Towards Visual Localization Interoperability: Cross-Feature for Collaborative Visual Localization and Mapping

Alberto Jaenal*[1] Paula Carbó Cubero*[1,2] José Araujo[1] André Mateus[1]

[1]Ericsson Research, Sweden [2]KTH Royal Institute of Technology

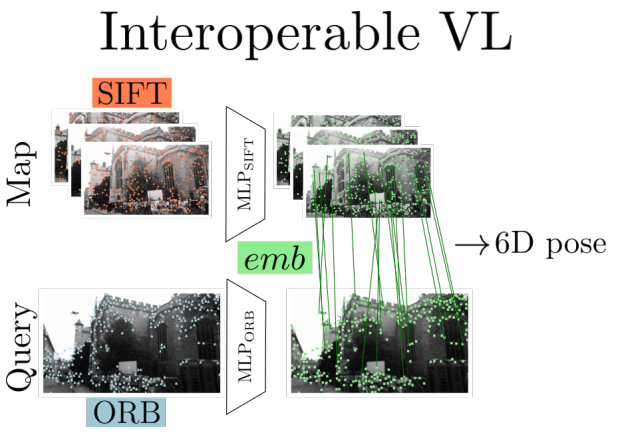
Abstract

The growing presence of vision-based systems in the physical world comes with a major requirement: highly accurate estimation of the pose, a task typically addressed through methods based on local features. The totality of the available feature-based localization solutions are designed under the assumption of using the same feature for mapping and localization. However, as the implementation provided by each vendor is based on heterogeneous feature extraction algorithms, collaboration between different devices is not straightforward or even not possible. Although there are some alternatives, such as re-extracting the features or reconstructing the image from them, these are impractical or costly to implement in a real pipeline. To overcome this, and inspired in the seminal work Cross-Descriptor [13], we propose Cross-Feature, a method that applies a patch-based training strategy to a simple MLP which projects features to a common embedded space. As a consequence, our proposal allows to establish suitable correspondences between features computed through heterogeneous algorithms, e.g. *SIFT [25] and SuperPoint [10]. We experimentally demonstrate the validity of Cross-Feature by evaluating it in tasks as Image Matching, Visual Localization and a new Collaborative Visual Localization and Mapping scenario. We believe this is the first step towards full Visual Localization interoperability. Code is available at* https://github.com/EricssonResearch/crossfeat.

1. Introduction

Nowadays, Visual Systems have become mature enough to bridge the gap between reality and the digital world. This is evidenced by the fact that they play a crucial role in complex applications such as autonomous robotics, extended reality or self-driving vehicles. There exist a plethora of commercialized products that, to fulfill their purpose, estimate their position in the environment by means of visual sensors, typically using local features due to their accuracy, reduced cost and reliability [29, 49]. Nevertheless, collaboration between devices requires them to operate within an unified coordinate system, which involves sharing maps and determining their relative poses [15]. Unfortunately, most solutions for Visual Localization (VL) are based on proprietary or specific visual pipelines that are not compatible, making interoperability (see Fig. 1a), the ability for devices from different vendors to work together, challenging to achieve.

There are some potential solutions for VL interoperability, but they struggle to meet the same requirements than their homogeneous counterparts. Sharing images across devices is generally avoided due to serious privacy concerns and increased bandwidth usage. Consequently, interoper-



(a) To enable matching heterogeneous features, *e.g.*, SIFT and ORB, we use a simple MLP that projects them to a common embedding space emb.

(b) Our patch-based training procedure increases the matchability of the projections in case of heterogeneous features, which overcomes the limitation of using the same detector from Cross-Descriptor [13].

Figure 1. In this work, we pursue improving the accuracy of Interoperable feature-based Visual Localization.

*Both authors contributed equally.

ability should occur after the devices transmit their features. Implementing multiple feature algorithms in the same device is unrealistic, as it adds computational strain and vendors may not willingly disclose their proprietary information. A costly alternative is to set an edge/cloud unit that reconstructs the image from local features as in [32]. Similar challenges arise when using cloud services (*e.g.* Google's Visual Positioning System [34], Microsoft's Azure Spatial Anchors [20], or Niantic's Lightship [52]), where maps may be constructed with different or legacy features. In this case, re-generating and maintaining separate maps of the same scene for each local feature is expensive and inefficient.

This paper deals with the challenge of enabling matching between heterogeneous local features for VL interoperability, assuming that no images are stored in the maps and features cannot be re-extracted. To the best of our knowledge, Cross-Descriptor [13] is the only work that addresses this task, by projecting heterogeneous descriptors to a common representation. Despite its well-defined set up and positive results, this approach has a main assumption that considerably reduces its applicability: that all features are extracted using an homogeneous detector, i.e., the **same keypoint**.

Overcoming this limitation (see Fig. 1b), we propose what, to the best of our knowledge, is the first solution for the Cross-Feature scenario. Our approach consists of a novel training procedure that makes use of patch overlap to address the disparity between different keypoint extractors. We apply this procedure to a lightweight encoder MLP (as in Cross-Descriptor) that maps features into a joint embedded space. This allows to establish valid correspondences between heterogeneous features, for example, matching SIFT and ORB (see Fig. 1), which is especially relevant for *Collaborative Localization and Mapping*. We train our method for four well-known feature algorithms (SIFT [25], ORB [39], SuperPoint [10] and DISK [58]), and evaluate it in Cross-Feature Image Matching and Visual Localization tasks over different datasets, namely Aachen Day & Night [48], Cambridge Landmarks [21] and 7scenes [50]. Additionally, we evaluate our method in a novel Collaborative Localization and Mapping scenario where multiple devices with heterogeneous feature algorithms co-exist in the same coordinate system. In short, besides overcoming the limitation of using an homogeneous detector from Cross-Descriptor, we experimentally demonstrate Cross-Feature to establish a viable first step towards interoperability.

2. Related work

Local features Local features are an essential part of several computer vision tasks such as Structure-from-Motion [49], SLAM [29] or Visual Localization [40]. In a nutshell, a local feature is composed by a keypoint, the location of a point of interest; and a descriptor, a compact vector that encodes the visual appearance of the patch around the keypoint. For more than two decades, feature extraction has been addressed through handcrafted pipelines, based on a *detect-then-describe* pipeline. Each local feature searches for different keypoints: SIFT [25] and SURF [4] detect blobs, ORB [39] and BRISK [23] detect corners. Handcrafted descriptors also vary in their nature, as gradient-based SIFT and SURF ensure robustness, while BRIEF [5], ORB or BRISK employ binary features for higher efficiency. In the last years, the community has shown growing interested in learning-based pipelines that reformulate the whole feature extraction process to improve different aspects. Some approaches focus on learning a robust descriptor given a set of precomputed keypoints (typically extracted through handcrafted methods) [28, 54, 55], or just densely over the image [16, 53]. Contrarily, other authors have addressed the process holistically, also formulating the keypoint detection in the learning process [10, 14, 36, 58]. Each feature offers different advantages (there exist examples of fine-tuning applications depending on the feature [17]), but they exhibit a high variability degree, as the keypoint locations and the visual information embodied in the descriptors vary substantially per approach.

Visual localization and mapping Visual Localization [26, 44, 56] is the task of estimating the position and orientation of a camera, given a previously build 3D map of the environment. The task is traditionally solved through local features, given their efficiency and the well-established theory of multi-view geometry. The task has been typically addressed the task through 2D-to-3D matching [6, 43, 44, 46, 56], but it becomes inefficient in case of large environments and under appearance variations [47]. Recently, the authors of [30] proposed a simple local-global feature aggregation over the 2D-to-3D matching to overcome these problems. Hierarchical pipelines [31, 40] also improve accuracy and efficiency in localization, by first retrieving similar database images using global descriptors [1, 35, 57] and then establish image-to-image correspondences. Other VL works have focused on improving efficiency by learning descriptor compression [11] or quantizing the map [22]. Geometry-only methods [59, 62] localize through geometric cues only but, despite reducing privacy concerns and computational load, their performance is still far from the visual descriptor-based approaches.

Interoperability and feature translation The most extended assumption for localization is to design a solution assuming homogeneity in the sensor and the platform. Some approaches have addressed different settings, as assuming cross-sensor localization between camera and LiDAR [7, 8], or cross-platform Visual Localization [37] by characterizing the different scenes using geometry based on planes. In [19] the authors propose an autoencoder that enables cross-global feature Image Retrieval. The only approach addressing heterogeneity in the local fea-

tures is Cross-Descriptor [13], which presents several VL scenarios for their proposed descriptor translation. However, the assumption of using the same keypoint is a limitation that avoids its application in realistic interoperable VL pipelines (as can be seen in section Sec. 5.2). Previous research [9] addresses interoperability through specific detector-descriptor augmentation. Yet another feasible solution for cross-feature VL would to apply geometry-only methods [59, 62]. However, their limited performance in the homogeneous scenario suggests even worse results in the heterogeneous feature scenario (see Tab. 1).

3. Problem definition

Consider a feature algorithm A that, given an image $\mathcal{I} \in \mathbb{R}^{H \times W \times C}$, extracts a set of N features $\mathbf{f}^A$, composed by a keypoint $\mathbf{k}^A = (k_u^A, k_v^A) \in \mathbb{R}^2$ and a descriptor $\mathbf{d}^A \in \mathbb{R}^D$:

$$A(\mathcal{I}) = \{\mathbf{f}_i^A\}^N = \{(\mathbf{k}_i^A, \mathbf{d}_i^A)\}^N, \tag{1}$$

with H, W and C respectively the height, width and channels of the image, and D the dimension of the descriptor.

We address *Cross-Feature translation* for two feature extraction algorithms A, B through a specific function t_A, t_B that projects their corresponding descriptors to a shared embedded space with dimension D_{emb}, for example:

$$t_A(\{\mathbf{d}_i^A\}) = \{\mathbf{d}_i^{emb,A}\}, \tag{2}$$

such that embeddings $\{\mathbf{d}_i^{emb,A}\}$ and $\{\mathbf{d}_i^{emb,B}\}$ are comparable and valid correspondences can be set between them. We define the combination of the original keypoints and the projected embeddings as projected features $\{\mathbf{f}_i^{emb,A}\}^N = \{(\mathbf{k}_i^A, \mathbf{d}_i^{emb,A})\}^N$.

Correspondences between features extracted homogeneously, i.e., with the same algorithm, are straightforward to define as they are one-to-one, and they can be easily mapped between different perspectives through reprojection. However, trying to set correspondences between two heterogeneous features is not simple, as keypoints are distributed differently across the image, see Fig. 2. We call this the *heterogeneous correspondence problem*, which raises two main problems, as discussed in [13]: information loss when keypoints do not overlap in parts of the image, and the one-to-many associations between near heterogeneous features.

4. Training method

This section describes our proposed method, addressing the Cross-Feature problem through a patch-based technique that solves the heterogeneous correspondence problem. We describe the translation architecture, the training method, the loss functions and the implementation details.

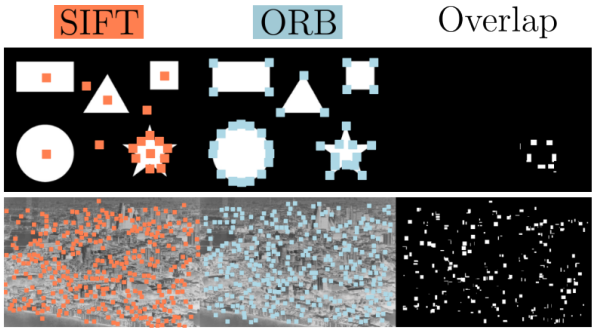


Figure 2. Example of the feature maps used for training the Cross-Feature approach for two images of different size and visual content. In the first two columns we can see the patches with $T_w = 15$ over the feature positions extracted by two different algorihtms (SIFT and ORB). The last column displays the resulting overlap between the two feature maps. So, even for a patch of considerable size, the overlap between heterogeneous features can be small, depending on the image and the keypoint distribution.

4.1. Encoder architecture

Inspired by the simplicity of Cross-Descriptor [13], we address the projection function to the embedded space t_A as a 2-layer Multi-Layer Perceptron encoder MLP$_A$. Unfortunately, due to the heterogeneous correspondence problem, it is not trivial how to define a decoder architecture as in [13]. Consequently, we just focused on the encoder.

4.2. Training on feature maps

The goal of the training procedure is to enforce similarity between embedded features from heterogeneous algorithms whose keypoints are close and push them apart otherwise. Since heterogeneous features can only be related by their keypoint positions, we propose to generate associations through feature maps, whose overlapping regions (see Fig. 2) generate the required correspondences for training.

Consider two images taken at the same scene $\mathcal{I}_1$ and $\mathcal{I}_2$, with $U_{1\to2} \in \mathbb{R}^{H \times W \times 2}$ denoting the dense ground-truth correspondence map between them. This correspondence map can be computed using optical flow, stereo matching or synthetic transformations, *e.g.*, homography or reprojection. Thus, if pixel $p_1 = (u_1, v_1)$ in $\mathcal{I}_1$ corresponds to pixel $p_2 = (u_2, v_2)$ in $\mathcal{I}_2$, then $U_{1\to2}(u_1, v_1) = (u_2, v_2)$.

Given the image I_k and the algorithm A with its corresponding encoder MLP$_A$, we define a feature map as $\mathbf{F}_{k,A} \in \mathbb{R}^{H \times W \times D_{emb}}$, that we handcraft to store the projection of the features to the embedded space $\{\mathbf{f}_i^{emb,A}\}$. A straightforward approach to construct the feature map would follow [32], placing each embedded descriptor $\mathbf{d}_i^{emb,A}$ at the 2D location where its corresponding keypoint $\mathbf{k}_i^A$ is present. However, the resulting overlap between feature maps computed with two different detectors will be minimal or even null (see Fig. 2), making it impossible to learn any valid Cross-Feature correspon-

dence. To increase the overlap, we set a patch of size T_w centered at the 2D location of the feature (see Fig. 2). Thus, for the i-th feature, we place its embedded descriptor $\mathbf{d}_i^{emb,A}$ in all the positions of a patch around its keypoint $[\mathbf{k}_{u,j}^A - T_w/2 \dots \mathbf{k}_{u,j}^A + T_w/2] \times [\mathbf{k}_{v,j}^A - T_w/2 \dots \mathbf{k}_{v,i}^A - T_w/2]$ on the feature map. However, when multiple adjacent features appear in the same feature map, their patches might overlap, leading to feature map positions filled with two or more different embedded features. We considered different aggregation methods to solve this issue, but we finally opted to deal with this through a last-served method, which we will discuss in the Computational cost subsection. Feature maps are initialized with a null value $\emptyset$, and then subsequently filled for all projected features. Note that, as features tend to be distributed unequally in the image, a high part of the positions in the feature map will be empty (see in Fig. 2), which will be uninformative during training.

4.3. Loss functions

To train our encoder, we chose a differentiable approximation of Average Precision (AP) [18] as the loss function, as it is a global metric that optimizes local descriptor learning for Nearest Neighbor matching. We found that this loss function was the only one that rendered some performance, compared with other local ranking losses (*e.g.*, triplet loss [2, 54, 55]) which were not sufficiently informative to make the model learn, since they depend on the samples chosen per batch. The loss $\mathcal{L}_{i\to j}$ is computed for a pair of feature maps $\mathbf{F}_i$ and $\mathbf{F}_j$ and the ground-truth correspondence map relating them $U_{i\to j}$. We define as $V_{i,j}$ the set of query pixel positions (u, v) in $\mathbf{F}_i$ that are non empty, and whose corresponding position within a threshold in $\mathbf{F}_j$ (using $U_{i\to j}$) has a valid embedded descriptor, i.e., is not empty. These correspondences are defined as positives, where we enforce similarity between the feature projections $\mathbf{f}_k^{emb,X}$ present in those positions. Similarly, for each query, we define the negative matches as all the non-empty pixel positions in $\mathbf{F}_j$ that do not hold a correspondence with the query, whose corresponding pixel is farther than a threshold. Similarly to [18], the AP for each query position $AP_{i\to j}(u, v)$ is computed by ranking the union of positive and negative matches in terms of the distance. The final loss is computed as:

$$\mathcal{L}_{i\to j} = \frac{1}{|V_{i,j}|} \sum_{(u,v)\in V_{i,j}} 1 - AP_{i\to j}(u, v). \quad (3)$$

We will describe the losses that arise for the minimal set: a pair of images $\mathcal{I}_1$ and $\mathcal{I}_2$, and a pair of extracting algorithms A and B. This way, three different valid combinations for the loss arise (the remaining combination $\mathcal{L}^{SS}$ is dropped because it is trivial and uninformative):

- Different image, same algorithm (*e.g.*, $\mathcal{L}^{DS}_{1,A\to 2,A}$): maximizes matchability between feature projections from an homogeneous algorithm, increasing the projection robustness to appearance variations due to perspective changes.
- Same image, different algorithm (*e.g.*, $\mathcal{L}^{SD}_{1,A\to 1,B}$): this is the simplest case for the Cross-Feature, as it enforces similarity for overlapping embedded projections from heterogeneous features on the same image. Note that the correspondence map is the identity $U_{i\to i}(u, v) = (u, v)$.
- Different image, different algorithm (*e.g.*, $\mathcal{L}^{DD}_{1,A\to 2,B}$): this is the most complex and representative case for the Cross-Feature, as it enforces overlapping projections from heterogeneous algorithms under perspective changes to be similar.

Each loss type is the average of its four variants, e.g.:

$$\mathcal{L}^{DS} = \frac{1}{4}(\mathcal{L}^{DS}_{1,A\to 2,A} + \mathcal{L}^{DS}_{2,A\to 1,A} + \mathcal{L}^{DS}_{1,B\to 2,B} + \mathcal{L}^{DS}_{2,B\to 1,B}), \quad (4)$$

and the final loss is the sum of the three averaged losses:

$$\mathcal{L} = \mathcal{L}^{DS} + \mathcal{L}^{SD} + \mathcal{L}^{DD} \quad (5)$$

4.4. Implementation details

Training data and sampling. We replicate the data and sampling conditions from the ones released in R2D2 [36]. The dataset consists on randomly transformed pairs from the Aachen Day-Night Dataset [45, 48], extracted with the help of SfM, from where dense correspondences were estimated using optical flow. It also contains distractors from the Oxford-Paris Revisited dataset [33]. For the sampling, images are randomly cropped to a size of 192x192, and we consider pixels on a regular grid with a step of 8 pixels. For further information, we refer to [36].

Features. For training and validating Cross-Feature, we employed four well-known handcrafted and learned features: SIFT [25] (using the OpenCV extractor and the kornia descriptor [38]), ORB [39] (using the implementation from ORB-SLAM2 [29]), the open-source implementation of SuperPoint [10] and the release version of DISK [58].

Training parameters. Training took 30 epochs, using Adam optimizer, batch size of 5, a learning rate of 0.001 and weight decay of 0.0005. We define the positive and negative thresholds for correspondences to be 4 and 8 respectively, and we set the full patch window size to $T_w = 15$ pixels (radius of 7). Training each encoder took $\sim$ 70 hours in a desktop computer equipped with NVIDIA GeForce RTX 4090 and a 13th Gen Intel(R) Core(TM) i9-13900K. The MLP encoder has 2 hidden layers of size 1024 for all features, using ReLU activation followed by batch normalization after each linear layer except the last. Similarly to [13], we set the shared embedding space dimension to $D_{emb} = 128$, and it is l_2 normalized. During training, we observed that each patch contained by average $\sim$ 300 features.

Runtime. The simplicity and compactness of the MLP architecture makes the Cross-Feature suitable for real-time in-

ference, taking 0.35 and 25 ms to project 4k features to the embedded space in GPU and CPU, respectively.

Computational cost. However, despite the lightweightness of the encoder, the computational cost derived from the training is substantially high. The reasons for this arise from the density of the feature maps and the wide variety of loss functions that need to be defined per extractor algorithm and per image. This has a quadratic cost and limits the training scalability of this approach. In our experiments, we observed that training for more than a pair of extractors was not computationally affordable, so we limited each training to two algorithms. As we run our tests for four different algorithms, we found two ways to proceed. First option is to run a specific training per each pair of algorithms, so the learned embedding space is dedicated for each combination. This gives better results per algorithm pair, but requires using different sets of weights depending on the pair (i.e., the SIFT encoder weights will not be the same when matching with ORB and DISK). This does not allow for multi-algorithm collaboration and, ultimately, for Visual Localization Interoperability. The second option is to reach a common embedding space for the four features in a suboptimal manner, by first training the encoders for an algorithm pair and then train the remaining algorithm encoders by freezing the weights of one of the encoders from the first training. This reduces the performance for the second-stage pairs, but we opted for it, since is the approach that more closely aligns with a real shared embedding space for all features. For this, we first trained the SIFT-SuperPoint encoders, and then we froze the SIFT encoder weights to train it separately with ORB and DISK. We found empirically that, due to the many-to-many associations arising from multiple adjacent heterogeneous features, using aggregation methods (e.g. average) between overlapping patches in the same feature map made the training process impractical, as it substantially increased the number of gradients and the training time ($30-50\times$). Consequently, we decided for a extreme simplification: set a single embedding feature in each feature map position, in a last-served basis. This drastic simplification solves the computational cost problem by substantially reducing the amount of available information, but we found it functional as the many-to-many associations are represented across different batches.

5. Experimental results

To experimentally assess the validity of our method to set valid Cross-Feature correspondences, we evaluate three different tasks: Image Matching, Visual Localization and a newly proposed Collaborative Visual Localization and Mapping scenario.

Results computed using feature embeddings projected to the common shared space with our method are named CF^{emb}. The main baseline to compare our method with is Cross-Descriptor [13], which projects descriptors to a common embedded space. We denote it as CD^{emb}. This approach was retrained for this work, using DoG from OpenCV as detector, and SIFT, SuperPoint and BRIEF descriptors (we found DISK descriptor more difficult to decouple from the available architecture, so we evaluated this representative subset). As BRIEF is a descriptor-only algorithm, we combined it with different detectors, for example DoG from SIFT (denoted as DoG:BRIEF in the experiments) or Oriented FAST from ORB (denoted as OrFAST:BRIEF). Note that, despite Cross-Descriptor approach was trained combining DoG with different descriptors, in the heterogeneous evaluations we followed a more realistic cross-feature approach, so we used the original setup for each feature (DoG with SIFT, Oriented FAST with ORB, full SuperPoint network) . Similarly to our approach, we set the encoder MLP with 2 hidden layers of 1024 for all descriptors and trained it for 30 epochs, with the remaining parameters as described in their paper.

5.1. Image Matching

To evaluate the potential of the embedded features to be identified as correspondences, we measure the raw matching performance on the HPatches [3] benchmark. We use the test split, consisting of a total of 116 sequences of 6 images recorded in different scenarios, half of them under photometric changes and half under perspective changes. We excluded 8 sequences [12] from the dataset. We measure the Mean Matching Accuracy (MMA) [27] (mean percentage of correctly matched points), and the number of inliers. The reprojection threshold error is varied between 1 and 5 pixels, using mutual Nearest Neighbors to search for matches. When using heterogeneous features, the map feature is applied on the first image of the sequence, while the query algorithm is applied on the remaining five. We use SIFT and SuperPoint as map features, and, to limit the amount of results, both plus ORB as query. We separate the matching performance obtained from the same or from heterogeneous feature extractors. We use the standard non-translation case for the map features as baseline, denoted as Direct; and we follow the naming convention map $-$ query for the heterogeneous feature cases.

Matching performance, shown in Sec. 5.1, demonstrates a high difference when the features projected to the embedding space are generated from the same or from heterogeneous feature extractors. In the first case, translating descriptors to the embedding space reduces the overall matchability compared to the standard approach, with our method improving compared to Cross-Descriptor, probably since the $\mathcal{L}^{DS}$ loss specifically enforces similarity between different perspectives. On the other hand, we can observe that Cross-Descriptor demonstrates poor performance when applied to different features, as it was designed to work un-

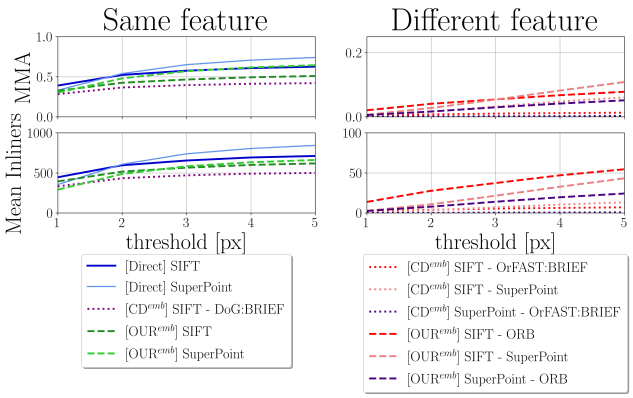


Figure 3. Performance on the HPatches sequences dataset. We average results on MMA and number of inliers for all the sequences, comparing embeddingswhen using homogeneous features (left) than heterogeneous ones (right). Note the difference in scale.

der the same keypoint assumption. The results show that our method is able to produce a small but consistent set of heterogeneous correct matches with a lower difference in keypoint position throughout the different 3D views, a promising result. However, it also generates a high number of outliers in the heterogeneous scenario, a concern to be addressed by future work. Matching SIFT and SuperPoint results in the best performance, presumably due to a higher overlap between their keypoints.

5.2. Visual Localization

We evaluated our proposed method on Visual Localization in three different datasets capturing different environments:

- **Aachen Day & Night** [48], which captures outdoor images from the historical center of Aachen. The dataset consists of ∼4.3k images captured at day for the database, and 922 for evaluation, counting 824 for day and 98 for night. We retriangulate the maps by using ground-truth camera poses and 3D models by VisualSfM [61].
- **Cambridge Landmarks** [21], recorded outdoors with a smartphone at five historical buildings within the University of Cambridge. The train and the test splits are composed by different sets of walking trajectories around the landmarks. We retriangulate the maps by using ground-truth camera poses and 3D models by VisualSfM [61].
- **7Scenes** [50], depicting 7 different indoor scenes with a total of 46 sequences from which 18 were used for testing, as in the original test split.

We use HLoc [40] to create the map and for subsequent localization. We set a maximum of 4k features to be extracted each image. Maps for the outdoor datasets were created by matching each image with 20 co-visible images retrieved using NetVLAD [1], while in the case of 7scenes we employed the models provided by PixLoc [42]. Regarding the localization, for the outdoor datasets, we retrieved the 20 most similar database images with NetVLAD, while

Table 1. Evaluation on Aachen Day and Night benchmark [48] for the compared methods. We show results for homogeneous and heterogeneous algorithms, grouping them depending on the algorithm used for mapping. Best results for each homogeneous/heterogeneous case in bold.

	Method	Algorithm		% localized queries (0.25 m, 2°) Day	(0.5 m, 5°) Day	(5 m, 10°) Day	(0.25 m, 2°) Night	(0.5 m, 5°) Night	(5 m, 10°) Night
		Map	Query						
Homog.	Direct	SIFT	SIFT	**80.8**	**90.4**	**95.6**	**55.1**	**68.4**	**80.6**
	GoMatch [62]	SIFT	SIFT	3.6	6.6	16.0	0.0	0.5	3.7
	CD^{emb} [13]	SIFT	DoG:BRIEF	67.6	73.5	78.8	9.2	13.3	20.4
Hom.	Direct	SuperPoint	SuperPoint	**88.3**	**94.9**	**97.9**	**86.7**	**91.8**	**98.0**
	GoMatch [62]	SuperPoint	SuperPoint	2.9	4.7	15.5	0.5	0.5	9.9
Heterogeneous	GoMatch [62]	SIFT	SuperPoint	0.2	0.2	4.9	0.0	0.0	4.1
	Im. Reconstr. [32]	SIFT	SuperPoint	34.0	43.6	52.2	5.1	5.1	7.1
			ORB	30.5	37.4	45.6	0.0	1.0	1.0
			DISK	32.5	43.9	54.5	2.0	2.0	5.1
	CD^{emb} [13]	SIFT	SuperPoint	20.4	28.2	42.6	2.0	2.0	3.1
			OrFAST:BRIEF	4.1	5.8	12.9	0.0	0.0	1.0
	CF^{emb}	SIFT	SuperPoint	**56.3**	**66.4**	**78.6**	**17.3**	**20.4**	**29.6**
			ORB	31.3	36.7	47.2	3.1	3.1	6.1
			DISK	33.5	40.9	54.1	4.1	6.1	10.2
Heterogeneous	GoMatch [62]	SuperPoint	SIFT	0.2	0.2	4.9	0.0	0.0	4.1
	Im. Reconstr. [32]	SuperPoint	SIFT	**77.8**	**85.7**	**90.0**	**51.0**	**59.2**	**77.6**
			ORB	56.3	63.6	74.0	7.1	8.2	12.2
			DISK	60.9	70.6	78.5	16.3	19.4	32.7
	CD^{emb} [13]	Superpoint	SIFT	9.7	12.3	16.6	0.0	0.0	0.0
			OrFAST:BRIEF	1.0	1.0	1.3	0.0	0.0	0.0
	CF^{emb}	Superpoint	SIFT	43.1	49.3	59.2	6.1	6.1	7.1
			ORB	13.5	16.5	24.9	1.0	1.0	2.0
			DISK	45.1	53.2	65.4	8.2	12.2	14.3

for 7scenes we used the top-10 DenseVLAD [57] retrievals provided by PixLoc. To limit the number of experiments, we use SIFT and SuperPoint for creating the maps, while the four algorihtms can serve as query. Mutual Nearest Neighbor is the matcher for all cases, except when creating the map with SuperPoint, where we used SuperGlue [41].

Additionally, for Aachen, besides Cross-Descriptor, we used two more baselines to compare our method to. The first one is GoMatch [62], a geometry-only method that uses keypoint information to estimate matches between the image and the map. As it only uses the position of the keypoints for its embeddings, it can be applied to Visual Localization with heterogeneous detectors. The second baseline is a re-implementation of the image reconstruction network proposed in [32], a U-Net architecture followed by a CNN that outputs a reconstruction of the image given a feature map. We trained a specific reconstruction network for each feature with a discriminator network based on VGG16 [51]. For the VL task, we simulated the case of an intermediate edge unit processing the features, so the reconstruction network is applied to the query features. Once the query image is reconstructed, the map features can be extracted from it and subsequently perform homogeneous matching. Note that this approach requires high computational resources and its inference time is also high (more than 100 ms).

Results for the four baselines in Aachen can be seen in Tab. 1, comparing the methods depending on the algorithm used to create the map and whether the localization uses homogeneous either heterogeneous algorithms. Per-

formance in the heterogeneous case is still far from homogeneous. However, our method outperforms the remaining baselines in heterogeneous Visual Localization. Despite Cross-Descriptor encoders demonstrate good performance in the homogeneous detector case (SIFT vs DoG:BRIEF), it demonstrates low performance when the detector is different. Note that, for our method, results are not bidirectional for each pair of algorithms. This can be explained by the fact that the feature employed for mapping heavily influences the amount and the spatial distribution of the reconstructed 3D points, becoming especially biased towards points of interest reliable for that feature, which then impacts the localization. GoMatch reveals low performance in this dataset (we did not find any other work reporting its results), and it becomes even lower in the heterogeneous algorithm case. We believe that GoMatch poor performance is due to the reduced overlap between query and map images, as it is not a sequential dataset, and the challenging dynamic conditions. The image reconstruction achieves good results when applying SuperPoint over the reconstructed images, but its performance drops noticeably when applying SIFT, as the handcrafted descriptor is not able to perform so good in the presence of the reconstruction artifacts.

As our method was trained on images from Aachen, to evaluate its generalizability to other datasets, we compare its performance with Cross-Descriptor in Cambridge (Tab. 2) and 7scenes (Tab. 3). Cross-Descriptor follows the same trend than in Aachen dataset for the heterogeneous feature case, even failing to localize in some scenes. Our proposed method demonstrates to perform also in these different environments. Querying projected DISK features to a SuperPoint database reports the best results, which can be explained by the information contained in their learned visual descriptors being more useful to create valid Cross-Feature correspondences.

5.3. Collaborative Mapping and Localization

Extending the *Collaborative mapping* scenario proposed in [13], we devise a scenario in which multiple devices using heterogeneous feature algorithms coexist. In this scenario, a subset of devices will be in charge of the mapping (as they may have higher processing capacity), while a set of lightweight devices (*e.g.*, phones, XR devices) will query the existing map to obtain a precise pose estimate. As mentioned earlier, we assume that images are not stored in the map, so new features cannot be extracted.

This scenario is implemented in the Cambridge Landmarks [21] dataset. We first split the mapping sequences for each scene in two (note that ShopFacade is skipped since it has a single mapping sequence), re-triangulating a different submap for each mapping feature (i.e., SIFT and SuperPoint). To create a common global map, both partial submaps are just aligned using the ground-truth poses from VisualSfM [61], and all their descriptors are projected to the embedding space. We applied no further post-processing algorithm: the submap elements are just stacked. Despite other more elaborated techniques could be used for merging the 3D points from both submaps, we found that they do not apply for submaps created from heterogeneous features. The resulting map is queried with all features.

Results for this collaborative scenario are shown in Tab. 4, comparing our method to Cross-Descriptor. Cross-Descriptor achieves good performance when using the same features of the map, but its performance decreases when DoG:BRIEF is used (as some parts of the map are constructed with SuperPoint), and fails when another detector is used (i.e., OrFAST). We can see that our method allows for an effective embedded feature space that enables interoperability for mapping and localization, with all query methods obtaining similar results. When the scene is bigger, querying with mapping features (SIFT, SuperPoint) achieve better performance, as matching is homogeneous in part. We can see that performance is in general lower than Tab. 2, which is understandable as the map contains heterogeneous features and some parts of the scene may have duplicated information. However, as the map is the result of aligning two submaps without any merging technique, downstream performance and consistency cannot be ensured. For example, this map contains a higher number of 3D heterogeneous points than its homogeneous counterpart, which we believe that ease the localization compared with the single feature maps, as these describe diverse elements of the scene. We also think that this is what allows for similar performance between all the query features.

6. Conclusions and Future work

In this paper, we presented a novel approach to effectively deal with the problem of matching features from heterogeneous feature extraction algorithms. We propose a patch-based training strategy based on overlap that allows a compact and fast MLP encoder to project features so that they can be used to create valid Cross-Feature correspondences. This allows what, to the best of our knowledge, is the first approach performing Cross-Feature tasks, such as Image Matching and Visual Localization. We extensively evaluate our method in different benchmarks and demonstrate that it outperforms the baselines, which show limited performance. In addition, our Cross-Feature method allows us to propose a new Collaborative Localization and Mapping scenario where multiple devices may co-exist for mapping and localization, by using different algorithms and projecting their features to the common embedded space. We believe that our approach sets an initial step towards Interoperability in feature-based Visual Localization.

This works represents a first step towards a novel research topic: Cross-Feature matching, which has many

Table 2. Evaluation on Cambridge Landmarks [21] for the compared methods. We show results for homogeneous and heterogeneous algorithms, grouping them depending on the algorithm used for mapping. Best results in bold, N/A refers to unsuccessful localization.

	Method	Algorithm		Median error (m, °)				
		Map	Query	ShopFacade	KingsCollege	OldHospital	StMarysChurch	GreatCourt
Homog.	Direct	SIFT	SIFT	0.05, 0.23	0.15, 0.25	0.19, 0.31	0.10, 0.28	0.30, 0.14
	CD^{emb} [13]	SIFT	DoG:BRIEF	0.05, 0.22	0.15, 0.25	0.18, 0.29	0.10, 0.28	0.29, 0.14
	Direct	SuperPoint	SuperPoint	**0.04, 0.19**	**0.12, 0.21**	**0.16, 0.30**	**0.07, 0.21**	**0.16, 0.11**
Heterogeneous	CD^{emb} [13]	SIFT	SuperPoint	0.09, 0.34	0.18, 0.32	0.32, 0.57	0.15, 0.46	0.96, 0.48
			OrFAST:BRIEF	N/A	N/A	N/A	N/A	N/A
	CF^{emb}	SIFT	SuperPoint	**0.06, 0.25**	**0.15, 0.25**	**0.25, 0.38**	0.11, **0.32**	**0.43, 0.23**
			ORB	**0.06**, 0.27	**0.15, 0.25**	0.26, 0.41	**0.10, 0.32**	0.61, 0.29
			DISK	0.07, 0.29	0.18, 0.28	0.50, 0.72	0.17, 0.59	2.60, 1.68
	CD^{emb} [13]	Superpoint	SIFT	0.24, 1.12	0.25, 0.37	0.91, 1.18	0.18, 0.59	1.30, 0.77
			OrFAST:BRIEF	N/A	N/A	N/A	N/A	N/A
	CF^{emb}	Superpoint	SIFT	**0.05, 0.26**	**0.16, 0.27**	0.38, 0.59	**0.10, 0.32**	**0.56**, 0.33
			ORB	0.10, 0.52	0.20, 0.32	0.57, 0.76	0.12, 0.38	0.83, 0.44
			DISK	0.06, 0.32	0.18, 0.29	**0.32, 0.50**	0.12, 0.37	0.61, **0.28**

Table 3. Evaluation on 7Scenes dataset [50] for the compared methods. We show results for homogeneous and heterogeneous algorithms, grouping them depending on the algorithm used for mapping. Best results in bold, N/A refers to unsuccessful localization.

	Method	Algorithm		Median Pose Error (cm, °)						
		Map	Query	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
Homog.	Direct	SIFT	SIFT	2.59, 0.88	**2.16**, 0.90	1.11, 0.86	4.00, 1.14	5.74, 1.62	4.69, 1.55	5.61, 2.14
	CD^{emb} [13]	SIFT	DoG:BRIEF	2.63, 0.90	**2.16, 0.89**	1.13, 0.89	4.10, 1.18	5.91, 1.66	4.72, 1.54	8.61, 2.14
	Direct	SuperPoint	SuperPoint	**2.43, 0.85**	2.26, **0.89**	**1.05, 0.75**	**3.36, 1.00**	**5.43, 1.36**	**4.60, 1.48**	**5.09, 1.44**
Heterogeneous	CD^{emb} [13]	SIFT	SuperPoint	3.64, 1.08	3.05, 1.13	2.36, 1.61	5.52, 1.58	7.69, 2.08	5.98, 1.75	23.98, 6.52
			OrFAST:BRIEF	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	CF^{emb}	SIFT	SuperPoint	3.24, 0.99	2.91, 1.09	2.31, 1.55	5.41, 1.53	7.59, 2.22	5.55, 1.68	20.62, 5.39
			ORB	**2.74, 0.94**	**2.55, 1.00**	**1.70, 1.18**	**4.68, 1.34**	**6.84, 1.93**	**5.20, 1.63**	**14.63, 3.92**
			DISK	3.69, 1.26	4.00, 1.52	5.64, 3.29	8.71, 2.45	9.38, 2.56	7.04, 2.07	48.19, 47.19
	CD^{emb} [13]	Superpoint	SIFT	3.50, 1.15	3.63, 1.35	3.21, 2.02	6.17, 1.68	8.19, 2.16	6.49, 1.87	N/A
			OrFAST:BRIEF	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	CF^{emb}	Superpoint	SIFT	3.24, 1.12	3.21, 1.21	3.14, 2.05	6.12, 1.73	8.13, 2.03	6.14, 1.80	34.59, 20.98
			ORB	**2.93, 0.99**	**3.01, 1.15**	**1.88, 1.27**	**4.68, 1.32**	7.15, **1.94**	**5.52, 1.65**	**12.26, 3.22**
			DISK	2.97, 1.05	3.29, 1.23	1.99, 1.35	5.10, 1.42	**7.10**, 1.98	5.85, 1.76	16.27, 4.41

Table 4. Evaluation on four Cambridge Landmarks [21] scenes of the Collaborative Localization and Mapping scenario. We use a map created by merging two different independent submaps created with SIFT and SuperPoint, and then queried by all features.

Method	Query	Median error (m, °)			
		KingsCollege	OldHospital	StMarysChurch	GreatCourt
CD^{emb} [13]	SIFT	0.33, 0.42	0.30, 0.56	0.14, 0.41	0.34, 0.18
	SuperPoint	0.21, 0.31	0.38, 0.64	0.11, 0.31	0.85, 0.45
	DoG:BRIEF	0.60, 0.78	0.39, 0.59	0.17, 0.54	0.41, 0.20
	OrFAST:BRIEF	N/A	N/A	N/A	N/A
CF^{emb}	SIFT	0.18, 0.27	0.22, 0.40	0.10, 0.32	0.30, 0.15
	SuperPoint	0.18, 0.27	0.22, 0.40	0.10, 0.32	0.30, 0.15
	ORB	0.19, 0.28	0.32, 0.46	0.11, 0.37	0.59, 0.28
	DISK	0.18, 0.27	0.37, 0.58	0.14, 0.41	0.42, 0.47

downstream applications. Despite promising, the matching results of our proposal are still far from the performance of the homogeneous case, with a high number of outliers, a major concern to be addressed by future work. We envision different ways to improve the matching performance, such as an additional translation module using the match information to correct the location of the heterogeneous keypoints and ensure that both features represent the same physical point (with a subsequent positive impact on the downstream tasks); or to make use of global context in the features (e.g., encoding geometric information for enhanced projections [60] as in [9]) or in the matching process (e.g., using end-to-end matchers [24, 41]). We discussed several computational limitations of our work that require future optimization to improve performance or, eventually, a whole redesign. For example, the many-to-many assignation problem or the scalability concern in order to enable simultaneous learning of a real common embedding space for multiple features. Furthermore, we believe that this paper opens up a wide set of paths for research in Visual Localization Interoperability, for example, to study approaches merging and optimizing 3D maps created from heterogeneous features or to adapt current collaborative SLAM pipelines to allow Cross-Feature.

7. Acknowledgements

We would like to thank Marcos Gómez Bracamonte for helping with the integration of GoMatch in HLoc.

References

[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *CVPR*, pages 5297–5307, 2016. 2, 6

[2] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *BMVC*, page 3, 2016. 4

[3] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, pages 5173–5182, 2017. 5

[4] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008. 2

[5] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *ECCV*, pages 778–792, 2010. 2

[6] Song Cao and Noah Snavely. Minimal scene descriptions from structure from motion models. In *CVPR*, pages 461–468, 2014. 2

[7] Tim Caselitz, Bastian Steder, Michael Ruhnke, and Wolfram Burgard. Monocular camera localization in 3d lidar maps. In *IROS*, pages 1926–1931. IEEE, 2016. 2

[8] Daniele Cattaneo, Matteo Vaghi, Augusto Luis Ballardini, Simone Fontana, Domenico G Sorrenti, and Wolfram Burgard. Cmrnet: Camera to lidar-map registration. In *2019 IEEE intelligent transportation systems conference (ITSC)*, pages 1283–1289. IEEE, 2019. 2

[9] Paula Carbó Cubero, Alberto Jaenal Gálvez, André Mateus, José Araújo, and Patric Jensfelt. Matcha: Cross-algorithm matching with feature augmentation. *arXiv preprint arXiv:2506.22336*, 2025. 3, 8

[10] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPRW*, pages 224–236, 2018. 1, 2, 4

[11] Hao Dong, Xieyuanli Chen, Mihai Dusmanu, Viktor Larsson, Marc Pollefeys, and Cyrill Stachniss. Learning-based dimensionality reduction for computing compact and effective local feature descriptors. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6189–6195. IEEE, 2023. 2

[12] 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. 5

[13] Mihai Dusmanu, Ondrej Miksik, Johannes L Schönberger, and Marc Pollefeys. Cross-descriptor visual localization and mapping. In *ICCV*, pages 6058–6067, 2021. 1, 2, 3, 4, 5, 6, 7, 8

[14] Johan Edstedt, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Dedode: Detect, don't describe—describe, don't detect for local feature matching. In *3DV*, pages 148–157, 2024. 2

[15] Johan Edstedt, André Mateus, and Alberto Jaenal. Colabsfm: Collaborative structure-from-motion by point cloud registration. In *CVPR*, pages 6573–6583, 2025. 1

[16] Mohammed E Fathy, Quoc-Huy Tran, M Zeeshan Zia, Paul Vernaza, and Manmohan Chandraker. Hierarchical metric learning and matching for 2d and 3d geometric correspondences. In *Proceedings of the european conference on computer vision (ECCV)*, pages 803–819, 2018. 2

[17] Alejandro Fontan, Javier Civera, and Michael Milford. Anyfeature-vslam: Automating the usage of any chosen feature into visual slam. *Proceedings of the Robotics: Science and Systems, Delft, The Netherlands*, pages 15–19, 2024. 2

[18] Kun He, Yan Lu, and Stan Sclaroff. Local descriptors optimized for average precision. In *CVPR*, pages 596–605, 2018. 4

[19] Jie Hu, Rongrong Ji, Hong Liu, Shengchuan Zhang, Cheng Deng, and Qi Tian. Towards visual feature translation. In *CVPR*, pages 3004–3013, 2019. 2

[20] Neena Kamath. Announcing azure spatial anchors for collaborative, cross-platform mixed reality apps, 2019. 2

[21] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, pages 2938–2946, 2015. 2, 6, 7, 8

[22] Zakaria Laskar, Iaroslav Melekhov, Assia Benbihi, Shuzhe Wang, and Juho Kannala. Differentiable product quantization for memory efficient camera relocalization. In *ECCV*, pages 470–489. Springer, 2024. 2

[23] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *ICCV*, pages 2548–2555. Ieee, 2011. 2

[24] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. Lightglue: Local feature matching at light speed. In *ICCV*, pages 17581–17592, 2023. 8

[25] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004. 1, 2, 4

[26] Simon Lynen, Bernhard Zeisl, Dror Aiger, Michael Bosse, Joel Hesch, Marc Pollefeys, Roland Siegwart, and Torsten Sattler. Large-scale, real-time visual–inertial localization revisited. *The International Journal of Robotics Research*, 39 (9):1061–1084, 2020. 2

[27] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE TPAMI*, 27(10):1615–1630, 2005. 5

[28] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. *NeurIPS*, 30, 2017. 2

[29] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017. 1, 2, 4

[30] Son Tung Nguyen, Alejandro Fontan, Michael Milford, and Tobias Fischer. Fuseloc: Fusing global and local descriptors to disambiguate 2d-3d matching in visual localization. *arXiv preprint arXiv:2408.12037*, 2024. 2

[31] Shuxue Peng, Zihang He, Haotian Zhang, Ran Yan, Chuting Wang, Qingtian Zhu, and Xiao Liu. Megloc: A robust and accurate visual localization pipeline. *arXiv preprint arXiv:2111.13063*, 2021. 2

[32] Francesco Pittaluga, Sanjeev J Koppal, Sing Bing Kang, and Sudipta N Sinha. Revealing scenes by inverting structure from motion reconstructions. In *CVPR*, pages 145–154, 2019. 2, 3, 6

[33] Filip Radenovic, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *CVPR*, pages 5706–5715, 2018. 4

[34] Tilman Reinhardt. Using global localization to improve navigation, 2019. 2

[35] Jerome Revaud, Jon Almazán, Rafael S Rezende, and Cesar Roberto de Souza. Learning with average precision: Training image retrieval with a listwise loss. In *ICCV*, pages 5107–5116, 2019. 2

[36] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2d2: Reliable and repeatable detector and descriptor. *NeurIPS*, 32, 2019. 2, 4

[37] Fernando Reyes-Aviles, Philipp Fleck, Dieter Schmalstieg, and Clemens Arth. Bag of world anchors for instant large-scale localization. *IEEE Transactions on Visualization and Computer Graphics*, 2023. 2

[38] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *WACV*, pages 3674–3683, 2020. 4

[39] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, pages 2564–2571, 2011. 2, 4

[40] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, pages 12716–12725, 2019. 2, 6

[41] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 6, 8

[42] 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 *CVPR*, pages 3247–3257, 2021. 6

[43] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *ICCV*, pages 667–674. IEEE, 2011. 2

[44] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Improving image-based localization by active correspondence search. In *ECCV*, pages 752–765. Springer, 2012. 2

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

[46] Torsten Sattler, Michal Havlena, Filip Radenovic, Konrad Schindler, and Marc Pollefeys. Hyperpoints and fine vocabularies for large-scale location recognition. In *CVPR*, pages 2102–2110, 2015. 2

[47] Torsten Sattler, Akihiko Torii, Josef Sivic, Marc Pollefeys, Hajime Taira, Masatoshi Okutomi, and Tomas Pajdla. Are large-scale 3d models really necessary for accurate visual localization? In *CVPR*, pages 1637–1646, 2017. 2

[48] 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 *CVPR*, pages 8601–8610, 2018. 2, 4, 6

[49] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 1, 2

[50] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *CVPR*, pages 2930–2937, 2013. 2, 6, 8

[51] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[52] Tory Smith. Introducing lightship vps, 2022. 2

[53] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. Inloc: Indoor visual localization with dense matching and view synthesis. In *CVPR*, pages 7199–7209, 2018. 2

[54] Yurun Tian, Bin Fan, and Fuchao Wu. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *CVPR*, pages 661–669, 2017. 2, 4

[55] Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. Sosnet: Second order similarity regularization for local descriptor learning. In *CVPR*, pages 11016–11025, 2019. 2, 4

[56] 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 TPAMI*, 44(4):2074–2088, 2020. 2

[57] Akihiko Torii, Relja Arandjelovic, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *CVPR*, pages 1808–1817, 2015. 2, 6

[58] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. *NeurIPS*, 33, 2020. 2, 4

[59] Shuzhe Wang, Juho Kannala, and Daniel Barath. Dgc-gnn: Descriptor-free geometric-color graph neural network for 2d-3d matching. In *CVPR*, 2024. 2, 3

[60] Xinjiang Wang, Zeyu Liu, Yu Hu, Wei Xi, Wenxian Yu, and Danping Zou. Featurebooster: Boosting feature descriptors with a lightweight neural network. In *CVPR*, pages 7630–7639, 2023. 8

[61] Changchang Wu. Towards linear-time incremental structure from motion. In *3DV*, pages 127–134, 2013. 6, 7

[62] Qunjie Zhou, Sérgio Agostinho, Aljoša Ošep, and Laura Leal-Taixé. Is geometry enough for matching in visual localization? In *ECCV*, pages 407–425, 2022. 2, 3, 6