Cross-Descriptor Visual Localization and Mapping

Mihai Dusmanu 1  Ondrej Miksik 2  Johannes L. Schönberger 2  Marc Pollefeys 1, 2
1 Department of Computer Science, ETH Zürich 2 Microsoft MR & AI Lab, Zürich

Abstract

Visual localization and mapping is the key technology underlying the majority of mixed reality and robotics systems. Most state-of-the-art approaches rely on local features to establish correspondences between images. In this paper, we present three novel scenarios for localization and mapping which require the continuous update of feature representations and the ability to match across different feature types. While localization and mapping is a fundamental computer vision problem, the traditional setup supposes the same local features are used throughout the evolution of a map. Thus, whenever the underlying features are changed, the whole process is repeated from scratch. However, this is typically impossible in practice, because raw images are often not stored and re-building the maps could lead to loss of the attached digital content. To overcome the limitations of current approaches, we present the first principled solution to cross-descriptor localization and mapping. Our data-driven approach is agnostic to the feature descriptor type, has low computational requirements, and scales linearly with the number of description algorithms. Extensive experiments demonstrate the effectiveness of our approach on state-of-the-art benchmarks for a variety of handcrafted and learned features.

1. Introduction

Mixed reality and robotics blend the physical and digital worlds to unlock the next evolution in human, machine, and environment interaction. This promise largely relies on the technical capability to build and localize against maps of the environment. For example, collaborative experiences require co-localization of multiple devices into the same coordinate space. Similarly, re-localization against existing maps enables persistence and retrieval of digital content attached to the real world. Most systems build upon localization and mapping pipelines predominantly based on vision sensors using local features for image matching.

Traditionally, these systems run in real-time on the device, but the industry has been increasingly moving localization and mapping capabilities to the cloud (e.g., Facebook LiveMaps [2], Google VPS [42], Magic Leap’s MagicVerse [3], or Microsoft Azure Spatial Anchors [23]). This is for a variety of reasons, such as reducing on-board compute or enabling collaborative experiences and crowdsourced mapping. In most settings, images are typically not shared across devices or uploaded to the cloud primarily due to privacy reasons [25], but also to reduce bandwidth and storage requirements. Instead, image features are computed locally on the device and only the extracted keypoints and descriptors are shared. In other words, it is impossible to re-extract features whenever a different representation is required, as the images are no longer available.

There are two fundamental limitations of existing systems. First, they cannot adopt new feature algorithms because incompatibilities of feature representations make re-localization against maps built using older features impossible. In a world of continuous progress on local features in the research community, this severely limits progress on localization and mapping. One might argue for sim-
ply re-building the maps from scratch whenever a significantly improved feature algorithm is available. However, content attached to the old maps would be lost and mapping is an expensive process, where it could take weeks or even months until the whole area is re-visited. Second, co-localization and collaborative mapping scenarios with devices using different features is impossible. The situation is further complicated by the fact that many devices implement specific algorithms in hardware for efficiency reasons, making a client-side upgrade of algorithms impossible. This also means that many of the existing commercial solutions might be significantly behind the state-of-the-art in the research community, because they cannot easily upgrade their algorithms and representations.

In this paper, we first define three novel scenarios addressing the challenges in a world of changing local feature representations (c.f. Figures 1 and 2):

- **Continuous deployment** of feature representations without requiring an explicit re-mapping phase.
- **Cross-device localization** when the localization and mapping devices use different features.
- **Collaborative mapping** with multiple heterogeneous devices and features.

Note that these scenarios are completely different from existing industrial or academic setups, where generally a fixed local feature extraction algorithm is used for co-localization and throughout the evolution of a map. Section 3 introduces the scenarios in more detail.

As a first step towards enabling these scenarios, we focus on local feature descriptors and propose what to our knowledge is the first principled and scalable approach to the underlying challenges. Our learned approach translates descriptors from one representation to another without any assumptions about the structure of the feature vectors and enables matching of features with incompatible dimensionality as well as distance metrics. For instance, we can match 512 dimensional binary BRIEF [10] against 128 dimensional floating point SIFT [28] or even deep learning models such as HardNet [33] or SOSNet [51] and vice versa. Our method has linear scalability in the number of algorithms and is specifically designed to have a small computational footprint to enable deployment on low-compute devices. The training data is generated automatically by computing different descriptors from the same image patches.

We evaluate our approach on relevant geometric tasks in the context of the newly proposed scenarios. We first consider localization (pose estimation) on the Aachen Day & Night benchmark [46]. Next, we assess the performance for 3D mapping from crowd-sourced images using the benchmark of Schönberger et al. [48]. In the supplementary material, we also show results on the HPatches descriptor benchmark [7], the Image Matching Workshop challenge [22], and the InLoc Indoor Visual Localization dataset [50]. Our experiments demonstrate the effectiveness and high practical relevance of our method for real-world localization and mapping systems.

To summarize our contributions, we i) introduce three novel scenarios to localization and mapping in a world of changing feature representations, ii) propose the first principled and scalable approach tackling these newly introduced scenarios, and iii) demonstrate the effectiveness of our method on challenging real-world datasets.

## 2. Related work

### Local image descriptors

Local descriptors are typically extracted from normalized images patches defined by local feature frames [28, 31, 44]. The most notable hand-crafted descriptors include binary BRIEF [10], gradient-based SIFT [28], and their variants [44, 32, 52, 6]. Recently, the community has moved to data-driven models leveraging large-scale datasets, various triplet losses, and hard-negative mining techniques [8, 33, 29, 51, 16]. A related line of research aims to reformulate the entire pipeline in an end-to-end trainable fashion [53, 39]. Finally, the describe-then-detect approach extracts local features from dense feature maps [38, 15, 43, 30]. While it is possible to handcraft relations between some of these descriptors (e.g. SIFT [28] ↔ Hue-SIFT [52]), they are typically not designed for inter-compatibility. In contrast, our approach aims to enable matching of local descriptors with incompatible feature representations, dimensionalities or metrics.

### Visual localization and mapping

Given the efficiency of local features and the well-established theory of (sparse) multi-view geometry, a majority of large-scale mapping [4, 19] and localization [27, 45] is based on local image features. Furthermore, there is a growing number of open-source tools for Simultaneous Localization and Mapping (e.g., ORB-SLAM [36, 37]), visual odometry (e.g., LIBVISO [26, 18]) or Structure-from-Motion (e.g., COLMAP [47], OpenMVG [35], Thea [49]) rendering this research domain more accessible. Recently, these pipelines were adopted for evaluation of visual localization and mapping – Schönberger et al. [48] and Jin et al. [22] considered mapping and localization performance using phototourism collections, while Sattler et al. [46] introduced several datasets for localization under challenging scenarios (e.g., seasonal / temporal changes). Also related to our work, the problem of collaborative mapping between multiple agents (e.g., robots, drones) has been thoroughly studied [17, 11, 34] in the field of robotics. However, all these systems and associated benchmarks use the same local image features throughout the whole process and are thus unable to tackle any of the scenarios discussed in Section 3.

### Domain translation

Several local feature inversion techniques [14, 40] reconstructing the original image from its
features were proposed. One way to enable descriptor translation would be to reconstruct the image and then follow the traditional pipeline to extract a different descriptor. In our early experiments with feature inversion networks, we noticed significant limitations in generalizing to new scenes and poor quality of recovered low-level gradient information. As such, it was not possible to successfully match images using this feature inversion based approach.

Image-to-image translation [54, 21, 55] attempts to convert one representation of a scene to another (e.g., RGB image to semantic labels). One could imagine adapting these models to translate between different local feature representations. However, similarly to feature inversion techniques, these approaches typically have very large computational and memory footprint, as they employ deep convolutional neural networks to produce full resolution images. In contrast, we propose to use shallow multi-layer perceptrons, taking a single descriptor as input and predicting its translation – a solution more suitable for low-compute devices.

Finally, research on domain adaptation (c.f., surveys of Csurka [12] and Zhuang et al. [56]) tackles the problem of adapting algorithms learned on a source data distribution to a dataset with a related but different distribution. In the case of deep learning, this is particularly interesting when the source domain has enough annotations to allow training while the target domain has little to no annotations available. In our case, instead of adapting to a different input data distribution, we try to adapt the output distributions to make them compatible for matching.

3. Scenarios

As our first contribution, we identify, introduce, and formalize three novel scenarios for localization and mapping. In all scenarios, we assume there are no images stored in the resulting maps. Therefore, it is impossible to re-extract or replace the underlying feature representation.

3.1. Continuous deployment

Given a sparse 3D map, associated local feature descriptors and no access to the original image data, the goal is to develop a mechanism enabling the continuous deployment of new feature representations (Figure 2 (a)) without the need for re-mapping. In other words, we aim to translate the features of the map from one representation to another. For instance, it should be possible to switch from handcrafted to learned descriptors (e.g., SIFT [28] → SOSNet [51]), change their dimensionality, or update the model weights (e.g., different training data, loss, or architecture).

Matching the new feature representation against a database of translated descriptors might lead to worse performance when compared to matching using a single feature representation. This is, however, only a transient issue and not of serious concern in practice, since the translated descriptors can be used in a bootstrapping fashion – as users re-visit the environment, the translated descriptors are gradually replaced by the newly extracted ones. Thus, the map will eventually contain the latest descriptors to close the performance gap. The key points are that i) the devices run only a single algorithm (due to real-time constraints) or cannot be easily updated (due to hardware implementation), ii) the scenario avoids an explicit re-mapping phase, which is generally very expensive and time-consuming.

3.2. Cross-device localization

The previous scenario assumes the same feature extraction algorithm is running on all client devices. However, this is generally not the case, as legacy devices often cannot be updated due to hardware limitations. In this case, a mechanism for backwards-compatibility is required to match across different device versions (see Figure 2 (b)). Similarly, specialized devices (e.g., headsets, autonomous cars, mapping platforms) could take advantage of additional on-board compute for a better localization perfor-
mance compared to devices such as mobile phones or light-weight drones. It would be beneficial to co-localize these devices inside the same maps to avoid a fragmented scene representation. The same functionality could be extended to enable localization between devices of different vendors, but, in this situation, new challenges arise due to differences in previous steps of the pipeline – notably feature detection, as discussed in detail in Section 6.

### 3.3. Collaborative mapping

Finally, we consider multiple heterogeneous devices collaboratively mapping the same environment. For the aforementioned reasons, it is fairly unlikely for all devices to use the same features as in a standard academic setup, even when produced by a single vendor. Consequently, we end up with each device providing different types of features from which we need to build a single coherent map. Therefore, we need a mechanism for translating all descriptors into a common representation that can be used for establishing correspondences (see Figure 1).

### 4. Method

In this section, we present the first principled solution to cross-descriptor localization and mapping. We start by formalizing the descriptor translation problem. We then propose to use a separate multi-layer perceptron (MLP) for each pair of description algorithms, trained using a translation loss (c.f. Figure 3 (a)). Finally, we extend this formulation by using an auto-encoder inspired model that embeds all descriptors from different algorithms into a joint embedding space (c.f. Figure 3 (b)). To make sure that the common embedding is suitable for establishing correspondences between different algorithms, we leverage additional supervision through a matching loss.

#### 4.1. Descriptor translation

A feature description algorithm $A$ defines a handcrafted or learned function mapping images to vectors as $A: \mathcal{I}_{h \times w} \rightarrow \mathbb{R}^n$, where $\mathcal{I}_{h \times w}$ is the set of images of size $h \times w$ and $n$ is the embedding dimension. In the case of local image descriptors, the domain is the set of patches $\mathcal{I}$ normalized based on the estimated feature geometry (position, scale, orientation, affine shape).

Let $A_1: \mathcal{I} \rightarrow \mathbb{R}^{n_1}$ and $A_2: \mathcal{I} \rightarrow \mathbb{R}^{n_2}$ be two description algorithms. A translation function $t_{1 \rightarrow 2}: \mathbb{R}^{n_1} \rightarrow \mathbb{R}^{n_2}$ maps a feature vector of $A_1$ onto the manifold of $A_2$, satisfying $t_{1 \rightarrow 2}(A_1(p)) = A_2(p)$ for all patches $p \in \mathcal{I}$. Our experiments (Section 5) empirically demonstrate that using MLPs to approximate the translation functions $t_{1 \rightarrow 2}$ is highly effective in enabling matching between different descriptors which, in turn, facilitates the scenarios introduced in Section 3. Note that we do not propose a matching algorithm but instead directly map descriptors from one space to another. Our approach thus relies on traditional descriptor comparison strategies, lending itself to efficient matching implementations (e.g., approximate nearest neighbor). Apart from the corresponding descriptors being extracted from the same patches, we make no additional assumptions regarding the descriptor dimension or the underlying algorithm (handcrafted or learned, binary or floating point).

In the following, we consider a set of description algorithms $A = \{A_1, A_2, \ldots\}$. Our training batches consist of normalized images patches $\mathcal{P} = \{p_1, p_2, \ldots\}$. Each description algorithm can be applied independently on a patch $p \in \mathcal{P}$ to obtain the associated descriptor, i.e., $A_i(p) = a_i$.

#### 4.2. Pair network

Given a pair of different description algorithms $(A_i, A_j)$, one could approximate the translation function $t_{i \rightarrow j}$ by a multi-layer perceptron $N_{i \rightarrow j}$ (c.f. Figure 3 (a)). This model translates features extracted using algorithm $A_i$ into the feature representation of algorithm $A_j$. To this end, we define the translation loss as the $\ell_2$ prediction error:

$$L_{i \rightarrow j}^T = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \|N_{i \rightarrow j}(A_i(p)) - A_j(p)\|_2. \quad (1)$$

If the target algorithm $A_j$ produces binary descriptors, the regression loss is replaced by a classification loss:

$$L_{i \rightarrow j}^T = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \text{BCE}(N_{i \rightarrow j}(A_i(p)), A_j(p)) \quad (2)$$

where BCE is the binary cross entropy loss function defined as $\text{BCE}(x, y) = -y \log(x) - (1 - y) \log(1 - x)$.

To compute the distance $d$ between two descriptors $a_i, a_j$ extracted using algorithms $A_i, A_j$, respectively, one can translate from $A_i$ to $A_j$ by using the associated MLP ($d = \|N_{i \rightarrow j}(a_i) - a_j\|$) or conversely ($d = \|N_{j \rightarrow i}(a_j) - a_i\|$). In most practical applications, the direction of translation is known. In the case of continuous software updates, the outdated map descriptors are “migrated” to the new ones, while for cross-device localization the query descriptors are always translated to map descriptors. However, it is not obvious how to use this approach in the collaborative mapping scenario as there are $O(|A|^2)$ possible translation directions. One could use a “progressive” strategy that translates the weaker to the better descriptor for each image pair, but deciding which descriptor is more suitable is not straightforward as it generally depends on the specific scene.  

#### 4.3. Encoder-decoder network

To address this issue, we turn our attention towards auto-encoders. Each description algorithm $A_i$ now has an associated encoder $E_i$ and decoder $D_i$ that ideally satisfy

---

1Please refer to the supplementary material for a scene-by-scene breakdown on the HPatches Sequences dataset [7] that supports this argument.
We propose an extension to the identity loss traditionally used for training auto-encoders [9], such that the codes (i.e., the output of encoders / the input of decoders) live in a joint space. In green, we highlight the networks that need to be used in order to translate from SIFT to HardNet.

Translation loss. As above, we consider the translation loss $L_{i\rightarrow j}^{T}$ for each algorithm pair $A_i, A_j \in \mathcal{A}^2$. Note that, for $i = j$, this loss is equivalent to the traditional auto-encoder loss [9]. The overall translation loss is defined as the average over all algorithm pairs as:

$$L^{T} = \frac{1}{|\mathcal{A}|^2} \sum_{A_i, A_j \in \mathcal{A}^2} L_{i\rightarrow j}^{T}.$$  

Matching loss. We use a triplet margin loss for each description algorithm pair $A_i, A_j \in \mathcal{A}^2$ to allow matching in the joint embedding space:

$$L_{i\rightarrow j}^{M} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \max \left( m + \text{pos}(p) - \text{neg}(p), 0 \right),$$

where pos($p$) and neg($p$) are the distances to the positive and negative samples, respectively. Given two corresponding descriptors $A_i(p) = a_i, A_j(p) = a_j$, the positive sample for the embedding of $a_i$ is simply the embedding of $a_j$:

$$\text{pos}(p) = \| E_i(A_i(p)) - E_j(A_j(p)) \|. \quad (5)$$

Following HardNet [33], we use in-batch hardest negative mining by selecting the closest non-matching embedding of a descriptor extracted using algorithm $A_j$ as negative sample for the embedding of the current descriptor $a_i$:

$$\text{neg}(p) = \arg \min_{p' \in \mathcal{P}, p' \neq p} \| E_i(A_i(p)) - E_j(A_j(p')) \|. \quad (6)$$

The overall matching loss is then defined as the sum over all description algorithm pairs as:

$$L^{M} = \frac{1}{|\mathcal{A}|^2} \sum_{A_i, A_j \in \mathcal{A}^2} L_{i\rightarrow j}^{M}. \quad (7)$$

Final loss. The final loss is a weighted sum of the translation and matching losses as: $\mathcal{L} = \mathcal{L}^{T} + \alpha \mathcal{L}^{M}$.

4.4. Implementation details

Training dataset. Our training dataset contains 7.4 million patches from 3190 random internet images of the Oxford-Paris revisited retrieval dataset distractors [41]. The normalized image patches are extracted around Difference-of-Gaussians (DoG) detections according to the estimated scale and orientation. Training descriptors are computed by applying all description algorithms on each patch to obtain a set of corresponding descriptors. For DoG keypoint extraction we use COLMAP [47] with default parameters.

Training methodology. We train all networks for 5 epochs using Adam [24] as optimizer with a learning rate of $10^{-3}$ and a batch size of 1024. Similar to Mishchuk et al. [33], we use a margin $m = 1$. According to validation results, we fix
the weighting of the matching loss $\alpha = 0.1$. The encoders and decoders are MLPs with 2 hidden layers of sizes 1024 and 256 for hand-crafted and learned features, respectively. We use ReLU followed by batch normalization [20] after each linear layer apart from the last. For consistency with state-of-the-art floating point local descriptors [28, 33, 51], the joint embedding is 128-dimensional and $\ell_2$ normalized. For binary descriptors [10], we use sigmoid activation after the last linear layer and, at test time, we use a threshold of 0.5 for binarization to allow efficient matching using bitwise operations. Similarly, if needed [28, 33, 51], the output of the network is $\ell_2$ normalized. Please refer to the supplementary material for additional details.

5. Experimental evaluation

We evaluate our approach on the tasks of visual localization and mapping in the context of the scenarios introduced in Section 3. Please refer to the supplementary material for additional results on the Hpatches descriptor benchmark [7], the Image Matching Workshop challenge [22], and the InLoc Indoor Visual Localization dataset [50]. We consider four well-established widely used descriptors – two handcrafted descriptors, with BRIEF [10] as a binary and SIFT [28] as a gradient-based approach as well as two state-of-the-art learned descriptors, HardNet [33] and SOSNet [51]. In our initial experiments, we found the pair network to perform on par with the encoder-decoder approach in terms of accuracy and runtime. However, the encoder-decoder architecture provides us with a joint embedding and covers all three discussed scenarios. Thus, all results presented in this section were obtained using the encoder-decoder approach. Please refer to the supplementary material for a comparison of the two approaches.

5.1. Visual localization

We consider the task of visual localization against pre-built maps on the challenging Aachen Day & Night localization benchmark [46]. We start by matching each database image with its 20 nearest spatial neighbors. COLMAP [47] is used for triangulation using the provided camera poses and intrinsics. Next, each query image is matched against its top 50 retrieval results according to NetVLAD [5]. Finally, we use COLMAP’s image registrator for localization with known intrinsics. According to standard procedure, the poses are submitted to the evaluation system [1] and we report the percentage of localized images at different real-world thresholds in Table 1.

Localization under continuous deployment. To evaluate this scenario, we build the reference map from a single description algorithm (e.g., SIFT). Then, all the feature descriptors of the map are translated to a target feature representation (e.g., SIFT $\rightarrow$ HardNet). For the queries, we use descriptors extracted directly by the target feature description algorithm (e.g., HardNet) and match them against the translated features. The resulting matches are used for localization. Our approach not only enables updating descriptors without re-mapping the environment, but in many cases also unexpectedly improves the re-localization performance.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Database descriptor</th>
<th>Query descriptor</th>
<th>Day (824 images)</th>
<th>Night (98 images)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.25m, 2°</td>
<td>0.5m, 5°</td>
</tr>
<tr>
<td>Standard</td>
<td>BRIEF</td>
<td>BRIEF</td>
<td>76.1</td>
<td>81.4</td>
</tr>
<tr>
<td></td>
<td>SIFT</td>
<td>SIFT</td>
<td>82.5</td>
<td>88.7</td>
</tr>
<tr>
<td></td>
<td>HardNet</td>
<td>HardNet</td>
<td>86.2</td>
<td>92.2</td>
</tr>
<tr>
<td></td>
<td>SOSNet</td>
<td>SOSNet</td>
<td>86.4</td>
<td>92.7</td>
</tr>
<tr>
<td>BRIEF $\rightarrow$ SIFT</td>
<td>HardNet</td>
<td>74.9</td>
<td>80.5</td>
<td>31.6</td>
</tr>
<tr>
<td>BRIEF $\rightarrow$ HardNet</td>
<td>SOSNet</td>
<td>81.4</td>
<td>85.3</td>
<td>54.9</td>
</tr>
<tr>
<td>BRIEF $\rightarrow$ SOSNet</td>
<td>76.6</td>
<td>73.1</td>
<td>19.4</td>
<td>32.6</td>
</tr>
<tr>
<td>SIFT $\rightarrow$ HardNet</td>
<td>HardNet</td>
<td>83.4</td>
<td>90.9</td>
<td>52.2</td>
</tr>
<tr>
<td>SIFT $\rightarrow$ SOSNet</td>
<td>84.2</td>
<td>91.4</td>
<td>55.1</td>
<td>63.1</td>
</tr>
<tr>
<td>HardNet $\rightarrow$ SIFT</td>
<td>HardNet</td>
<td>70.5</td>
<td>76.7</td>
<td>22.4</td>
</tr>
<tr>
<td>HardNet $\rightarrow$ SOSNet</td>
<td>68.8</td>
<td>74.8</td>
<td>18.4</td>
<td>46.9</td>
</tr>
<tr>
<td>SOSNet $\rightarrow$ SIFT</td>
<td>HardNet</td>
<td>81.7</td>
<td>87.5</td>
<td>42.9</td>
</tr>
<tr>
<td>SOSNet $\rightarrow$ SOSNet</td>
<td>85.9</td>
<td>92.4</td>
<td>63.3</td>
<td>2.0</td>
</tr>
<tr>
<td>BRIEF $\leftarrow$ SIFT</td>
<td>HardNet</td>
<td>75.6</td>
<td>80.8</td>
<td>28.6</td>
</tr>
<tr>
<td>BRIEF $\leftarrow$ SOSNet</td>
<td>68.8</td>
<td>74.0</td>
<td>20.4</td>
<td>12.3</td>
</tr>
<tr>
<td>SIFT $\leftarrow$ HardNet</td>
<td>HardNet</td>
<td>81.1</td>
<td>87.3</td>
<td>46.9</td>
</tr>
<tr>
<td>SIFT $\leftarrow$ SOSNet</td>
<td>80.1</td>
<td>87.3</td>
<td>42.9</td>
<td>91.1</td>
</tr>
<tr>
<td>HardNet $\leftarrow$ SIFT</td>
<td>HardNet</td>
<td>82.5</td>
<td>88.2</td>
<td>44.9</td>
</tr>
<tr>
<td>HardNet $\leftarrow$ SOSNet</td>
<td>84.0</td>
<td>91.1</td>
<td>51.0</td>
<td>14.3</td>
</tr>
<tr>
<td>SOSNet $\leftarrow$ SIFT</td>
<td>HardNet</td>
<td>85.3</td>
<td>91.9</td>
<td>66.3</td>
</tr>
<tr>
<td>SOSNet $\leftarrow$ SOSNet</td>
<td>84.8</td>
<td>90.9</td>
<td>57.1</td>
<td>8.2</td>
</tr>
</tbody>
</table>

Table 1: Localization under continuous deployment. A reference map is built using the database description algorithm. The descriptors of this map are translated to a target query descriptor. Cross-device localization. A reference map is built using the database description algorithm. The descriptors of query images are translated to be compatible with the map. Localization to collaborative maps. The database images are partitioned in 4 balanced subsets, one for each description algorithm. Both database and query descriptors are mapped to the common embedding space. The absolute performance difference (superscript) is color-coded according to the baseline used as reference.
performance. In particular, hand-crafted descriptors [10, 28] are significantly better after translating them to learned ones [33, 51]. For SIFT, the increase is of more than 2% and 5% in absolute performance at the largest threshold for day and night queries, respectively, and for BRIEF the difference is even more accentuated with 5% and 10%. While there is still a gap w.r.t. re-building the map from scratch using the learned descriptors, the performance can be further improved using a bootstrapping strategy (e.g. Section 3.1).

Cross-device localization. Inverse to the previous experiment, we now build maps using a description algorithm (e.g., SIFT) and instead translate the query descriptors (e.g., HardNet → SIFT). Similarly to the previous case, compared to standard localization, we notice increased performance when translating hand-crafted descriptors to learned ones. Maps built from state-of-the-art descriptors (HardNet, SOSNet) preserve their advantages no matter the description algorithm used for queries. Not surprisingly, translating other descriptors to BRIEF [10] yields worse performance due to the more limited discriminative power of binary descriptors. Overall, our approach enables localization with a good performance in this previously infeasible scenario.

Localization to collaborative maps. For the final scenario, we randomly split the database images of the map into balanced subsets (one for each description algorithm) and translate all descriptors to the joint embedding space (i.e., SIFT, HardNet, ... → embedding). Similarly, all query descriptors are translated to the joint embedding space for matching. Although the map is built from heterogeneous descriptors, the localization performance at day is comparable to the best results in the previous experiments. Moreover, note that the performance of HardNet and SOSNet in this scenario is just slightly below the current state-of-the-art patch-based descriptor [51]. In addition, BRIEF queries achieve a substantial improvement of more than 5% for both day and night folds at the coarsest threshold. This is a very encouraging result for cloud-based solutions, as we enable multiple devices implementing different algorithms to not only collaborate on mapping a location, but also remain competitive with the state-of-the-art in localization.

5.2. Collaborative mapping

Next, we consider the large-scale Structure-from-Motion benchmark of Schönberger et al. [48]. Similar to other datasets [46, 22], to evaluate the camera poses, we generate pseudo-ground-truth intrinsics and extrinsics via an initial SfM process. Only the images registered in this step are considered.\(^2\) For each method, we exhaustively match all remaining images using a mutual nearest-neighbors matcher with symmetric second nearest neighbor ratio test [28] (the threshold is set to 0.9 for all methods). Given

\(^2\)Please refer to the supplementary material for additional details.
6. Discussion

The main limitation comes from the same keypoint assumption, as our work only considers the compatibility of different description algorithms. While our assumption is likely applicable for devices produced by a single manufacturer that has control over the entire pipeline, different manufacturers likely use different keypoint detection algorithms. Possible directions for future research include studying cross-detector repeatability or improving feature inversion techniques for better generalization with different local feature detection pipelines. Despite our proposed solution being only a first step, it still has significant real-world applications: i) migration of existing maps from, e.g., handcrafted descriptors to learned ones and, similarly, continuous updates of learned descriptors in case of novel architectures / losses / datasets, ii) support of legacy devices inside maps built with newer descriptors, and iii) collaborative mapping between devices of the same manufacturer with different compute capabilities. Furthermore, while we have made a significant step towards enabling cross-detector localization and mapping, several challenges still remain. Information loss. Each description algorithm only encodes a subset of the information available in the input patches and these subsets might be distinct for different methods. This is further exacerbated in efficient real-world applications taking advantage of quantization and dimensionality reduction. One-to-many associations. A one-to-one mapping may not even exist as two given patches might lead to a similar descriptor for one algorithm but completely different feature vectors for another one. Similarly to other data-driven approaches, translation performance has to be verified experimentally as there are no formal guarantees. To address these limitations, future research could try to take advantage of local or global context to better disambiguate the visual information.

7. Conclusion

We have identified, introduced, and formalized three novel scenarios for localization and mapping in the presence of heterogeneous feature representations. Towards addressing the open challenges underlying these scenarios, we presented the first principled solution to cross-detector localization and mapping. We demonstrated the effectiveness of our approach on representative and well-established benchmarks. We believe our work will not only spark new research on the topic of translating local image features but will also make an immediate impact on commercial applications in the area of cloud-based localization and mapping.

Acknowledgments. This work was supported by the Microsoft MR & AI Lab Zürich PhD scholarship.
References


[18] Andreas Geiger, Julius Ziegler, and Christoph Stiller. StereoScan: Dense 3D Reconstruction in Real-time. In Intelligent Vehicles Symposium (IV), 2011. 2


[33] Anastasiya Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor’s mar-
gins: Local descriptor learning loss. In Advances in NeurIPS, 2017. 2, 5, 6, 7


[38] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-Scale Image Retrieval with Attentional Deep Local Features. In Proc. ICCV, 2017. 2


