMC + GN + rPNP	36.1 / 75.9 / 95.2	23.1 / 63.5 / 93.6	43.3 / 77.4 / 96.2	42.6 / 70.4 / 90.9	60.3 / 84.6 / 96.9	53.6 / 82.8 / 95.7	49.3 / 74.9 / 94.0	53.0 / 79.2 / 95.7	33.7 / 75.2 / 96.5
DMC + rPnP (all)	35.0 / 71.4 / 94.5	19.6 / 57.5 / 91.6	43.8 / 77.9 / 96.2	41.1 / 71.7 / 90.9	59.4 / 83.8 / 96.9	51.9 / 83.8 / 95.7	47.2 / 78.3 / 93.8	55.8 / 82.2 / 95.9	36.3 / 74.7 / 96.8
Train from scratch + rPnP (all)	2.0 / 10.7 / 21.8	8.4 / 23.7 / 61.9	38.2 / 71.5 / 95.9	30.4 / 60.7 / 90.4	57.5 / 84.1 / 96.9	42.7 / 80.0 / 95.5	40.6 / 72.9 / 92.5	48.0 / 77.9 / 95.9	29.4 / 68.0 / 95.7
DMC + rPnP	36.6 / 76.4 / 95.2	21.2 / 62.6 / 94.1	44.4 / 77.4 / 96.2	41.1 / 72.4 / 90.9	57.5 / 83.8 / 96.9	52.6 / 83.4 / 95.7	45.1 / 76.0 / 94.0	54.1 / 80.7 / 95.9	37.4 / 75.6 / 96.5

Table 6. Detailed results for all conditions on the RobotCar Seasons dataset.

	Extended CMU Seasons Dataset										
scene	*urban*	*suburban*	*park*	*overcast*	*sunny*	*foliage*	*mixed foliage*	*no foliage*	*low sun*	*cloudy*	*snow*
m	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5	0.25 / 0.5 / 5
deg	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10	2 / 5 / 10
Original + fPnP	50.6 / 73.9 / 93.5	42.6 / 66.0 / 91.7	29.0 / 49.7 / 80.4	41.7 / 64.2 / 88.7	37.9 / 59.4 / 85.1	38.8 / 60.6 / 86.1	41.4 / 64.2 / 89.7	46.3 / 70.2 / 94.3	41.7 / 64.8 / 90.8	45.0 / 68.2 / 91.5	38.1 / 61.7 / 90.8
Original + rPnP	47.2 / 71.5 / 93.3	39.3 / 63.8 / 91.5	26.0 / 47.5 / 80.2	38.3 / 62.0 / 88.6	34.7 / 57.2 / 84.8	35.7 / 58.4 / 85.8	37.9 / 61.9 / 89.4	43.4 / 68.1 / 94.2	38.6 / 62.6 / 90.6	41.6 / 65.6 / 91.2	34.9 / 59.1 / 90.6

Table 7. Detailed results for all conditions on the Extended CMU Seasons dataset.