Place Recognition in Semi-Dense Maps: Geometric and Learning-Based Approaches

Yawei Ye
yye@student.ethz.ch
Titus Cieslewski
titus@ifi.uzh.ch
Antonio Loquercio
loquercio@ifi.uzh.ch
Davide Scaramuzza
sdavide@ifi.uzh.ch

Robotics and Perception Group
University of Zurich
http://rpg.ifi.uzh.ch
Zurich, Switzerland

Abstract

For robotics and augmented reality systems operating in large and dynamic environments, place recognition and tracking using vision represent very challenging tasks. Additionally, when these systems need to reliably operate for very long time periods, such as months or years, further challenges are introduced by severe environmental changes, that can significantly alter the visual appearance of a scene. Thus, to unlock long term, large scale visual place recognition, it is necessary to develop new methodologies for improving localization under difficult conditions. As shown in previous work, gains in robustness can be achieved by exploiting the 3D structural information of a scene. The latter, extracted from image sequences, carries in fact more discriminative clues than individual images only. In this paper, we propose to represent a scene's structure with semi-dense point clouds, due to their highly informative power, and the simplicity of their generation through mature visual odometry and SLAM systems. Then we cast place recognition as an instance of pose retrieval and evaluate several techniques, including recent learning based approaches, to produce discriminative descriptors of semi-dense point clouds. Our proposed methodology, evaluated on the recently published and challenging Oxford Robotcar Dataset, shows to outperform image-based place recognition, with improvements up to 30% in precision across strong appearance changes. To the best of our knowledge, we are the first to propose place recognition in semi-dense maps.

1 Introduction

Visual place recognition has become a highly active area of research both in computer vision and robotics communities over past years and has many practical applications, such as autonomous navigation and augmented reality. A major challenge in visual place recognition is to achieve robustness to the appearance changes of scenes encountered in the real world due to lighting, weather and seasonal changes.

Visual place recognition is typically cast as an image retrieval problem, where the approach is to recognize places by matching individual images of these places. One of the

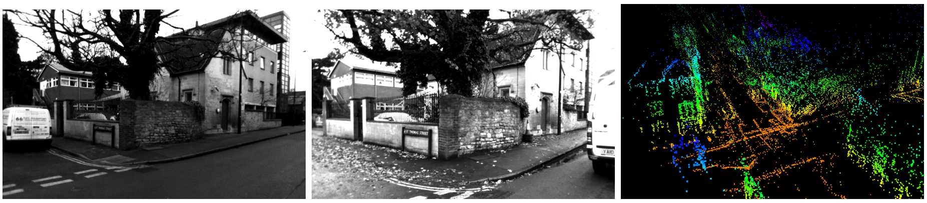


Figure 1: Images of the same intersection in the Oxford Robotcar Dataset [19] captured on February and October 2015, and a point cloud of the same place obtained from a sequence in May. The appearance of the scene strongly changes due to vegetation, lighting, camera exposure and traffic. The geometrical structure of the scene as captured by DSO however remains mostly consistent. Using the structure furthermore allows us to capture data on a larger scope than from just one image.

most widely used approaches for matching these images is to utilize sparse local features selected at salient regions of the image. Locally salient regions of the image are detected and descriptors are calculated that allow direct matching of these points. Classically, hand-crafted descriptors were used [1, 3, 15, 17, 27]. In contrast, recent methods allow learning of more discriminative descriptors in a data-driven manner [6, 16, 30].

For computational efficiency, a query image is not directly matched to all images in the database. Instead, images are typically reduced to compact representations [13, 22, 34] that are inherently less expensive to match and can further exploit efficient matching methods such as inverted indices or kd-trees. Moreover, image-based place recognition based on these techniques has been recently shown to be trainable in an end-to-end manner [2].

A drawback of the image matching approach is that it is inherently sensitive to changes in appearance that can be due to lighting, weather and seasonal changes. Exploiting the geometry of scenes can help to remedy this problem, but to date only very few image-based approaches for visual place recognition exist that do this [14, 23, 35, 36]. It is common practice to use geometrical verification [38] as a post-matching step, yet this can only be used to reject false positive matches, but not to obtain more true positives. An interesting alternative approach to image-to-image matching is to match 2D features directly to the points of a 3D point cloud [29], which results in a tighter integration of matching and geometric verification.

In a robotic context, regular cameras provide more information that can be used for place recognition than just individual images. It has been previously proposed to exploit information contained in the sequence in which images occur [18, 21] and in sparse point clouds extracted by feature-based structure from motion (SfM) algorithms [7]. We propose to go one step further and to exploit the information contained in semi-dense point clouds. In particular, we mean point clouds that are created by methods that track and triangulate all image points with a gradient, not just corners. Examples for such methods are LSD-SLAM [8], DSO [9] and more recently methods using event-based cameras [25, 26]. Based on examples like that in Fig. 1, our intuition is that the overall shape of these point clouds is more invariant to environmental changes than the appearance of the scenes or the sparse point clouds obtained from triangulating tracked features. To the best of our knowledge, we are the first to consider semi-dense point clouds for place recognition.

To allow for a similar efficiency as in image-based place recognition, we base our method on local descriptor matching. We evaluate the ability to discriminatively describe semi-dense point clouds of several methods from the literature. In particular, we also propose to use a

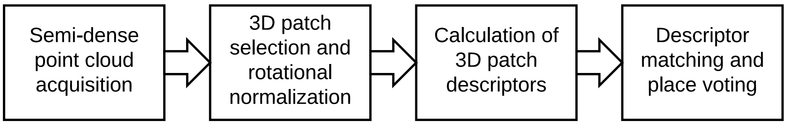

Semi-dense point cloud acquisition
3D patch selection and rotational normalization
Calculation of 3D patch descriptors
Descriptor matching and place voting

Figure 2: The used place recognition pipeline. From an initial semi-dense point cloud we extract and normalize local patches. The patches are described using either hand-crafted descriptors or a 3D convolutional neural network. Finally, the resulting descriptors are matched to descriptors in the database and the descriptor matches are aggregated to keyframe matches.

trainable neural network, which, in accordance with recent trends, outperforms methods using hand-crafted descriptors. The methods are trained and evaluated on the Oxford Robotcar Dataset [19], which contains several sequences collected by a car over a period of more than a year. As a consequence, the recorded sequences contain strong and realistic variations in lighting, weather and seasonal conditions, as well as occlusions by other traffic participants.

2 Related Work

2.1 Hand-crafted 3D Structural Descriptors

Many structural descriptors have been introduced for *dense* 3D point clouds. As an example, *Point Feature Histograms* (PFH) [28] and the *Sigature of Histograms of Orientations* (SHOT) [37] characterize the local geometry by computing surface normals within a neighbourhood and using a histogram to describe the extracted features. However, semi-dense point clouds do not have the required point density to estimate surface normals. Thus, we are restricted to using descriptors which do not rely on surface normals. From those, the neighbour-binary landmark density descriptor (NBLD) [7, 10] and the multiview 2D projeciton descriptor (M2DP) [11] have been shown to perform best in recent literature. We thus consider NBLD and M2DP in our evaluation and detail their working principles in Section 3.3.1.

2.2 Learned 3D Structural Descriptors

A recent development in the design of descriptors is to move away from hand-crafted features towards features learned from massive amounts of data, in particular using neural networks. Both *VoxNet* [20] and *3D ShapeNets* [39] introduced the use of deep learning to encode shapes of different objects. They focus on learning features from complete dense 3D point clouds at object level and are designed for object classification. Work that is more aimed at place recognition within point clouds was recently done by *3DMatch* [40], which learns a local structural descriptor using a Siamese network [5] for establishing correspondences between partial 3D RGB-D data. The use of Siamese networks for traning discriminative descriptors has recently proven to be successful also in the context of image-based place recognition [2, 16].

3 Place Recognition

An outline of the place recognition pipeline we adopt is provided in Fig. 2. In this section, we briefly describe the components of this pipeline.

3.1 Semi-dense point cloud acquisition

As mentioned in the introduction, the semi-dense point clouds that we use for place recognition can be acquired using any system that tracks and triangulates points with nonzero image gradient. Throughout our experiments, we use DSO [9] to acquire the semi-dense point clouds, since it proves to track the longest on the Oxford Robotcar Dataset.

3.2 3D patch selection and rotational normalization

A random five percent of the resulting points are selected as keypoints $\vec{p}$ and local cylindrical patches Γ are extracted around them. The patches Γ are centered around the keypoints and their size is chosen to be as small as possible while fitting the domain Ω of the used descriptor. Ω is defined as the subspace of $\mathbb{R}^3$ which contains all points contributing to the descriptor, and no points that do not contribute to it. Since we evaluate descriptors with different shapes of Ω, Γ varies depending on the descriptor used.

Since none of the evaluated descriptors are invariant to orientation, but the point clouds may originate from differently oriented frames of reference, we normalize the patches in orientation. We assume that the gravity vector is known and aligned with the z-axis, so that the patches only need to be normalized in yaw. The shape of the patches Γ is cylidrical to allow for a normalization that is independent of the initial orientation of the point cloud. We use the normalization proposed in [4]: first, the sample covariance C of the points within the patch is computed. Let $\vec{e}_0$ be the unitary eigenvector corresponding to the smallest eigenvalue λ_0 of C. Then, the projection of $\vec{e}_0$ onto the horizontal plane is chosen as the x-axis of the normalized patch. Since $\vec{e}_0$ is ambiguous ($-\vec{e}_0$ is also an eigenvector of λ_0), we choose the $\vec{e}_0$ which points towards the first observer of the keypoint $\vec{p}$.

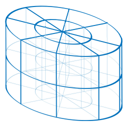
3.3 Calculation of 3D patch descriptors

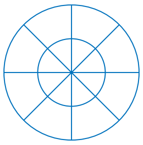
3.3.1 Hand-crafted Descriptors

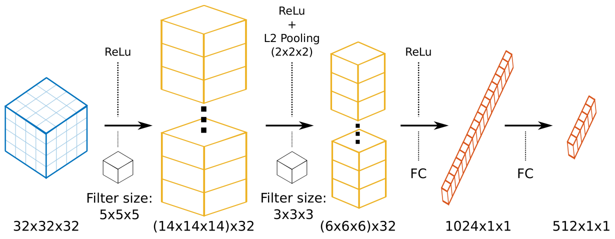
We evaluate two hand-crafted descriptors that were originally designed for dense and sparse point clouds: NBLD [7] and M2DP [11]. These are recently proposed hand-crafted descriptors that unlike many other structural descriptors do not rely on surface normals. They report superior performance to previous descriptors that also do not use surface normals.

The NBLD descriptor has a cylidrical domain Ω, with respectively n_a, n_r and n_z subdivisions in azimuthal, radial and z-direction, see Fig. 3(a). It considers the point density in each cell, that is, the point count divided by the cell volume. Inspired by the success of binary photometric descriptors [1, 15, 27], NBLD does not use absolute densities but rather comparsions between densities of neighboring cells. Its $3 \cdot n_r \cdot n_a \cdot n_z$ binary coefficients indicate for each cell how its point density compares to its neighbors in radial, azimuthal and z directions.

M2DP projects the points within its domain Ω into $n_E \cdot n_A$ 2D planes. We set Ω to be cylidrical with a radius r and to have an infininte extent in z. Each of the $n_E \cdot n_A$ planes crosses the keypoint $\vec{p}$ and is defined by its normal vector, which is expressed as $(1, \theta, \phi)^T$ in spherical coordinates. θ and ϕ respectively assume n_E and n_A linearly spaced values in $[0, \frac{\pi}{2}]$ (elevation angles) and $[-\frac{\pi}{2}, \frac{\pi}{2}]$ (azimuth angles). Into each plane, a polar grid with n_a azimuthal and n_r radial subdivisions (up to radius r) is inscribed (Fig. 3(b)). For each bin of this grid, the amount of points that project into it are counted. All these values are stored in a $(n_A \cdot n_E) \times (n_a \cdot n_r)$ matrix A. Finally, the singular value decomposition $A = U\Sigma V^T$ is applied to A, and the first columns of U and V respectively are concatenated into the $(n_A \cdot n_E + n_a \cdot n_r)$-dimensional M2DP descriptor.


(a) NBLD domain and subdivision.


(b) 2D pattern used in M2DP projections.



(c) Our 3D convolutional neural network (CNN) architecture. The input 3D patch is convolved twice and delinearized with a ReLU each time. The output is L2-pooled and fed to two fully connected layers.

Figure 3: Visualizations for the used descriptors.

3.3.2 Convolutional Neural Network (CNN) descriptor

Inspired by the recent success of CNNs in place recognition, we adapt a 3D CNN similar to the ones proposed in [20] and [40]. We have explored several network architectures and present here the one with top performance. First, the points inside a cubic domain Ω are transformed into a 32x32x32 regular voxel grid X in either of two ways: *1) binary occupancy grid*: each voxel has a binay value, which is 1 if it has any points inside, and 0 otherwise. *2) Binary density comparison grid*: similarly to NBLD, the point density of each voxel is compared to each adjacent voxel in each axis, and the comparison results are converted into binary values. The three axes in which the comparison can be made are represented as three channels of X. Our CNN has two 3D convolutional layers, each followed by a rectified linear unit (ReLU), a pooling layer and two fully connected layers to map from the voxel grid X to a 512 dimensional descriptor, as depicted in Fig. 3(c). In particular, we use L_2 pooling for the pooling layer, which has been shown to perform better than max pooling [32]. We describe the way in which we train the neural network in Section 4.

3.4 Descriptor matching and place voting

Similarly to image-based place recognition we equate a "place" to an image frame. Thus, we ultimately use our descriptors to match poses from which images were taken, using a simple voting-based method [12]: first, we build an inverted index of the image frames that can be retrieved as matching places. We call these the *database frames* $\{d\}$. A kd-tree is populated with all descriptors extracted from the database sequence, and to each entry we add a reference to all image frames that have observed the corresponding keypoint $\vec{p}$. As is common practice [2, 7], we use principal component analysis (PCA) to reduce the dimensionality, which results in more efficient matching. The image frame references are obtained from DSO.

A place query is then executed by identifying an image frame from the query sequence which is considered the *query frame* q. Again, we can extract what keypoints $\{\vec{p}\}$ are observed by q from DSO. Of these keypoints, the patches are extracted and described. For each resulting descriptor we match the closest descriptor with respect to the ℓ_2 distance from the database kd-tree. No threshold needs to be applied to the matching.

The matched descriptor now votes for the database frame to be matched to the query frame: each matched descriptor casts a vote for each frame from which its keypoint $\vec{p}$ has been observed. Finally the vote count that each database frame d receives is defined as $v(d)$, and the database frame with the highest $v(d)$ is matched to the query frame q, provided $v(d)$

exceeds a vote count threshold λ_v.

4 Training the CNN

Given the place matching method, how should we train the CNN for outputting meaningful descriptors? Patches obtained at the same place should result in descriptors with a low ℓ_2 distance, while patches obtained at different places should have a higher distance between them. This can be achieved with a triplet loss function L as presented in [31] and used in [2, 16, 40]. A triplet is defined as a collection of 3 voxel grids (X_1, X_2, X_3) where X_1 and X_2 are the voxel grids of matching patches from the training data, and X_3 is the voxel grid of a non-matching patch. The triplet loss function is defined as:

$$L(X_1, X_2, X_3) = \max(\| D(X_1) - D(X_2) \|_2 + m - \| D(X_1) - D(X_3) \|_2, 0)^2, \tag{1}$$

where $D(X)$ is the descriptor of X and the margin m is a tunable hyperparameter. The triplet loss is 0 if the distance between the matching descriptors is lower than the distance between the non-matching ones by at least a margin m.

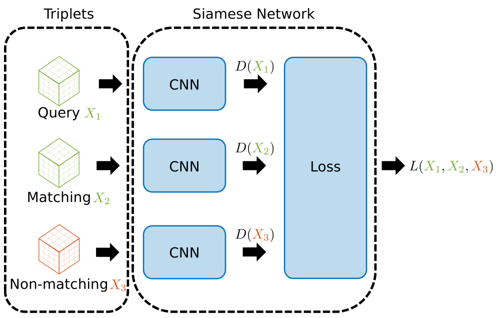
In order to back-propagate from this loss, we set up a so-called *Siamese architecture*, since several forward passes from the network are required to compute a single cost value. As illustrated in Fig 4(a), this architecture is constituted by three instances of the descriptor CNN shown in Fig. 3(c) with shared weights. These instances convert each of (X_1, X_2, X_3) into their descriptors, and feed the descriptors into the differentiable loss function L.

4.1 Generating Training Data

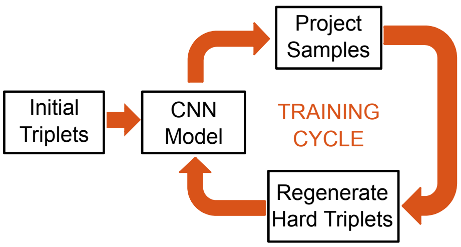
In order to obtain the triplets (X_1, X_2, X_3) from the training data, we need a method to reliably generate corresponding and non-corresponding matches. Naively, corresponding patches would be obtained from centering them around exactly the same location across the different datasets. The problem with this is that while ground truth camera poses are provided, there is no guarantee that DSO will triangulate the same points in consistent locations across sequences. In fact, the trajectory estimate of DSO is not perfectly aligned with the ground truth in the first place, and it sometimes loses track. Thus, we first align the trajectory estimated by DSO to the GPS data. At this point, we extract from each sequence local point clouds around the same geo-referenced point that is given by a random keypoint in one of the sequences. Before extracting the patches Γ that are considered to be corresponding, the local point clouds are aligned using iterative closest point (ICP). If the root-mean-square error after ICP exceeds a threshold λ_{ICP}, we reject the patches. ICP alignment is done to counteract any inconsistencies present between the DSO maps obtained for different sequences. Next, the patches Γ are extracted from the aligned local point clouds, and normalized in orientation using the method presented in Section 3.2. Finally, the voxel grid representation X is extracted from the aligned Γ. This allows us to extract N corresponding voxel grids, where N is the amount of sequences in which the patch is observed. With this, we have enough corresponding voxel grids for $\binom{N}{2}$ triplets, which provides us with plenty of training samples for each randomly chosen point. To generate a non-corresponding patch to append to a corresponding pair, we extract Γ at a randomly selected position, which is far away from the location of the corresponding patches.

4.2 Batch Sampling Strategy

To train the network, we perform stochastic gradient descent with a batch B of 25 triplets. Concretely, a batch B is built of random samples from a triplet set T of cardinality $20'000$,



(a) In order to back-propagate the triplet loss L (1), we apply a Siamese network in which our CNN is applied 3 times with shared weights for two matching patches and one non-matching patch.



(b) We use a batch sampling strategy which preferentially samples triplets that exhibit large loss, thus accelerating the learning process.

Figure 4: Illustrations of the concepts used in the training of our CNN.

generated through the method described in the previous section.

Inspired by boosting, multiple works have recently reported the benefits of a hard negative mining policy, where the optimization is focused on samples with high loss [16, 24, 33]. However, as proposed by [16], we do not generate difficult samples in a pre-processing step, but we compute them on the fly. The advantage of this approach is that the optimizer points its attention on *currently* mis-classified samples, *i.e.*, on *hard* triplets. Those are defined as (X_1, X_2, X_3) triplets in which X_3 is one the nearest neighbours of X_1 in descriptor space, even though they should not be matched.

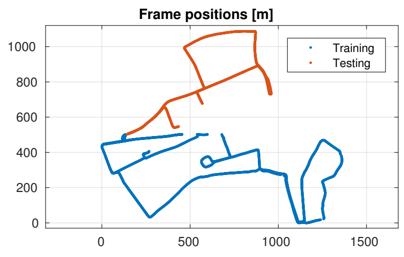
To prevent the optimizer from spending too much effort on correctly classifying noisy examples, henceforth over-fitting, we substitute every $K = 2$ epochs only 5% of the training samples with *hard* triplets, produced from the same training dataset. This cycling strategy, depicted in Fig. 4(b), experimentally proved to reduce training times while drastically increasing performance, confirming results reported in previous work.

5 Experiments

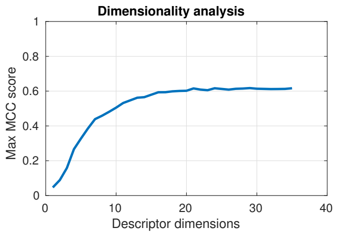
5.1 Dataset

For all of our experiments, we employ the challenging *Oxford RobotCar Dataset* [19]. This contains a large set of image sequences recorded by a car traversing the same path several times over the course of more than a year. Several sequences were recorded for each season, allowing to test the localization performance on a large scale. The dataset was chosen because of its high variance in places' visual appearance, as shown in Fig 1. Strong visual changes result in fact from the presence of occluders, from difference in illumination and viewpoint conditions, and eventually from environment variations across seasons.

To account for all seasons while restricting ourselves to a manageable amount of data, we select 11 image sequences, with at least 2 of them for each season. Each sequence has a spatial extent of around 10km, and is divided into geographically non-overlapping training and testing sets, as shown in Fig. 5(a). The training splits from all chosen sequences are joined together to create a single large training set. This training set is used for training the proposed CNNs and for fine-tuning a visual place recognition baseline, see below.



(a) The spatial extent and geographical training-testing split, illustrated on the 15-05-19-14-06 sequence.



(b) MCC for different descriptor dimensions, evaluated on the summer-winter match (Fig. 6(d)) of the CNN.

Figure 5: Training-testing split and dimensionality analysis with the Matthews Correlation Coefficient (MCC).

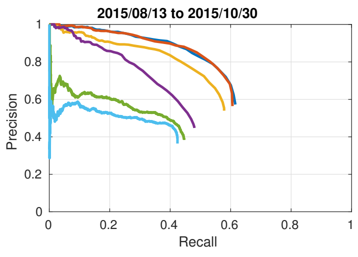
		tp	fp	fn
Database place d is matched to query q.	If $\lvert\vec{p}_q - \vec{p}_d\rvert \leq r_e$	1	0	0
	Else, if $\exists d' \in D, \lvert\vec{p}_q - \vec{p}_{d'}\rvert < r_e$	0	1	1
	Otherwise	0	1	0
No database place is matched to query q.	If $\exists d' \in D, \lvert\vec{p}_q - \vec{p}_{d'}\rvert < r_e$	0	0	1
	Otherwise	0	0	0

Table 1: True positives, false positives and false negatives evaluated for each query place q. Here, $\vec{p}_i$ are ground truth positions.

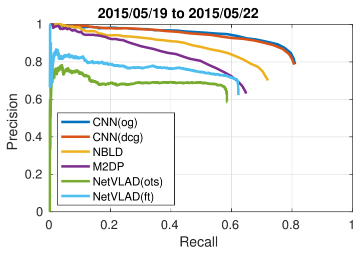
5.2 Evaluating Place Recognition Performance

To evaluate the performance of our methods, we match pairs of testing sequences against each other. From all the possible pairs, we selected 10 of them, in such a way that one sequence for each season would be matched not only against another one from the same season, but also against one from each other season. For each evaluated pair, precision and recall are computed using the (true positive, false positive, false negative) case distinction presented in Table 1, where the GPS distance threshold r_e is set to $25m$.

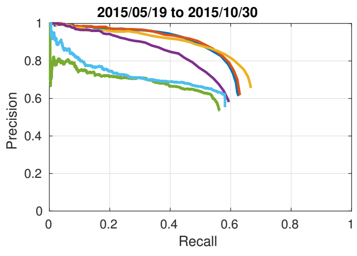
In the following, we report results when using NBLD, M2DP and our CNN in the pipeline described in Section 3. Before evaluation, all of the above descriptors are projected down to a 25-dimensional space via PCA. Fig 5(b) reports an evaluation of the CNN descriptor at different target dimensions. Heuristically, we have observed the dimensionality of 25 to be the best trade off between performance and computational complexity. Furthermore, we compare to NetVLAD [2] as a state-of-the-art image matching based method. The NetVLAD descriptor, however, is projected with PCA to a 128 dimensional space, as it was reported by the authors to be one of the most performing projection dimensions. We evaluate NetVLAD for two sets of weights: the provided off-the-shelf weights (VGG-16 + NetVLAD + whitening, Pittsburgh) and a set of weights obtained by fine-tuning the off-the shelf weights on our training data for 8 epochs. Given the large size of our training dataset, this corresponded to 80 hours. For the fine-tuning, the VGG-16 part of the network was fixed.



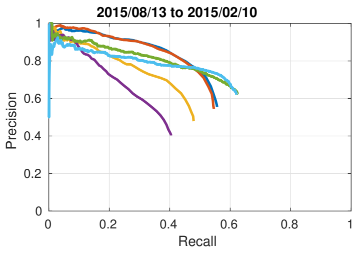
(a) Summer-fall



(b) Spring-spring



(c) Spring-fall



(d) Summer-winter

Figure 6: Precision-recall curves for the evaluated methods (our CNN with *og*: occupancy grid, *dcg*: density comparison grid, the hand-crafted descriptors NBLD and M2DP as well as the baseline image matching method NetVLAD with *ots*: off-the-shelf and *ft*: fine-tuned weights). Learning-based descriptors use input patches of size 20x20x20m^3, NBLD uses a radius of 10m and a height of 20m. We randomly select 5% of the point cloud points as keypoints.

6 Results

In the 10 considered place recognition cases, we observe a wide range of behaviours for the evaluated methods. Precision-recall curves for the four most representative cases are reported in Fig. 6 while the area-under-curve (AUC) measure for all of them are reported in Table 2. In six out of ten cases, all structural descriptors outperform NetVLAD, that is, structural descriptors have consistently better precision and recall like in Figs. 6(a) , 6(c). In two out of ten cases, NetVLAD outperforms the structural descriptors in recall, although it exhibits lower precision at lower recall values, like in Fig. 6(d). Finally, there remain two cases like in Fig. 6(b), where NetVLAD partially beats the hand-crafted descriptors but not the learned ones. Overall, the learned structural descriptors, which are barely distinguishable in performance, outperform the hand-crafted ones by margin between $5-20\%$, with the one notable exception shown in Fig. 6(c). Here, NBLD outperforms the learned descriptors by a small margin in recall. We speculate that this is due to the inherent robustness of the cylindrical descriptor domain of NBLD to noise in the orientation normalization: in a cylindrical descriptor, all bins in radial direction are equally affected by noise in rotation, while in a cubic domain, such as the voxel grid used as CNN input, the outer bins are disproportionally highly affected by rotational normalization noise.

For the structural descriptors, there is a good correlation between the performance and whether the sequences stem from the same season. Interestingly, this correlation is not as strong for NetVLAD. However, we have found that out of the six pairs in which NetVLAD performs poorly, four involve the sequence 2015-10-30-13-52-14. When inspecting this

Sequence pairs	CNN (ours, og)	CNN (ours, dcg)	NBLD [7]	M2DP [11]	NetVLAD [2](ots)	NetVLAD [2](ft)
2015-05-19-14-06-38 to **2015-05-22**-11-14-30	**0.774**	0.767	0.651	0.561	0.408	0.482
2015-05-19-14-06-38 to **2015-08-13**-16-02-58	**0.736**	0.731	0.7	0.561	0.572	0.583
2015-05-19-14-06-38 to **2015-10-30**-13-52-14	0.583	0.589	**0.611**	0.52	0.396	0.427
2015-05-19-14-06-38 to **2015-02-10**-11-58-05	0.419	0.41	0.351	0.272	0.495	**0.537**
2015-08-13-16-02-58 to **2014-07-14**-14-49-50	**0.764**	0.751	0.672	0.507	0.64	0.587
2015-08-13-16-02-58 to **2015-10-30**-13-52-14	**0.557**	0.551	0.496	0.382	0.259	0.218
2015-08-13-16-02-58 to **2015-02-10**-11-58-05	0.489	0.482	0.379	0.294	**0.512**	0.496
2015-10-30-13-52-14 to **2014-11-28**-12-07-13	**0.599**	0.579	0.454	0.329	0.003	0.002
2015-10-30-13-52-14 to **2015-02-10**-11-58-05	**0.443**	0.434	0.351	0.28	0.069	0.078
2015-02-10-11-58-05 to **2014-12-12**-10-45-15	0.594	**0.597**	0.491	0.364	0.138	0.158

Table 2: The area under the precision-recall curve (AUC) for the evaluated sequence pairs (*query* to *database*) and methods (*og*: occupancy grid representation, *dcg*: density comparison grid representation, *ots*: off-the-shelf, *ft*: fine-tuned).

sequence, we found that it was captured with exceptionally low exposure when compared to the other sequences. Low exposure seems to disproportionally affect NetVLAD more than the structural descriptors. Fine-tuning NetVLAD on our training data gives it a performance boost in six out of ten cases, but does not affect much its performance relative to the structural descriptors. In four instances, the fine-tuning even has a negative effect on NetVLAD's performance, indicating that the off-the-shelf weights generalize very well to the Robotcars dataset.

7 Conclusion

In this paper, we have presented a novel approach for vision-based place recognition using semi-dense point clouds. Moreover, we have presented a novel approach to learn powerful descriptors in a data-driven manner, showing large improvements over hand-crafted feature based models. Throughout several experiments, we have shown that the information carried by semi-dense 3D point clouds can provide an improvement in robustness over image based approaches, in particular when tested on challenging conditions. Future work could explore the possibility of combining our methodology with appearance-based methods in order to achieve further robustness in long-life, large-scale localization.

8 Acknowledgments

This research was funded by the DARPA FLA Program, the National Center of Competence in Research (NCCR) Robotics through the Swiss National Science Foundation and the SNSF-ERC Starting Grant.

References

[1] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[2] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European Conf. on Computer Vision (ECCV)*, 2006.

[4] Michael Bosse and Robert Zlot. Place recognition using keypoint voting in large 3d lidar datasets. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2013.

[5] Jane Bromley, James W. Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. *IJPRAI*, 1993.

[6] Zetao Chen, Obadiah Lam, Adam Jacobson, and Michael Milford. Convolutional neural network-based place recognition. *arXiv preprint arXiv:1411.1509*, 2014.

[7] Titus Cieslewski, Elena Stumm, Abel Gawel, Mike Bosse, Simon Lynen, and Roland Siegwart. Point cloud descriptors for place recognition using sparse visual information. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016.

[8] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conf. on Computer Vision (ECCV)*, 2014.

[9] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2017.

[10] Abel Gawel, Titus Cieslewski, Renaud Dubé, Mike Bosse, Roland Siegwart, and Juan Nieto. Structure-based vision-laser matching. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2016.

[11] Li He, Xiaolong Wang, and Hong Zhang. M2dp: A novel 3d point cloud descriptor and its application in loop closure detection. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2016.

[12] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. *European Conf. on Computer Vision (ECCV)*, 2008.

[13] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[14] Edward Johns and Guang-Zhong Yang. Generative methods for long-term place recognition in dynamic scenes. *International Journal of Computer Vision*, 2014.

[15] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Int. Conf. on Computer Vision (ICCV)*, 2011.

[16] Antonio Loquercio, Marcin Dymczyk, Bernhard Zeisl, Simon Lynen, Igor Gilitschenski, and Roland Siegwart. Efficient descriptor learning for large scale localization. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2017.

[17] David G Lowe. Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 2004.

[18] Simon Lynen, Michael Bosse, Paul Furgale, and Roland Siegwart. Placeless place-recognition. In *3D Vision (3DV), 2014 2nd International Conference on*, 2014.

[19] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000km: The oxford robotcar dataset. *The Int. Journal of Robotics Research*, 2016.

[20] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[21] Michael J Milford and Gordon F Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2012.

[22] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[23] Rohan Paul and Paul Newman. Fab-map 3d: Topological mapping with spatial and visual appearance. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2010.

[24] James Philbin, Michael Isard, Josef Sivic, and Andrew Zisserman. Descriptor learning for efficient retrieval. In *European Conf. on Computer Vision (ECCV)*, 2010.

[25] Henri Rebecq, Guillermo Gallego, and Davide Scaramuzza. EMVS: Event-based multi-view stereo. In *British Machine Vision Conf. (BMVC)*, September 2016.

[26] Henri Rebecq, Timo Horstschaefer, Guillermo Gallego, and Davide Scaramuzza. Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real-time. *IEEE Robotics and Automation Letters*, 2016.

[27] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Int. Conf. on Computer Vision (ICCV)*, 2011.

[28] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Micheal Beetz. Aligning point cloud views using persistent feature histograms. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2008.

[29] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *Int. Conf. on Computer Vision (ICCV)*, 2011.

[30] Johannes Lutz Schönberger, Hans Hardmeier, Torsten Sattler, and Marc Pollefeys. Comparative evaluation of hand-crafted and learned local features. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[31] Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. In *Conf. on Neural Information Processing Systems (NIPS)*, 2003.

[32] Pierre Sermanet, Soumith Chintala, and Yann LeCun. Convolutional neural networks applied to house numbers digit classification. In *Int. Conf. on Pattern Recognition (ICPR)*, 2012.

[33] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *Int. Conf. on Computer Vision (ICCV)*, 2015.

[34] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *Int. Conf. on Computer Vision (ICCV)*, 2003.

[35] Henrik Stewénius, Steinar Gunderson, and Julien Pilet. Size matters: exhaustive geometric verification for image retrieval. *European Conf. on Computer Vision (ECCV)*, 2012.

[36] Elena Stumm, Christopher Mei, Simon Lacroix, and Margarita Chli. Location graphs for visual place recognition. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2015.

[37] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In *European Conf. on Computer Vision (ECCV)*, 2010.

[38] Junqiu Wang, Hongbin Zha, and Roberto Cipolla. Coarse-to-fine vision-based localization by indexing scale-invariant features. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 2006.

[39] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[40] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.