Augmenting Visual Place Recognition with Structural Cues

Amadeus Oertel, Titus Cieslewski, and Davide Scaramuzza

***Abstract*—In this paper, we propose to augment image-based place recognition with structural cues. Specifically, these structural cues are obtained using structure-from-motion, such that no additional sensors are needed for place recognition. This is achieved by augmenting the 2D convolutional neural network (CNN) typically used for image-based place recognition with a 3D CNN that takes as input a voxel grid derived from the structure-from-motion point cloud. We evaluate different methods for fusing the 2D and 3D features and obtain best performance with global average pooling and simple concatenation. On the Oxford RobotCar dataset, the resulting descriptor exhibits superior recognition performance compared to descriptors extracted from only one of the input modalities, including state-of-the-art image-based descriptors. Especially at low descriptor dimensionalities, we outperform state-of-the-art descriptors by up to 90%.**

***Index Terms*—Recognition, Localization**

I. Introduction

PLACE recognition is a key concept for localization and autonomous navigation, especially so in GPS-denied environments. In particular, the ability to recognize a previously observed scene is a fundamental component in Simultaneous Localization and Mapping (SLAM). Here, place recognition is used to provide loop closure candidates, which can be used to compensate for accumulated drift, thereby enabling globally consistent mapping and tracking. Furthermore, place recognition can support the purpose of localization with respect to a pre-built map of the environment [1]. The de-facto standard approach involves casting its formulation as an image-retrieval problem [2] in which a query image is matched to the most similar one in a database of images representing the visual map. The 6 degree-of-freedom pose of the query camera frame can subsequently be inferred from the retrieved database image.

Traditionally, image matching has been achieved by first extracting handcrafted sparse local feature descriptors [4]–[6]. Related research has focused on various ways to efficiently match such local features and to meaningfully aggregate them for generating matches between full images [2], [7]–[12]. However, challenges arise when images representing the same location are captured under strong variations in appearance [13]. These variations can be due to changes in illumination, weather conditions, and camera viewpoint. Considering the ultimate goal of achieving large-scale long-term place recognition, the problem setting is further complicated by severe seasonal changes in appearance, structural scene modifications over time, for example due to roadworks, and perceptual aliasing in repetitive environments.

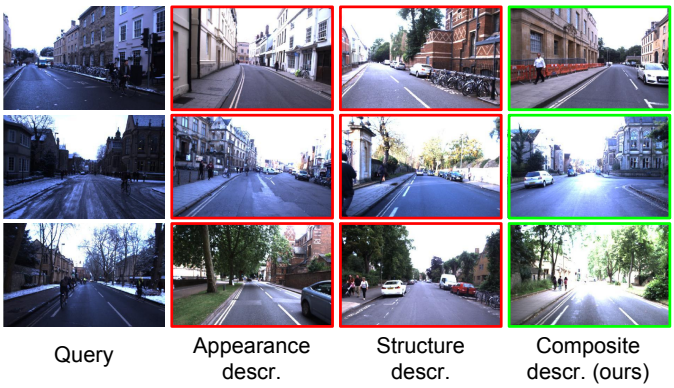


Fig. 1. To enhance visual place recognition, we propose a composite descriptor that combines both visual and structural cues – obtained using structure from motion – in a single representation. This figure illustrates three queries and the retrieved image using different types of descriptors. While descriptors that separately leverage either appearance or structure produce incorrect matches, each query is successfully matched using our composite descriptor despite changes in viewpoint and visual appearance. See the supplementary material [3] for further examples.

To cope with these challenges, recent works focus on replacing the handcrafted local features with convolutional neural networks (CNNs), which have proven useful in various object-level recognition tasks [14]–[16]. CNNs are used to either improve the discriminability of *learned* local 2D image features [17], [18] or directly learn powerful global image descriptors in an end-to-end manner [19]–[21].

However, due to the susceptibility of image-based approaches to variations in scene appearance, a different branch of research has developed local [22], [23] and global [24], [25] 3D structural descriptors to perform place recognition in 3D reconstructed maps. Most of these structural descriptors are based on maps generated from LiDAR scanners or RGBD cameras. As an exception, [22], [23] proposed a local descriptor matching approach in which learned descriptors are derived from the point cloud obtained by structure-from-motion. These structural descriptors were compared to state-of-the-art appearance-based descriptors, with mixed results – in some cases, structure outperforms appearance, while in other cases, appearance outperforms structure. Inspired by these results, we show in this paper that place recognition performance can improve significantly by learning to incorporate both visual and 3D structural features, that contain

Manuscript received: February, 24th, 2020; Revised May, 27th, 2020; Accepted June, 19th, 2020.

This paper was recommended for publication by Editor Cesar Cadena Lerma upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the National Centre of Competence in Research Robotics (NCCR) through the Swiss National Science Foundation and the SNSF-ERC Starting Grant.

The authors are with the Robotics and Perception Group, Dept. of Informatics, University of Zurich, and Dept. of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland—http://rpg.ifi.uzh.ch.

Digital Object Identifier (DOI): see top of this page.

information extracted over *sequences* of images, into a unified location descriptor. To this end, we propose a simple yet effective deep CNN architecture that is trained end-to-end on the combination of both input modalities. We demonstrate the superior matching and retrieval performance of the resulting descriptors compared to descriptors of the same dimensionality extracted from only one of the two input domains. Specifically, the structural features are obtained from image sequences using structure from motion, such that no additional sensors are needed to deploy our method in practice. We verify our method throughout several experiments by comparing its retrieval performance to state-of-the-art baselines including NetVLAD [19], DenseVLAD [26], SeqSLAM [27], and Multi-Process Fusion [28]. To the best of our knowledge, we are the first to propose learned composite descriptors that incorporate both appearance and structure for the task of visual place recognition. To summarize, we propose to derive place recognition descriptors from both appearance and structure and show that the resulting descriptor outperforms variants obtained from only one of the two modalities. Several methods for fusing visual and structural features are evaluated, and we show that of the evaluated methods, the simplest one – concatenation of globally pooled deep convolutional features – results in best performance.

II. Related Work

Our related work can be broadly divided into methods that rely on matching of either images or structural segments.

Place recognition in visual maps. Visual place recognition is typically treated as an image retrieval problem [2] where the task of recognizing places is solved by matching individual images of the same location. Classically, this has been achieved by extracting *handcrafted* sparse local feature descriptors, such as SIFT [4], ORB [6], or FREAK [5], at salient regions of the images. Subsequently, such local descriptors are typically combined to form a global descriptor for each image using aggregation methods such as bag-of-visual-words [7]–[9], VLAD [10], or Fisher Vectors (FV) [11], which allow for direct matching of the represented images. Alternatively, matches between local features can be accumulated to a match between images using nearest-neighbour voting schemes [29], [30]. State-of-the-art performance has been demonstrated using VLAD with descriptors calculated at every pixel, rather than at sparse locations [31]. See [12] for a survey of handcrafted visual place recognition methods.

Early retrieval methods that use convolutional neural networks (CNNs) have employed off-the-shelf networks, usually pre-trained on ImageNet for image classification, as black box feature extractors [32], [33]. While able to outperform retrieval based on traditional global representations, these approaches do not reach the performance of previous methods based on local descriptor matching. Therefore, subsequent works have proposed hybrid approaches based on conventional aggregation of local CNN features including FV [34] and VLAD [35], [36]. More recently, architectures have been proposed that allow the training of full image descriptors in an end-to-end fashion. These approaches typically interpret the output of a CNN as densely extracted descriptors that are subsequently aggregated with a differentiable pooling method like NetVLAD [19] or Generalized-Mean Pooling [21], resulting in state-of-the-art performance on various benchmarks.

Place recognition in structural maps. Despite advancements in CNN-based retrieval, the proneness to strong changes in visual appearance of a scene constitutes a major disadvantage of place recognition systems using single images. As an alternative, using 3D structure can offer benefits in terms of robustness to challenging environmental scene alterations and changes in illumination. As in image representations, early works propose *handcrafted* local descriptors [37]–[41], while more recent works leverage neural networks. Among the latter, two approaches to extracting structural features from point clouds have emerged: in the first approach, 3D points are aggregated into cells of a 3D grid, and features are learned using 3D convolutions [24], [42]. In the second approach, features are learned directly from the point cloud using multi-layer perceptron responses to point locations [25], [43]. All aforementioned works use dense point clouds, as extracted from specific sensors such as LiDAR scanners or RGBD cameras. In contrast, we focus on situations where only visual information is available.

It has also been shown that place recognition is possible based on the sparser structural data generated by vision-based structure-from-motion (SfM) algorithms [22], [23]. A key difference to the denser LiDAR or RGBD data, which is particularly relevant to extracting handcrafted descriptors [22], is that surface normals cannot be computed reliably. Therefore, like [42], the CNN-based approach of [23] uses features learned from a 3D grid representation. Compared to visual place recognition, they report mixed results – in some cases, structure outperforms appearance, while in other cases appearance outperforms structure.

Hence, to leverage useful features from both modalities, we propose fusing them to form a unified descriptor. This idea has very recently been applied to object detection [44] and bounding box regression [45]. To the best of our knowledge, we are the first to apply this concept to place recognition. Note that structure-from-motion represents information extracted over a sequence of images. In contrast, approaches like SeqSLAM [27] and more recently Multi-Process Fusion (MPF) [28] use image sequences directly. Specifically, MPF presents an alternative approach by fusing information from multiple image processing methods including histogram of gradients and CNN features, showing significant improvements over SeqSLAM and NetVLAD.

III. Methodology

In this section, we describe our general network architecture as well as our training methodology. In our place recognition system, an observation i of a place is composed of an image I_i and a voxel grid G_i encoding the corresponding local 3D structure. We describe the extraction of G_i from an image sequence in Section III-C. While it is possible to use alternative input representations for feeding 3D structure into a neural network, such as in PointNet [43] or PointNet++ [46], we use

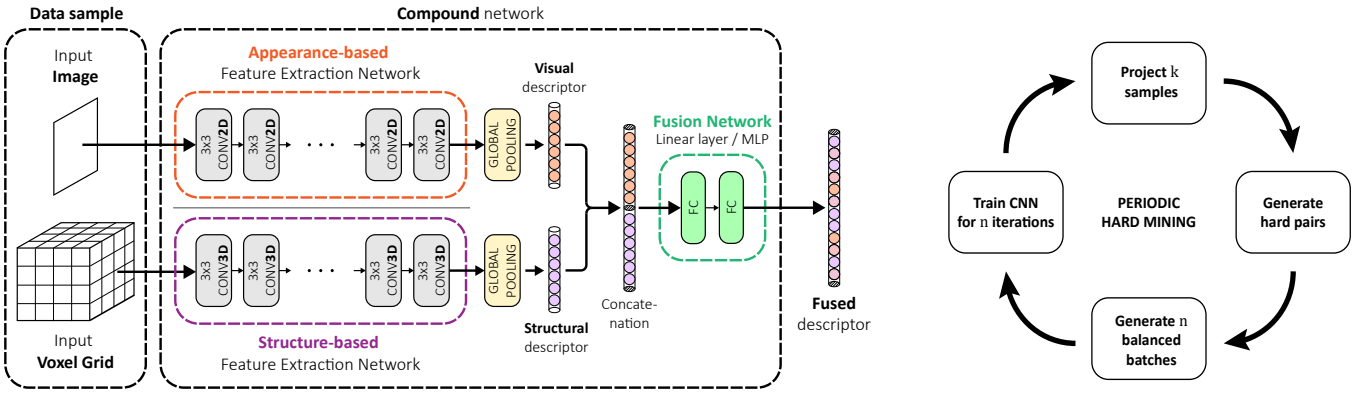


Fig. 2. (a) Overview of our proposed network architecture consisting of three building blocks including two separate CNN-based feature extraction networks followed by an optional fusion network. (b) Schematic of our hard mining policy which is a modified version of that presented in [17].

a grid representation based on the methodologies of [23], [24], [40]. Our goal is to learn the function f_θ, parametrized with a set of parameters θ, that maps I_i and G_i to a descriptor vector $f_\theta(I_i, G_i)$, such that the distance $d(f_\theta(I_i, G_i), f_\theta(I_j, G_j))$ is small if observations i and j are of the same place, and large otherwise. While any distance metric can be used as d, we have empirically found the L_1 distance to work best in our experiments. The goal is to train f_θ such that this property is maintained under strong viewpoint and natural appearance changes between different observations.

A. Architectural Details

The overall structure of our proposed *compound* network is summarized in Fig. 2(a). The three main components comprise two fully convolutional networks [47] dedicated to visual and structural *feature extraction*, each producing a descriptor specific to its modality, and an optional *feature fusion* network that combines both sets of features into a composite descriptor. We investigate different approaches to combine the intermediary descriptors ranging from simple concatenation to propagating the concatenated descriptors through a multi-layer perceptron.

Feature extraction. Features are separately extracted for the image I_i and the voxel grid G_i, respectively. The visual feature extraction network is composed of twelve 3×3 convolutional layers with stride 1 while every other layer starting with the second one is followed by non-overlapping 2×2 max-pooling with stride 2. The first six convolutional layers use 64 output channels, which is doubled to 128 for the remaining layers. The structural feature extraction network is composed analogously to its visual counterpart with the exception of using nine $3 \times 3 \times 3$ convolutional layers interspersed with 3D *average* pooling operations. For computational efficiency, the first two layers are fixed at 32 output channels. The subsequent four layers use 64 and the remaining three layers use 128 channels, respectively. Both networks apply a Rectified Linear Unit (ReLU) non-linearity to the output of every convolutional layer. Note that these are the configurations describing our best performing compound model and deviate slightly for some of the feature extraction networks evaluated separately in Section V. Refer to our supplemental material [3] for further information.

Global average pooling. As shown in Fig. 2(a), we apply global average pooling [21] to the output of each extraction network's final convolutional layer, resulting in c_f-dimensional visual and structural descriptors g_A and g_S. $c_f = 128$ for the standard configuration described above. Compared to using fully-connected (FC) layers, global average pooling results in robustness to spatial translations, reduces proneness to overfitting, and allows processing of arbitrarily-shaped inputs, which is particularly interesting for changes in camera resolution across applications [48]. We use no other form of normalization besides the division by the image resolution involved in average pooling.

Feature fusion. We evaluate and compare four different methods to obtain a unified representation from intermediary visual and structural feature vectors. The first one is a *simple concatenation*, such that the final descriptor is given by

$$f_\theta(I_i, G_i) = \left[g_A(I_i) \;\; g_S(G_i) \right] . \tag{1}$$

The second approach is a *weighted concatenation* using two learnable scalar weights w_A and w_S to rescale each of the intermediary feature vectors prior to concatenation,

$$f_\theta(I_i, G_i) = \left[w_A \cdot g_A(I_i) \;\; w_S \cdot g_S(G_i) \right] . \tag{2}$$

To allow for more flexibility in learning correlations across feature domains, the third version of our compound architecture trains a $dim_f \times 2c_f$ *linear projection* W_c of the concatenation, as defined by

$$f_\theta(I_i, G_i) = W_c \left[g_A(I_i) \;\; g_S(G_i) \right] . \tag{3}$$

Finally, we use a *multi-layer perceptron* g_{mlp} to investigate learning a non-linear mapping between concatenated features and final composite descriptor,

$$f_\theta(I_i, G_i) = g_{mlp} \left(\left[g_A(I_i) \;\; g_S(G_i) \right] \right) . \tag{4}$$

The standard configuration employs two FC layers with 256 units each, followed by ReLU activation.

B. Training Methodology

Our network is trained using a margin-based loss on training samples which are pairs of inputs labeled with whether they should result in a positive or a negative match. While some

place recognition methods are trained on samples from the database sequence used during deployment, we want our network to learn invariance to appearance conditions, so our training set needs to represent such changes in appearance. Furthermore, our goal is to obtain a system which can be deployed in different environments without the need for retraining, so we use strictly disjoint sets of training and evaluation samples. In this section, we describe our choice of loss function and the batch sampling strategy we have used for training. Training has been done using stochastic gradient descent on a single Graphics Processing Unit (GPU) – an Nvidia Titan Xp or an Nvidia RTX 2080 Ti.

Margin-based loss. We associate each observation i with a data sample $\mathbf{X}_i = (I_i, G_i)$ consisting of an image I_i and corresponding voxel grid G_i. The network learns to distinguish matching and non-matching locations by optimizing a *margin-based loss* [49] that aims at minimizing the L_1 distance $d(\mathbf{X}_i, \mathbf{X}_j)$ between a pair of mapped samples $\mathbf{X}_i, \mathbf{X}_j$ corresponding to matching observations while maximizing that of non-matching observations. With the hinge loss denoted as $\ell(x) = max(x, 0)$, the margin-based loss function is given by

$$\mathcal{L}\big(y_{ij}, \mathbf{X}_i, \mathbf{X}_j\big) = \ell\Big(\alpha + y_{ij} \cdot \big(d(\mathbf{X}_i, \mathbf{X}_j) - m\big)\Big), \qquad (5)$$

where $y_{ij} \in \{-1, 1\}$ denotes the ground truth of whether both descriptors should match and the parameters α and m are such that the distance between matching descriptors is nudged below $m-\alpha$ while the distance between non-matching descriptors is nudged above $m + \alpha$. Compared to the more commonly known contrastive loss [50], a margin-based loss allows matching samples to be within a certain distance of each other rather than enforcing them to be as close as possible. A very popular alternative is given by the *triplet loss* [51] which has been widely applied to different problem settings [17], [52]. However, [49] highlights the importance of sampling for deep embedding learning and shows that a simple margin-based loss is capable of outperforming other losses including contrastive and triplet loss on the task of image retrieval and a range of related tasks. Indeed, preliminary experiments using margin-based versus triplet loss resulted in faster convergence and better discriminative power, especially in conjunction with the hard mining strategy described next.

Batch sampling strategy. Uniform random sampling from all possible training pairs rapidly yields an increasing fraction of "easy" pairs that result in a loss of zero. To avoid significantly prolonged convergence times and inferior performance of the converged models, we adopt the concept of hard mining previously exploited in similar learning-based settings [17], [23], [53]. In hard mining, training is focused on samples with high loss. "Hard" samples can be determined using forward passes, which are computationally cheaper than training passes. In our particular pair-based training setup, we exploit the fact that the loss of x^2 pairs can be evaluated using only $2x$ descriptor forward passes. The set of hard samples technically changes with every iteration, but it would be prohibitive to look for the hardest pair of the entire training dataset at every iteration. Consequently, we rely on a randomized hard mining strategy, where the n hardest samples are selected from k^2 random pairs every n training iterations, see Fig. 2(b). k and n are iteratively adapted as training progresses: k is increased over time, and if there are too many samples with zero loss, n is decreased. To avoid focus on outliers and to account for the natural imbalance between true negatives and true positives, we use a balanced batch composition equally divided into hard pairs, randomly selected positive pairs, and randomly selected negative pairs. We use the largest batch size divisible by three that fits GPU memory - 12 when training the compound network and 24 when training the feature extraction networks individually.

C. Voxel Grid Representation

The goal of our setup is to rely on vision sensors only, and so we use 3D structural information extracted from the input image sequences. The reconstructed 3D point segments are discretized into regular voxel grids to represent local structural information. In principle, any kind of Visual Odometry [54] or SLAM [55], [56] framework may be used for 3D point cloud generation from image sequences. However, we are interested in exploiting the rich structural information provided by semi-dense reconstructions following the promising results reported in [23]. To this end, we use a variant of the publicly available *Direct Sparse Odometry* (DSO) framework [57] extended to perform pose tracking and mapping using stereo cameras [58]. As a direct method, DSO is able to track and triangulate all image points that exhibit intensity gradients, including edges.

Given the 3D reconstruction of an image sequence, we generate one voxel grid per image. A point cloud submap is extracted from a rectangular box centered at the camera pose at which the image was taken. We assume each submap to be aligned with both the z-axis of the world frame which can be achieved using an Inertial Measurement Unit (IMU), and with the yaw orientation of the corresponding camera pose. The size of the box is fixed in our method and needs to be adapted according to the environment in which it is used. A submap contains all points observed by DSO over the set of N preceding keyframes (ending with the frame associated with this submap) that are located within the box boundaries. Next, the submap is discretized into a regular voxel grid. We evaluate three different methods to populate the grids: with *binary occupancy* (bo), a voxel with any 3D points located inside is assigned a value of 1 and 0 otherwise. With *point count* (ptc), each voxel value is equal to the number of 3D points within that voxel. With *soft occupancy* (so), each 3D point receives a weight equal to 1.0 which is distributed among the eight nearest voxel centers using tri-linear interpolation. We compare the performance achieved when using each of these representations in Section V-A.

IV. Experiments

In this section, we describe the dataset and our evaluation methodology, and provide quantitative and qualitative results to validate our approach.

TABLE I
NUMBER OF IMAGES PER CONDITION AND SEASON PART OF OUR TRAINING AND TESTING SETS, WHICH ARE SELECTED FROM THE ORIGINAL SEQUENCES WITH TIMESTAMPS LISTED BELOW.

Train	**sun** spring / summer (8183 / 5739), **snow** (7975), **rain** (8127), **overcast** spring / summer / winter (7377 / 8118 / 7860)
Test	**sun** spring / summer / autumn (2406 / 1827 / 2416), **overcast** spring / summer / winter (2178 / 2461 / 7407), **dawn** (4851), **snow** (2450), **rain** (1949)
spring	2015-03-10-14-18-10 / 2015-05-19-14-06-38,
summer	2014-07-14-14-49-50 / 2015-08-13-16-02-58,
autumn	2014-11-18-13-20-12,
winter	2014-12-09-13-21-02 / 2014-12-12-10-45-15 / 2014-12-02-15-30-08 / 2015-02-03-08-45-10 / 2015-02-13-09-16-26

A. Dataset

While there exist several datasets for visual place recognition in challenging conditions, most of them provide only *single, isolated images* as queries. By contrast, our approach requires *sequences of images* as queries in order to reconstruct the scene. Therefore, for training and evaluation, we use sequences from the popular and challenging Oxford RobotCar Dataset [13]. Within this dataset, the same $10\,\mathrm{km}$ route through central Oxford was captured approximately twice a week over more than a year. We choose a set of ten sequences that represent the large variance in visual appearance to be expected in a long-term navigation scenario, see Table I. With approximately two sequences of the main route selected per season, our subset exhibits a large diversity in illumination, weather conditions including snow, sun, and rain as well as structural changes. Our selection effectively compresses the about 100 traversals of the original dataset while preserving its challenging characteristics for visual place recognition. We follow the common procedure of splitting each traversal into *geographically non-overlapping* training, validation, and testing segments, resulting in approximately $51\,\mathrm{K}$ training samples selected from seven sequences, $17\,\mathrm{K}$ validation, and $24\,\mathrm{K}$ testing samples chosen across ten sequences. Images are cropped to remove the hood of the car, and downscaled by a factor of two when used as visual CNN input (but not when used as DSO input). Furthermore, fully overexposed images and long sequences during which the car is stationary are discarded to ensure tracking stability of DSO. The discretized volume around each camera pose is fixed at $40 \times 40 \times 20\,\mathrm{m}$ with a grid resolution of $96 \times 96 \times 48$ voxels.

B. Evaluation Methodology

Our experiments evaluate place recognition based on pairwise matching across the sequences in our testing set. All 45 unique sequence combinations are taken into account. Each model is evaluated in terms of exhaustive pairwise matching and nearest-neighbor retrieval. To evaluate a model on a given sequence pair, we loop over the images of the first sequence while the images of the second sequence are used to build the database. The best-performing model is selected based on the results obtained on the validation split while the testing set is used exclusively to obtain the final results.

Exhaustive pairwise matching. Given a sequence pair, we evaluate how well a model discriminates between matching and non-matching descriptor pairs within the set of *all* possible pairs where one descriptor comes from the query sequence and the other from the database sequence. Each descriptor pair is classified into *should* and *should-not* matches: they should be matched if they represent locations with relative ground truth distance of less than $5\,\mathrm{m}$ and heading difference of less than $30\,\mathrm{degrees}$. In contrast, pairs with relative ground truth distance larger than $20\,\mathrm{m}$ between the associated locations should not be matched. Descriptor pairs corresponding to locations with relative distance between $5\,\mathrm{m}$ and $20\,\mathrm{m}$ and any relative heading can but do not have to be matched. A descriptor pair is deemed to match if their L_1 distance is lower than a predefined threshold $d_{emb,th}$. Consequently, matched descriptors are categorized into true and false positives while not matched descriptors are categorized as either true or false negatives. This allows for computation of precision-recall (PR) curves parameterized by $d_{emb,th}$. These are summarized using mean average precision (mAP), which is equivalent to the area under the PR curve.

TABLE II
COMPARISON OF MEAN AVERAGE PRECISION (MAP) AND RECALL@1 BASED ON 128-D STRUCTURAL DESCRIPTORS FOR VARYING DEPTH d_S OF THE FEATURE EXTRACTION NETWORK AND GRID REPRESENTATIONS.

Depth d_S	mAP [-]			Recall@1 [%]		
	bo	so	ptc	bo	so	ptc
6	0.883	–	–	93.9	–	–
8	0.901	–	–	<u>94.5</u>	–	–
9	<u>0.905</u>	0.741	0.756	<u>94.5</u>	90.4	89.5
10	0.879	–	–	93.1	–	–
12	0.867	0.759	0.731	92.3	90.5	89.3

Recall@1 – Nearest-neighbor retrieval. To verify the utility of the various descriptors, we follow the common procedure [19], [59] to evaluate retrieval by looking at the N nearest neighbors among all database descriptors for a given query descriptor. It is deemed correctly recognized if there is at least one descriptor within the N retrieved ones with associated ground truth distance below $20\,\mathrm{m}$. For each sequence pair, we iterate over all descriptors of the query sequence and compute recall@N as the percentage of correctly recognized query descriptors. Since some of the recorded sequences show deviations from the main route, we only consider query descriptors for which at least one truly matching database descriptor exists. Due to place constraints, we restrict our analysis to the most difficult setting, $N = 1$.

V. RESULTS AND DISCUSSION

A. Voxel Discretization Method

We separately train the structural feature extraction network for varying layer counts d_S using the three grid representations described in Section III-C. To this end, the network learns by minimizing the margin-based loss given in (5) for structural descriptors only. Results averaged over all 45 testing sequence pairs are reported in Table II. We observe that using a binary occupancy representation achieves the best performance for all considered configurations. This is likely because the point count inside a voxel can vary depending on texture and appearance of the scene, which, unlike binary occupancy, is

TABLE III
RECALL@1, IN PERCENT, FOR DIFFERENT FUSION METHODS AGAINST VISUAL AND STRUCTURAL DESCRIPTORS, AND SEVERAL BASELINE METHODS, CLEARLY SHOWING PERFORMANCE GAINS WHEN BOTH INPUT MODALITIES ARE COMBINED. SEE THE SUPPLEMENTARY MATERIAL [3] FOR A BREAKDOWN INCLUDING EACH OF THE 45 SEQUENCE PAIRINGS.

Composite descr. (ours)				Appearance descr.	Structure descr.	NetVLAD descr. (ft)	Multi-Process Fusion	DenseVLAD descr.	SeqSLAM 40 m (20 m)
Concat	Weight. concat	Linear	MLP						
98.0	96.7	96.9	95.7	94.2	93.9	90.0	89.3	83.1	73.1 (64.5)

sensitive to seasonal changes. Hence, this representation is used in all other experiments.

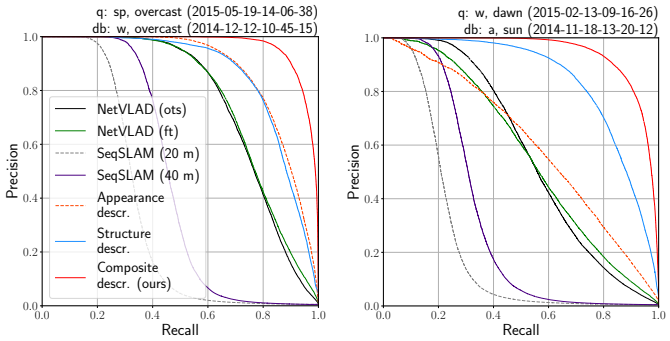


Fig. 3. PR curves from exhaustively matching two sequence pairs recorded in different seasons. Generally, superiority of using either visual or structural descriptors varies across sequence pairings. For all evaluated combinations, however, fusing both visual and structural cues using our composite representation results in significant performance gains. See the supplementary material [3] for the plots of all 45 sequence combinations.

B. Feature Fusion

To show the contribution of combining visual and structural features, we investigate the gains in retrieval using our composite descriptors over using only one of the two input modalities. Results are reported in Table III. We observe performance gains when using composite descriptors regardless of the fusion method used to generate them. In particular, we find that a simple concatenation of descriptors $g_A(I_i)$ and $g_S(G_i)$ results in the best overall performance with significant improvements over using only $g_A(I_i)$ or only $g_S(G_i)$. The fact that simple concatenation performs best could be because it forces both features to be learned – other methods can degenerate into situations where parts of the input features are ignored. Furthermore, more complicated fusion methods could be prone to overfitting, resulting in worse validation and testing performance. Still, further investigation of feature fusion methods, including more advanced methods like attention refinement [60] are interesting future work. Additionally, as exemplified in Fig. 3, we observe that for some of the sequence pairings descriptors encoding structural cues perform better compared to those encoding visual features while for other pairings, visual descriptors slightly outperform structural ones. The left of Fig. 3 shows the PR curves resulting when exhaustively matching two sequences recorded during spring and winter, respectively. Both sequences are subject to similar illumination and weather conditions and we observe that our appearance- and structure-based descriptor variants perform very similarly. To the right of Fig. 3, we provide the PR curves for a pairing again recorded during different seasons and additionally under drastically varying illumination and weather conditions. Images in the query sequence are subject to very low exposure while those in the database sequence are affected by direct sunlight resulting in greatly overexposed areas within the images. As a consequence, we observe the appearance-based variants to perform much worse compared to the structure-based descriptor. For both illustrated sequence pairings, our composite descriptor benefits from fusing both visual and structural cues into a unified representation. We highlight that for *each* of the 45 investigated sequence pairings, our proposed composite descriptor consistently outperforms all other variants.

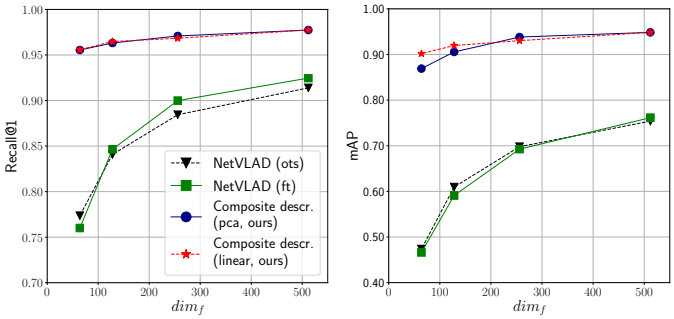


Fig. 4. Mean average precision (mAP) and recall@1 shown for varying descriptor dimensionality dim_f. Both NetVLAD variants' performance degrades significantly more severely when reducing dim_f. This results in a relative gain of up to 90.5% in mAP and 23.6% in recall@1 of our composite descriptor against NetVLAD for $dim_f = 64$.

C. Baseline Comparison

We evaluate our method against SeqSLAM [27], DenseVLAD [26], NetVLAD [19] and Multi-Process Fusion [28]. We ensure that images are spaced by $0.5m$ to avoid problems when the car stands still. We evaluate SeqSLAM sequence lengths of 40 and 80 frames (around $20m$ and $40m$) and use a linear trajectory search velocity of 0.8 to 1.2. For DenseVLAD, we train the visual vocabulary on 25 million RootSIFT descriptors extracted from the training set. The final descriptors are projected to 256 dimensions to match the dimensions of all other methods. For the evaluation of Multi-Process Fusion, we use the publicly available VGG-16 trained on Places365 and keep the suggested default parameters. The comparison in retrieval performance is given in Table

TABLE IV
AVERAGE TIMINGS FOR COMPUTING A SINGLE INSTANCE OF EACH DESCRIPTOR VARIANT MEASURED USING AN NVIDIA TITAN XP. VALUES IN PARENTHESES INDICATE COMPUTATION TIMES WHEN DESCRIPTORS ARE COMPUTED IN BATCHES.

Descriptor type	**Avg. computational cost per descriptor**
Appearance	8.96 ms (7.53 ms)
Structure	11.93 ms (10.62 ms)
Composite, concat	18.80 ms (17.50 ms)
NetVLAD	60.82 ms (46.68 ms)
DenseVLAD	1690 ms (-)

III. The reported values result from averaging the Recall@1 measured for each of the 45 sequence pairs. Note that while this would suggest that our appearance-only branch outperforms NetVLAD, these are results specific to the narrow-baseline Oxford Robotcar Dataset. On the wide-baseline VGG Oxford Buildings dataset [9], we have found for example that NetVLAD performs better than our appearance-only branch, with an mAP of 0.54 versus 0.15. Unfortunately, our training data is restricted to narrow baselines, as we have not found a dataset that provides both image sequences and a wide variety of wide baseline matches. As NetVLAD outperforms all of the other baselines, we thoroughly compare our composite descriptor to two variants of NetVLAD descriptors while also considering different descriptor dimensions dim_f. Using few descriptor dimensions can significantly boost nearest neighbour search performance and reduce memory requirements. The first version uses the publicly available *VGG-16 + NetVLAD layer + PCA whitening* off-the-shelf weights trained on Pitts30k. The second version represents the off-the-shelf weights after fine-tuning them on our training set for 15 epochs lasting approximately 70 hours on an Nvidia RTX 2080 Ti. Following [19], we use PCA to generate NetVLAD descriptors of varying dimensionality ranging between 64 and 512. Even though concatenation of $g_A(I_i)$ and $g_S(G_i)$ results in best performance, training a linear projection allows us to precisely control the number of target dimensions dim_f of our composite descriptor. Hence, we individually train both feature extraction networks modified to produce 256- instead of 128-dimensional descriptors. We then initialize our compound network using these pretrained extraction networks and continue training different weight sets by varying dim_f. By using the concatenation of $g_A(I_i)$ and $g_S(G_i)$, we obtain 512-dimensional composite descriptors. Additionally, we train three more variants each using a linear projection to control dim_f. The results are illustrated in Fig. 4, which shows that matching and retrieval performance of both NetVLAD variants severely diminishes when reducing descriptor dimensionality. For $dim_f = 256$ – the best-performing compact projection dimension reported in [19] – our composite descriptor outperforms NetVLAD by 39.1% in exhaustive pairwise matching and 8.9% in nearest-neighbor retrieval. By decreasing dim_f, our approach outperforms NetVLAD by as much as 90.5% and 23.6% in matching and retrieval, respectively. We also evaluate the projection of our descriptor using a PCA trained on the same data as the NetVLAD PCA instead of a linear projection using our training method. As seen in Figure 4, recall@1 is the same, while mean average precision is slightly worse. Note, however, that the original dimension of NetVLAD (4096) is much higher than that of our descriptor (512).

D. Computational Costs

Training the visual and structural feature extraction networks described in Section III-A using a single GPU requires approximately $35\,\mathrm{hrs}$ and $77\,\mathrm{hrs}$, respectively. The compound architecture is initialized with the resulting weights and further trained for approximately $27\,\mathrm{hrs}$. Note that a large fraction of training time is spent on batch sampling using the hard mining strategy detailed in Section III-B. Furthermore, the time spent on hard mining increases significantly as training progresses since such hard sample pairs become increasingly difficult to find. In Table IV, we further provide an overview of average inference times, including those generated by the purely descriptor-based baselines NetVLAD and DenseVLAD.

VI. Conclusion

In this paper, we have proposed to augment visual place recognition using structural cues. We have shown that a concatenation of feature vectors obtained from appearance and structure performs best among the evaluated fusion methods. Our approach is completely vision-based and does not require additional sensors to extract structure. In all of our experiments, our composite descriptors consistently outperform vision- and structure-only descriptors alike, as well as all baselines. Specifically, when comparing our composite descriptor against NetVLAD, the relative performance gain, especially at low descriptor dimensions, can be as high as 90.5% and 23.6% in exhaustive pairwise matching and nearest-neighbor retrieval. The good performance at low dimensions means that our approach is particularly well suited to fast, large-scale nearest neighbour retrieval.

References

[1] P. Sarlin, F. Debraine, M. Dymczyk, and R. Siegwart, "Leveraging deep visual descriptors for hierarchical efficient localization," in Conf. on Robotics Learning (CoRL), pp. 456–465, 2018.

[2] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," Int. J. Robot. Research, vol. 27, no. 6, pp. 647–665, 2008.

[3] "Supplementary material." http://rpg.ifi.uzh.ch/docs/RAL20_Oertel_Supplementary.pdf, 2020.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," Int. J. Comput. Vis., vol. 60, pp. 91–110, Nov. 2004.

[5] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast retina keypoint," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 510–517, 2012.

[6] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An Efficient Alternative to SIFT or SURF," in Int. Conf. Comput. Vis. (ICCV), pp. 2564–2571, 2011.

[7] J. Sivic and A. Zisserman, "Video Google: a text retrieval approach to object matching in videos," in Int. Conf. Comput. Vis. (ICCV), pp. 1470–1477, 2003.

[8] H. Jegou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation.," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 3304–3311, 2010.

[9] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching.," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), 2007.

[10] R. Arandjelović and A. Zisserman, "All about VLAD," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 1578–1585, 2013.

[11] H. Jegou, F. Perronnin, M. Douze, J. Sanchez, P. Pérez, and C. Schmid, "Aggregating local image descriptors into compact codes," IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 9, pp. 1704–1716, 2012.

[12] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," IEEE Trans. Robot., vol. 32, pp. 1–19, Feb. 2016.

[13] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," Int. J. Robot. Research, vol. 36, no. 1, pp. 3–15, 2017.

[14] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in Conf. Neural Inf. Process. Syst. (NIPS), pp. 1097–1105, 2012.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 580–587, 2014.

[16] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 3431–3440, 2015.

[17] A. Loquercio, M. Dymczyk, B. Zeisl, S. Lynen, I. Gilitschenski, and R. Siegwart, "Efficient descriptor learning for large scale localization," in IEEE Int. Conf. Robot. Autom. (ICRA), pp. 3170–3177, 2017.

[18] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 3476–3485, 2017.

[19] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 5297–5307, 2016.

[20] A. Gordo, J. Almazán, J. Revaud, and D. Larlus, "Deep image retrieval: Learning global representations for image search," in Eur. Conf. Comput. Vis. (ECCV), pp. 241–257, 2016.

[21] F. Radenović, G. Tolias, and O. Chum, "Fine-tuning CNN image retrieval with no human annotation," IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, pp. 1655–1668, July 2019.

[22] T. Cieslewski, E. Stumm, A. Gawel, M. Bosse, S. Lynen, and R. Siegwart, "Point cloud descriptors for place recognition using sparse visual information," in IEEE Int. Conf. Robot. Autom. (ICRA), pp. 4830–4836, 2016.

[23] Y. Ye, T. Cieslewski, A. Loquercio, and D. Scaramuzza, "Place recognition in semi-dense maps: Geometric and learning-based approaches," in British Mach. Vis. Conf. (BMVC), pp. 74.1–74.13, 2017.

[24] A. Cramariuc, R. Dubé, H. Sommer, R. Siegwart, and I. Gilitschenski, "Learning 3D segment descriptors for place recognition," arXiv e-prints, Apr. 2018.

[25] M. A. Uy and G. H. Lee, "PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), 2018.

[26] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla, "24/7 place recognition by view synthesis," IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, pp. 257–271, Feb. 2018.

[27] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in IEEE Int. Conf. Robot. Autom. (ICRA), pp. 1643–1649, 2012.

[28] S. Hausler, A. Jacobson, and M. Milford, "Multi-process fusion: Visual place recognition using multiple image processing methods," IEEE Robot. Autom. Lett., vol. 4, no. 2, pp. 1924–1931, 2019.

[29] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in Eur. Conf. Comput. Vis. (ECCV), pp. 304–317, 2008.

[30] S. Lynen, M. Bosse, P. Furgale, and R. Siegwart, "Placeless place-recognition," in 3D Vision (3DV), pp. 303–310, Dec. 2014.

[31] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla, "24/7 place recognition by view synthesis," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), 2015.

[32] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW), 2014.

[33] A. Babenko, A. Slesarev, A. Chigorin, and V. S. Lempitsky, "Neural codes for image retrieval," in Eur. Conf. Comput. Vis. (ECCV), pp. 584–599, 2014.

[34] F. Perronnin and D. Larlus, "Fisher vectors meet neural networks: A hybrid classification architecture," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 3743–3752, 2015.

[35] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in Eur. Conf. Comput. Vis. (ECCV), pp. 392–407, 2014.

[36] M. Paulin, M. Douze, Z. Harchaoui, J. Mairal, F. Perronnin, and C. Schmid, "Local convolutional features with unsupervised training for image retrieval," in Int. Conf. Comput. Vis. (ICCV), pp. 91–99, 2015.

[37] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS), 2008.

[38] F. Tombari, S. Salti, and L. D. Stefano, "Unique signatures of histograms for local surface description," in Eur. Conf. Comput. Vis. (ECCV), pp. 356–369, 2010.

[39] L. He, X. Wang, and H. Zhang, "M2DP: a novel 3D point cloud descriptor and its application in loop closure detection," in IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS), pp. 231–237, 2016.

[40] R. Dubé, D. Dugas, E. Stumm, J. I. Nieto, R. Siegwart, and C. Cadena, "SegMatch: Segment based place recognition in 3D point clouds," in IEEE Int. Conf. Robot. Autom. (ICRA), pp. 5266–5272, 2017.

[41] M. Bosse and R. Zlot, "Place recognition using keypoint voting in large 3D lidar datasets," in IEEE Int. Conf. Robot. Autom. (ICRA), pp. 2677–2684, 2013.

[42] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3DMatch: Learning local geometric descriptors from RGB-D reconstructions," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), 2017.

[43] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), 2017.

[44] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3D object detection," in Eur. Conf. Comput. Vis. (ECCV), pp. 641–656, 2018.

[45] D. Xu, D. Anguelov, and A. Jain, "PointFusion: Deep sensor fusion for 3D bounding box estimation," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 244–253, 2018.

[46] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in Conf. Neural Inf. Process. Syst. (NIPS), pp. 5099–5108, 2017.

[47] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, pp. 640–651, 2017.

[48] M. Lin, Q. Chen, and S. Yan, "Network in network," Int. Conf. Learn. Representations (ICLR), 2014.

[49] C. Wu, R. Manmatha, A. J. Smola, and P. Krähenbühl, "Sampling matters in deep embedding learning," in Int. Conf. Comput. Vis. (ICCV), pp. 2859–2867, 2017.

[50] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 1735–1742, 2006.

[51] M. Schultz and T. Joachims, "Learning a distance metric from relative comparisons," in Conf. Neural Inf. Process. Syst. (NIPS), pp. 41–48, 2003.

[52] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in IEEE Conf. Comput. Vis. Pattern Recog. (CVPR), pp. 815–823, 2015.

[53] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, "Discriminative learning of deep convolutional feature point descriptors," in Int. Conf. Comput. Vis. (ICCV), pp. 118–126, 2015.

[54] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect visual odometry for monocular and multicamera systems," IEEE Trans. Robot., vol. 33, no. 2, pp. 249–265, 2017.

[55] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," IEEE Trans. Robot., vol. 33, pp. 1255–1262, Oct. 2017.

[56] J. Engel, J. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in Eur. Conf. Comput. Vis. (ECCV), pp. 834–849, 2014.

[57] J. Engel, V. Koltun, and D. Cremers, "Direct Sparse Odometry," IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, pp. 611–625, Mar. 2018.

[58] J. Wu, D. Yang, Q. Yan, and S. Li, "Direct sparse odometry with stereo cameras." https://github.com/HorizonAD/stereo_dso, 2018.

[59] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt, "Image retrieval for image-based localization revisited," in British Mach. Vis. Conf. (BMVC), pp. 76.1–76.12, 2012.

[60] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in Eur. Conf. Comput. Vis. (ECCV), pp. 325–341, 2018.

VII. Supplementary material

A. PR AUC and Recall@1 Per Sequence Pairing

To provide more insights into how the various methods reported in Table III perform on each of the sequence pairings, Fig. 5 shows the precision-recall curves resulting from exhaustive pairwise matching of each sequence combination. Since Multi-Process Fusion does not provide an interface for exhaustive matching, we are not able to add it to the precision-recall plots. We further report retrieval performance for $N = 1$ on each combination of query and database sequences in Table VII. As outlined in Section V-B, we compare our composite descriptor to both visual and structural descriptors. We further evaluate against NetVLAD variants obtained using either off-the-shelf (ots) or fine-tuned (ft) weights as well as against Multi-Process Fusion, DenseVLAD and SeqSLAM for sequence lengths of $20\,\mathrm{m}$ and $40\,\mathrm{m}$, respectively. For this comparison, all descriptors including those produced by NetVLAD and DenseVLAD are projected to $dim_f = 256$ dimensions. Our composite descriptor outperforms the single-modality descriptors as well as all of the baselines in exhaustive pairwise matching of each sequence combination. On top of that, it shows best retrieval performance on all except two sequence pairings. Our results further reveal that fine-tuning of NetVLAD on our training dataset somewhat deteriorates the pairwise matching accuracy for several sequence combinations but improves retrieval performance for the majority of sequence pairings. Similar observations have been reported in [23].

B. Network Configurations

TABLE V
THE NUMBER OF OUTPUT CHANNELS AND THE ARRANGEMENT OF POOLING OPERATIONS FOR EACH OF THE d_S LAYERS OF THE STRUCTURE-BASED FEATURE EXTRACTION NETWORKS USED TO OBTAIN THE RESULTS REPORTED IN TABLE II.

d_S	# Channels per CONV-layer	Pooling indices
6	$2 \times [32], 2 \times [64], 2 \times [128]$	$2, 3, 4$
8	$2 \times [32], 3 \times [64], 3 \times [128]$	$2, 4, 6$
9	$2 \times [32], 4 \times [64], 3 \times [128]$	$2, 4, 6, 8$
10	$2 \times [32], 4 \times [64], 4 \times [128]$	$2, 4, 6, 8$
12	$2 \times [32], 5 \times [64], 5 \times [128]$	$2, 4, 7, 10$

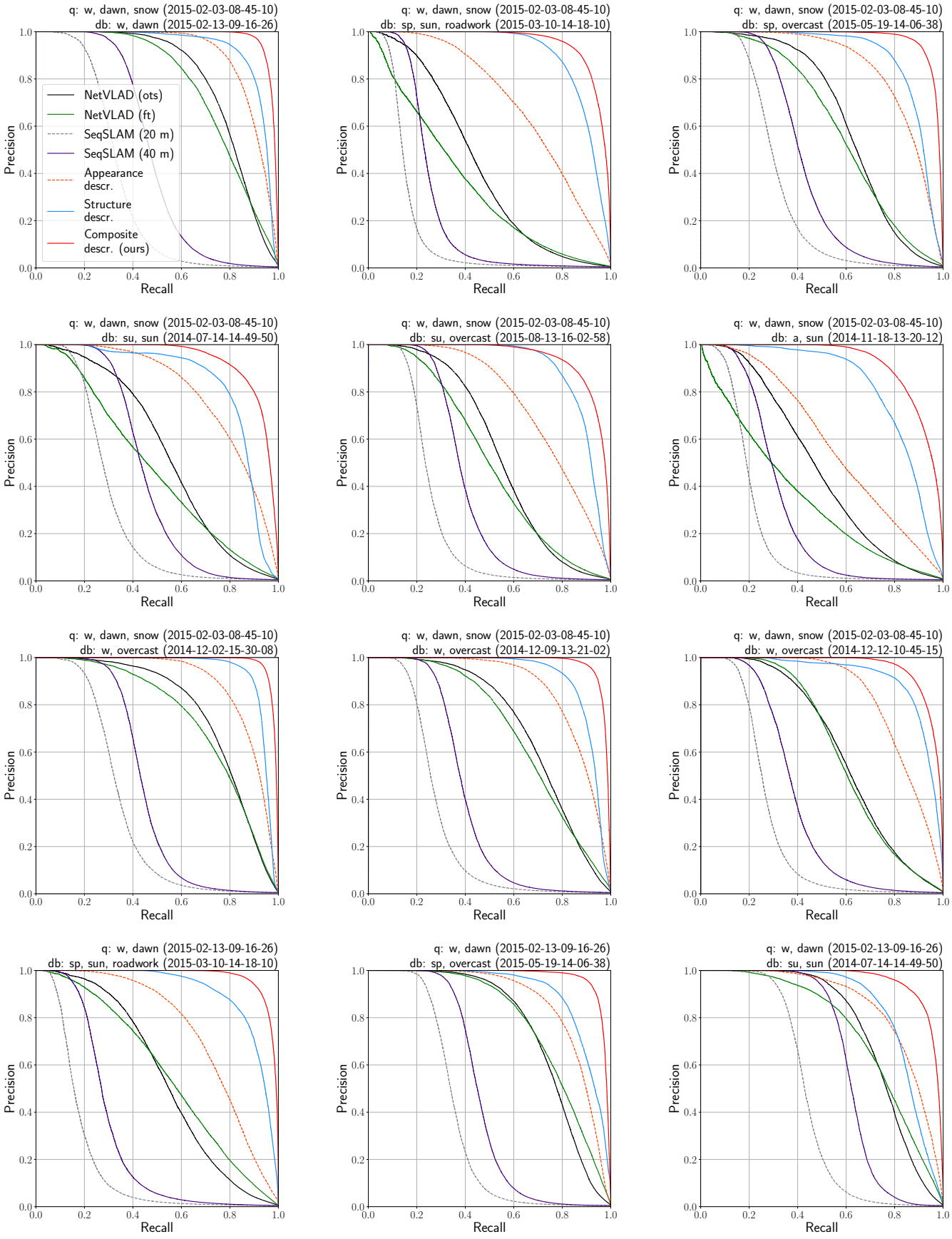
In Table V, we provide an overview of all structure-based feature extraction network configurations evaluated and compared in Section V-A. The pooling indices denote which convolutional layers are followed by pooling operations – with the networks with depths $d_S = 6$ and $d_S = 12$ not following the pattern described in Section III-A. Furthermore, Table VI shows the feature extraction configurations used in our best performing compound network as well as the slightly modified versions used to obtain the 256-dimensional appearance and structural descriptors reported in Table III.

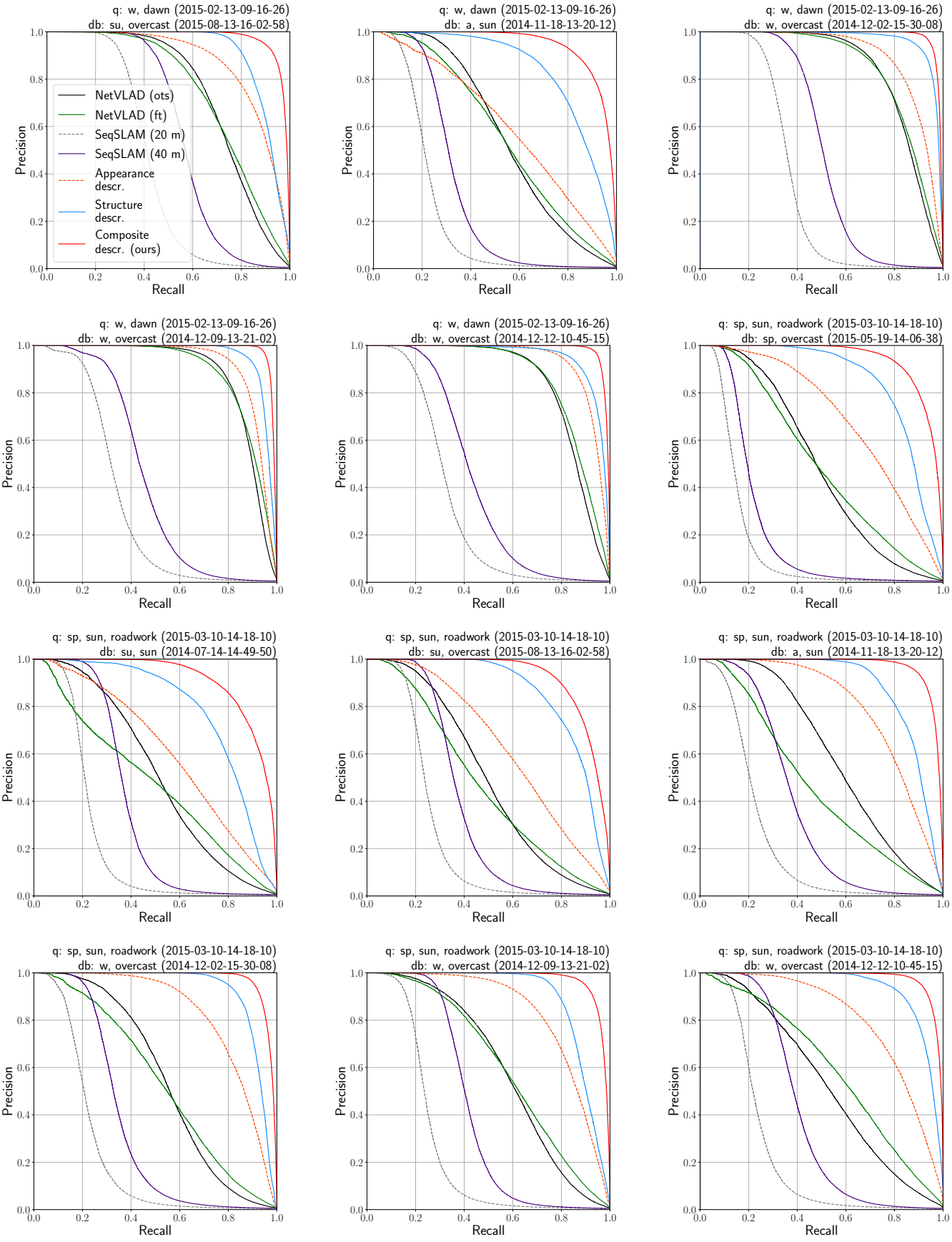
C. Image Retrieval

We provide additional examples of top retrieved database instances when using either appearance, structure, or composite descriptors, see Fig. 6. Composite descriptors are produced using the best-performing model architecture which uses $d_A = 12$ and $d_S = 9$ layers for both visual and structural feature extraction networks, respectively, and a simple concatenation of the extracted intermediary features. While descriptors that only encode either appearance or structure often produce incorrect matches, each of the shown queries is correctly matched when using our composite descriptor.

TABLE VI
OVERVIEW OF THE APPERANCE- (A) AND STRUCTURE-BASED (S) FEATURE EXTRACTION NETWORK CONFIGURATIONS REPORTED IN TABLE III. LISTED ARE THE CONFIGURATION OF OUR BEST PERFORMING COMPOUND NETWORK AS WELL AS THE SLIGHTLY MODIFIED VERSIONS USED TO GENERATE THE 256-DIMENSIONAL SINGLE-MODALITY DESCRIPTORS.

Type	d	# Channels per CONV-layer	Pooling indices
Compound network			
A	12	$6 \times [64], 6 \times [128]$	$2, 4, 6, 8, 10$
S	9	$2 \times [32], 4 \times [64], 3 \times [128]$	$2, 4, 6, 8$
Appearance- / structure-based networks			
A	12	$5 \times [64], 5 \times [128], 2 \times [256]$	$2, 4, 6, 8, 10$
S	9	$2 \times [32], 3 \times [64], 2 \times [128], 2 \times [256]$	$2, 4, 6, 8$


q: w, dawn, snow (2015-02-03-08-45-10)
db: w, dawn (2015-02-13-09-16-26)
q: w, dawn, snow (2015-02-03-08-45-10)
db: sp, sun, roadwork (2015-03-10-14-18-10)
q: w, dawn, snow (2015-02-03-08-45-10)
db: sp, overcast (2015-05-19-14-06-38)
q: w, dawn, snow (2015-02-03-08-45-10)
db: su, sun (2014-07-14-14-49-50)
q: w, dawn, snow (2015-02-03-08-45-10)
db: su, overcast (2015-08-13-16-02-58)
q: w, dawn, snow (2015-02-03-08-45-10)
db: a, sun (2014-11-18-13-20-12)
q: w, dawn, snow (2015-02-03-08-45-10)
db: w, overcast (2014-12-02-15-30-08)
q: w, dawn, snow (2015-02-03-08-45-10)
db: w, overcast (2014-12-09-13-21-02)
q: w, dawn, snow (2015-02-03-08-45-10)
db: w, overcast (2014-12-12-10-45-15)
q: w, dawn (2015-02-13-09-16-26)
db: sp, sun, roadwork (2015-03-10-14-18-10)
q: w, dawn (2015-02-13-09-16-26)
db: sp, overcast (2015-05-19-14-06-38)
q: w, dawn (2015-02-13-09-16-26)
db: su, sun (2014-07-14-14-49-50)
NetVLAD (ots)
NetVLAD (ft)
SeqSLAM (20 m)
SeqSLAM (40 m)
Appearance descr.
Structure descr.
Composite descr. (ours)
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0


q: w, dawn (2015-02-13-09-16-26)
db: su, overcast (2015-08-13-16-02-58)
q: w, dawn (2015-02-13-09-16-26)
db: a, sun (2014-11-18-13-20-12)
q: w, dawn (2015-02-13-09-16-26)
db: w, overcast (2014-12-02-15-30-08)
q: w, dawn (2015-02-13-09-16-26)
db: w, overcast (2014-12-09-13-21-02)
q: w, dawn (2015-02-13-09-16-26)
db: w, overcast (2014-12-12-10-45-15)
q: sp, sun, roadwork (2015-03-10-14-18-10)
db: sp, overcast (2015-05-19-14-06-38)
q: sp, sun, roadwork (2015-03-10-14-18-10)
db: su, sun (2014-07-14-14-49-50)
q: sp, sun, roadwork (2015-03-10-14-18-10)
db: su, overcast (2015-08-13-16-02-58)
q: sp, sun, roadwork (2015-03-10-14-18-10)
db: a, sun (2014-11-18-13-20-12)
q: sp, sun, roadwork (2015-03-10-14-18-10)
db: w, overcast (2014-12-02-15-30-08)
q: sp, sun, roadwork (2015-03-10-14-18-10)
db: w, overcast (2014-12-09-13-21-02)
q: sp, sun, roadwork (2015-03-10-14-18-10)
db: w, overcast (2014-12-12-10-45-15)
NetVLAD (ots)
NetVLAD (ft)
SeqSLAM (20 m)
SeqSLAM (40 m)
Appearance descr.
Structure descr.
Composite descr. (ours)
Precision
Recall

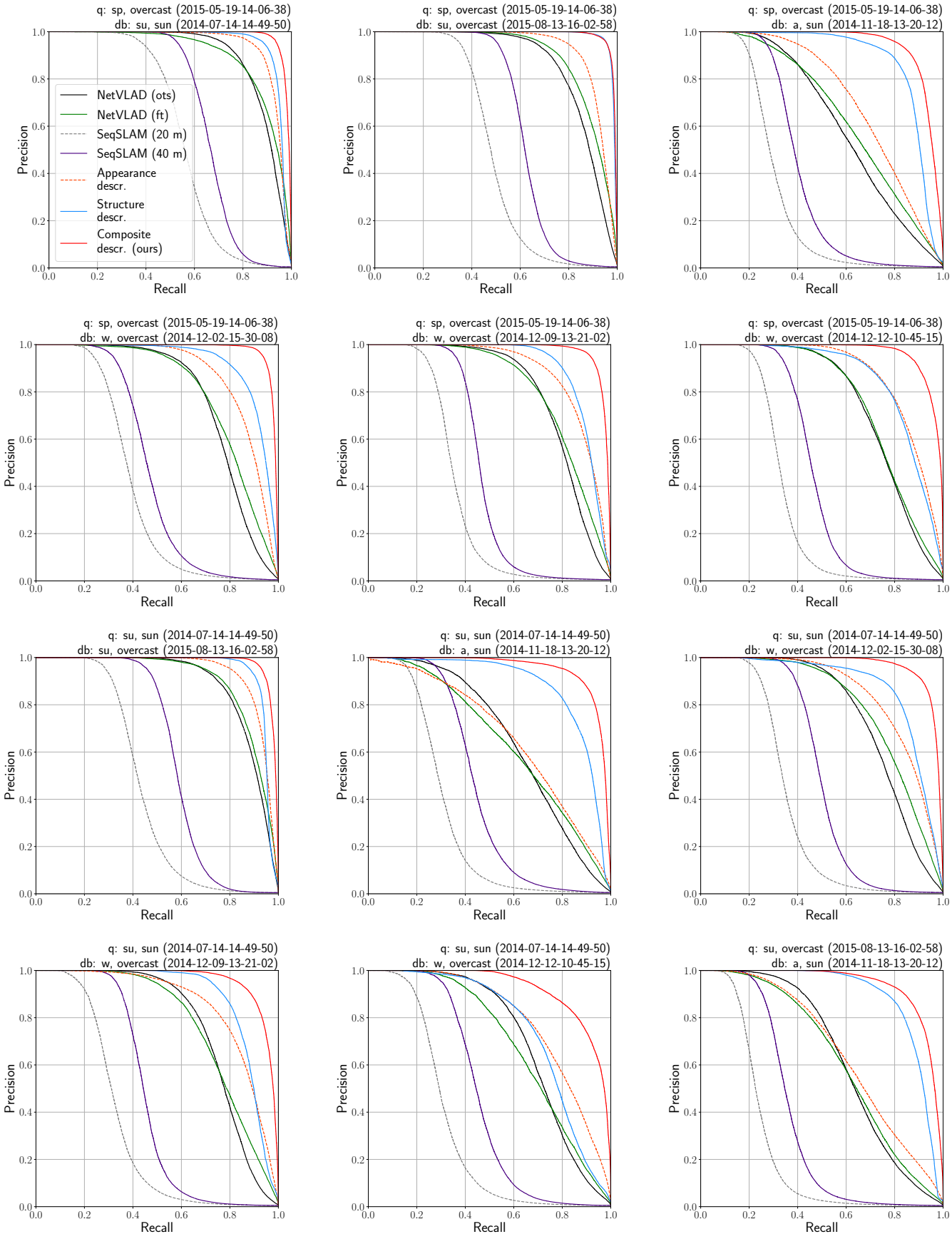

q: sp, overcast (2015-05-19-14-06-38)
db: su, sun (2014-07-14-14-49-50)
q: sp, overcast (2015-05-19-14-06-38)
db: su, overcast (2015-08-13-16-02-58)
q: sp, overcast (2015-05-19-14-06-38)
db: a, sun (2014-11-18-13-20-12)
q: sp, overcast (2015-05-19-14-06-38)
db: w, overcast (2014-12-02-15-30-08)
q: sp, overcast (2015-05-19-14-06-38)
db: w, overcast (2014-12-09-13-21-02)
q: sp, overcast (2015-05-19-14-06-38)
db: w, overcast (2014-12-12-10-45-15)
q: su, sun (2014-07-14-14-49-50)
db: su, overcast (2015-08-13-16-02-58)
q: su, sun (2014-07-14-14-49-50)
db: a, sun (2014-11-18-13-20-12)
q: su, sun (2014-07-14-14-49-50)
db: w, overcast (2014-12-02-15-30-08)
q: su, sun (2014-07-14-14-49-50)
db: w, overcast (2014-12-09-13-21-02)
q: su, sun (2014-07-14-14-49-50)
db: w, overcast (2014-12-12-10-45-15)
q: su, overcast (2015-08-13-16-02-58)
db: a, sun (2014-11-18-13-20-12)
NetVLAD (ots)
NetVLAD (ft)
SeqSLAM (20 m)
SeqSLAM (40 m)
Appearance descr.
Structure descr.
Composite descr. (ours)
Precision
Recall

TABLE VII

RECALL FOR $N = 1$ FOR EACH PAIRING OF ROBOTCAR SEQUENCES. A QUERY LOCATION IS DEEMED CORRECTLY MATCHED IF THE RETRIEVED DATABASE INSTANCE EXHIBITS A GROUND TRUTH DISTANCE OF LESS THAN 20 m. EACH SEQUENCE IS DENOTED USING ITS ABBREVIATED TIMESTAMP (*mmddss*). WE UNDERLINE THE BETTER VALUES WHEN COMPARING OFF-THE-SHELF AND FINE-TUNED NETVLAD DESCRIPTORS AS WELL AS APPEARANCE- AND STRUCTURE-BASED DESCRIPTORS. FINE-TUNING NETVLAD IMPROVES ITS RETRIEVAL PERFORMANCE FOR THE MAJORITY OF SEQUENCE PAIRINGS. SUPERIORITY OF USING EITHER VISUAL OR STRUCTURAL CUES VARIES ACROSS SEQUENCE PAIRINGS. HOWEVER, FUSING BOTH MODALITIES INTO OUR COMPOSITE REPRESENTATION RESULTS IN BEST RETRIEVAL PERFORMANCE (PRINTED IN BOLD) ON ALL EXCEPT TWO SEQUENCE PAIRINGS. THE AVERAGES ACROSS ALL SEQUENCE PAIRINGS ARE ALSO REPORTED IN TABLE III.

Query seq.	Database seq.	NetVLAD (ots)	NetVLAD (ft)	MPF	DenseVLAD	SeqSLAM (20 m)	SeqSLAM (40 m)	Appearance descr.	Structural descr.	Composite descr.
020310	**021326**	0.9669	0.9722	0.9823	0.9678	0.7555	0.8312	0.9751	0.9861	**0.9890**
020310	**031010**	0.7355	0.6743	0.7905	0.4845	0.4675	0.5471	0.9780	0.9204	**0.9906**
020310	**051938**	0.8584	0.8792	0.9156	0.7902	0.6733	0.7821	0.9306	0.9845	**0.9955**
020310	**071450**	0.8438	0.8072	0.7329	0.6287	0.6943	0.7929	0.8744	0.9542	**0.9786**
020310	**081358**	0.8473	0.8269	0.9053	0.7478	0.6162	0.6731	0.9302	0.9514	**0.9816**
020310	**111812**	0.7518	0.7241	0.7901	0.5743	0.4810	0.6031	0.9086	0.9008	**0.9776**
020310	**120208**	0.9359	0.9433	0.9626	0.8857	0.7173	0.7715	0.9792	0.9890	**1.0000**
020310	**120902**	0.9249	0.9294	0.9354	0.8849	0.6844	0.7736	0.9755	0.9820	**0.9996**
020310	**121215**	0.8429	0.8312	0.8255	0.7886	0.6830	0.7297	0.9161	0.9502	**0.9583**
021326	**031010**	0.7984	0.8126	0.8060	0.7293	0.4559	0.5784	0.9792	0.9146	**0.9875**
021326	**051938**	0.9159	0.9534	0.9744	0.9584	0.6995	0.8100	0.9525	0.9833	**0.9992**
021326	**071450**	0.8948	0.9178	0.7530	0.8864	0.8012	0.8988	0.8645	0.9492	**0.9744**
021326	**081358**	0.9159	0.9267	0.9492	0.9279	0.7246	0.8290	0.9559	0.9575	**0.9863**
021326	**111812**	0.8247	0.8555	0.8589	0.8442	0.5681	0.6667	0.9413	0.9038	**0.9775**
021326	**120208**	0.9721	0.9733	0.9933	0.9842	0.7598	0.8309	0.9988	0.9888	**1.0000**
021326	**120902**	0.9867	0.9858	0.9891	0.9913	0.6587	0.7449	0.9896	0.9925	**1.0000**
021326	**121215**	0.9012	0.9233	0.8513	0.9007	0.6603	0.6989	0.9122	0.9370	**0.9476**
031010	**051938**	0.8063	0.8687	0.8738	0.6218	0.4594	0.5517	0.9219	0.9273	**0.9788**
031010	**071450**	0.7808	0.8337	0.6978	0.5452	0.5205	0.6508	0.8887	0.8462	**0.9705**
031010	**081358**	0.7984	0.8466	0.8529	0.5777	0.5888	0.7161	0.9273	0.8807	**0.9755**
031010	**111812**	0.9252	0.9381	0.9455	0.8433	0.5450	0.6957	0.9472	0.9443	**0.9875**
031010	**120208**	0.8246	0.8865	0.9790	0.7510	0.5691	0.6896	0.9850	0.9622	**1.0000**
031010	**120902**	0.8263	0.8990	0.9547	0.7469	0.5858	0.6848	0.9618	0.9414	**1.0000**
031010	**121215**	0.8075	0.8837	0.8738	0.6839	0.6063	0.7342	0.9297	0.9110	**0.9521**
051938	**071450**	0.9579	0.9636	0.7674	0.9602	0.8697	0.8905	0.9411	0.9717	**0.9931**
051938	**081358**	0.9674	0.9848	0.9866	0.9770	0.7551	0.7899	**0.9904**	0.9766	0.9894
051938	**111812**	0.8898	0.8921	0.9272	0.8696	0.5611	0.6180	0.9284	0.9265	**0.9899**
051938	**120208**	0.9316	0.9578	0.9639	0.9490	0.7297	0.7839	0.9816	0.9729	**1.0000**
051938	**120902**	0.9500	0.9628	0.9685	0.9477	0.6746	0.7498	0.9587	0.9780	**0.9959**
051938	**121215**	0.8662	0.8758	0.8012	0.8156	0.6254	0.6887	0.8657	0.9208	**0.9369**
071450	**081358**	0.9721	0.9825	0.9840	0.9787	0.5578	0.6209	0.9628	0.9803	**0.9912**
071450	**111812**	0.8774	0.8862	0.9275	0.8101	0.5214	0.6305	0.9502	0.9119	**0.9929**
071450	**120208**	0.9321	0.9617	0.9635	0.9371	0.5482	0.5912	0.9332	0.9699	**0.9869**
071450	**120902**	0.9398	0.9480	0.9408	0.9130	0.5624	0.6599	0.9294	0.9628	**0.9863**
071450	**121215**	0.8363	0.8587	0.7582	0.7194	0.5077	0.5491	0.8065	0.8905	**0.9079**
081358	**111812**	0.8842	0.8854	0.9189	0.8306	0.5548	0.6565	0.9293	0.9021	**0.9955**
081358	**120208**	0.9290	0.9336	0.9467	0.9311	0.7309	0.8108	0.9633	0.9649	**0.9794**
081358	**120902**	0.9479	0.9587	0.9369	0.9326	0.7041	0.7778	0.9517	0.9558	**0.9826**
081358	**121215**	0.8504	0.8798	0.7718	0.7618	0.6688	0.7585	0.8669	0.9047	**0.9270**
111812	**120208**	0.8854	0.9102	0.9662	0.9043	0.6948	0.8060	0.9580	0.9257	**0.9862**
111812	**120902**	0.8764	0.8995	0.9441	0.8638	0.7611	0.8371	0.9336	0.9067	**0.9887**
111812	**121215**	0.8018	0.8212	0.8022	0.7367	0.7029	0.8166	0.8748	0.8023	**0.9141**
120208	**120902**	0.9901	0.9942	0.9962	0.9909	0.7547	0.8292	0.9954	0.9963	**1.0000**
120208	**121215**	0.9251	0.9402	0.8573	0.9156	0.7642	0.8697	0.9051	**0.9475**	0.9466
120902	**121215**	0.9316	0.9415	0.8486	0.9225	0.7861	0.8535	0.9023	0.9436	**0.9449**
Average recall@1		**0.8851**	**0.8999**	**0.8927**	**0.8314**	**0.6447**	**0.7305**	**0.9389**	**0.9421**	**0.9796**

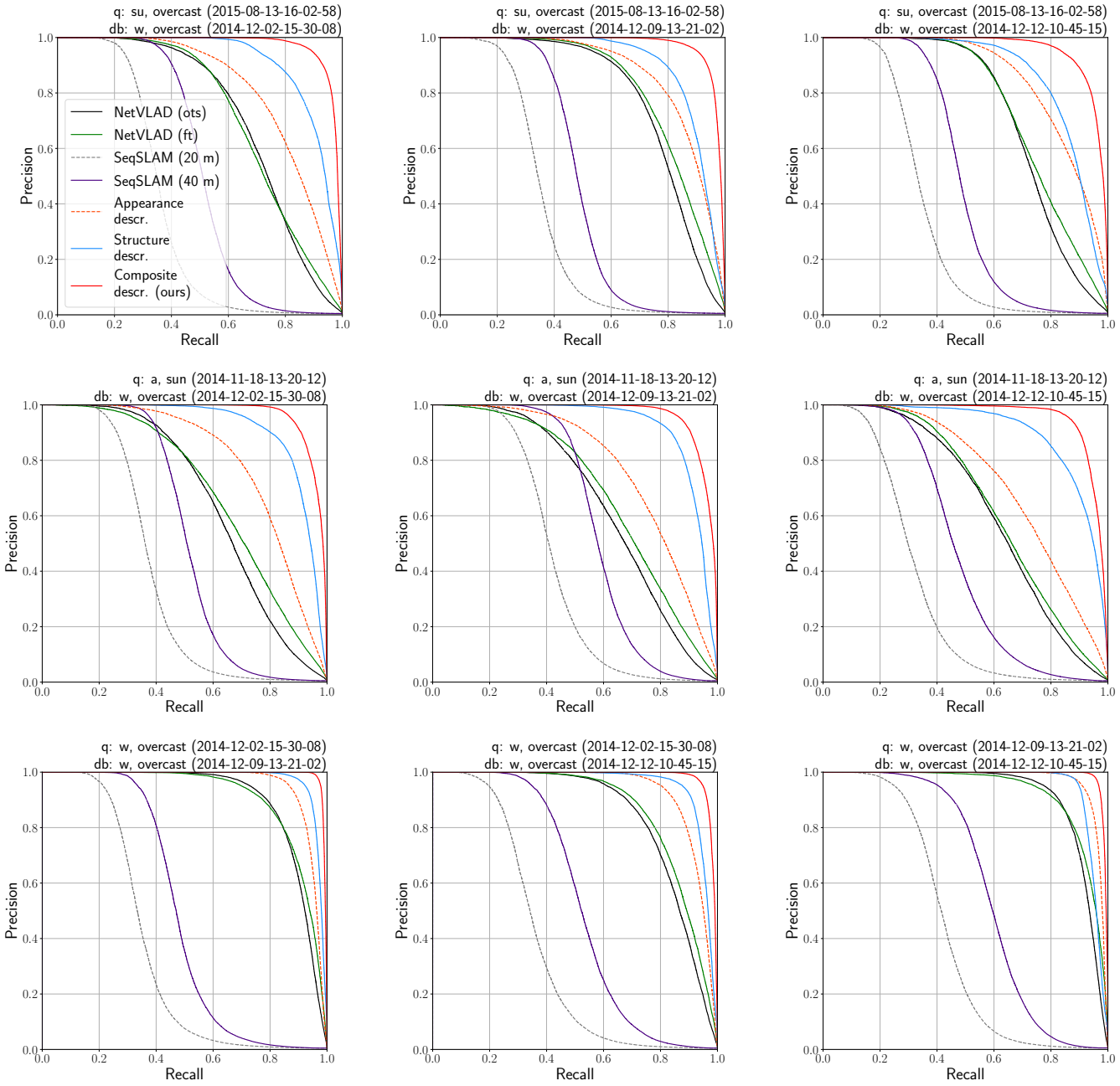


Fig. 5. Precision-recall curves for each pairing of query and database sequences out of a total of 45 sequence combinations. Each curve shows precision and recall, parametrized on the distance threshold $d_{emb,th}$ in embedding space, for exhaustive pairwise matching of query to database sequences. Each plot further details the varying conditions for both query and database sequences which are referenced using their timestamps. We compare our composite descriptors – obtained using a simple concatenation for combining intermediary visual and structural descriptors – to descriptors using only one of the input modalities as well as to off-the-shelf (ots) and fine-tuned (ft) variants of NetVLAD descriptors, as detailed in Section V-B. Specifically, pairwise matching based on our composite descriptor outperforms all other descriptors for *every* sequence combination.

Fig. 6. Examples of top retrieved database instances using appearance, structure, or composite descriptors. Only images of the retrieved instances are shown. Green and red borders indicate correct and incorrect matches. While descriptors that separately leverage either appearance or structure often produce incorrect matches, each query is successfully matched using our composite descriptor despite changes in viewpoint and visual appearance.