Robotics and Autonomous Systems 133 (2020) 103625

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Visual Place Recognition by spatial matching of high-level CNN features

Luis G. Camara *, Libor Přeučil

Czech Institute of Informatics, Robotics and Cybernetics (CIIRC), Czech Technical University in Prague, 160 00 Prague, Czech Republic

ARTICLE INFO

Article history:
Available online 25 August 2020

Keywords:
Visual Place Recognition
Convolutional neural networks
Autonomous navigation
Image retrieval
Computer vision
Deep learning

ABSTRACT

We present a Visual Place Recognition (VPR) pipeline that achieves substantially improved precision as compared with approaches commonly appearing in the literature. It is based on a standard image retrieval configuration, with an initial stage that retrieves the closest candidates to a query from a database and a second stage where the list of candidates is re-ranked. The latter is realized by the introduction of a novel geometric verification procedure that uses the activations of a pre-trained convolutional neural network. It is both remarkably simple and robust to viewpoint and condition changes. As a stand-alone, general spatial matching methodology, it could be easily added and used to enhance existing VPR approaches whose output is a ranked list of candidates. The proposed two-stage pipeline is also improved through extensive optimization of hyperparameters and by the implementation of a frame-based temporal filter that takes into account past recognition results.

1. Introduction

In recent years, research in the field of lifelong navigation of mobile robots has been particularly active [1–4]. Robust navigation under the changing conditions associated with long operation times is a requirement of truly autonomous robots. Solving this task attracts a lot of interest because it could significantly widen the range of applications where robots can be deployed, e.g. space exploration, underwater research, medicine, automotive, military, entertainment, or service robots to name a few.

Mobile robot navigation is commonly based on *Simultaneous Localization And Mapping* (SLAM) approaches [5,6]. In long-term SLAM implementations, an essential component of the system performs what is know as *loop closure* [7–10]. This is the task of deciding whether or not a robot has returned to a previously visited place and, if such is the case, using that knowledge to reduce uncertainties in the map estimate.

Specifically, mobile robots whose main sensors are cameras typically solve loop closure by *Visual Place Recognition* (VPR) [11], a fundamental and still open problem in computer vision and also the main subject of this paper. VPR is, however, a challenging task. This is especially true in uncontrolled outdoor environments and over extended periods. Images of the same place but taken at different times may differ significantly from each other. The differences can be caused by environmental factors such as differences in illumination, day–night cycles, seasons of the year, occlusions, etc. or by purely geometric ones, such as changes in viewpoint between two traverses.

The latest trend in VPR research is inspired by the great success of Convolutional Neural Networks (CNN) in several computer vision tasks. Common approaches discard their fully connected layers and use the output of their middle and latest convolutional layers to encode rich semantic information that may be robust to several image changes.

CNNs resemble in many ways biological vision [12,13]. Besides performing at similar levels as humans in several recognition tasks, they behave in a hierarchical fashion that mirrors to some extend how the visual cortex process information. Recent work comparing CNNs and some previous models of the visual system suggest that deep neural networks are the best existing approach to explain representations of spatial layout [14].

In addition, research in neuroscience [15] suggests that humans localize themselves and navigate the world by creating cognitive maps in the hippocampus, which support memory during these tasks. Localization then proceeds by associating or anchoring those maps to what is being perceived. Discrete environmental landmarks (buildings, statues, etc.), extended topographical ones (e.g. the arrangement of buildings at an intersection, a valley, a ridge, etc.) as well as object categories and textures are all involved in the anchoring process, but one of the most relevant pieces of information for localization seems to be the geometric arrangement or layout of items in the local scene [16].

* Corresponding author.
E-mail addresses: luis.gomez.camara@cvut.cz (L.G. Camara), libor.preucil@cvut.cz (L. Přeučil).
URLs: http://imr.ciirc.cvut.cz (L.G. Camara), http://imr.ciirc.cvut.cz (L. Přeučil).

https://doi.org/10.1016/j.robot.2020.103625
0921-8890/

In this work, we take inspiration from the above ideas and also follow our own intuition in that humans recognize places by first identifying semantic elements in a scene and then finding their geometric relationships [17]. Our goal is to advance the state of the art in visual place recognition under the challenging conditions likely to be present in a lifelong navigation scenario.

The main contribution of this work is two-fold:

1. We present a visual place recognition pipeline that achieves substantially better precision than current approaches on the benchmark datasets considered. It is based on the system introduced by us in [18] and represents a significant improvement and extension of it. The key to its good performance is the combination of several well-established ideas in the field, such as the use of CNN features, a frame-based temporal filter, and a geometric consistency check between images. All of them are integrated within a standard image retrieval framework consisting of (i) an image database filtering stage and (ii) a spatial matching post-verification stage, both subjected to a thorough optimization of hyperparameters.
2. We develop a simple and yet very powerful CNN-based geometric verification scheme that robustly measures similarities between images of places, even under large viewpoint and condition changes. We suggest its use as a standalone method that can be easily added to existing image retrieval-based place recognition approaches to enhance their performance.

The remaining of this paper is structured as follows. In Section 2 we review relevant literature and put our work into context. Our methodology is described in Section 3, whereas Section 4 is devoted to present, compare and discuss experimental results. Finally, Section 5 is left to conclusions and future work.

2. Related work

Visual place recognition has been traditionally approached in the literature as an image retrieval task, using Bag of (visual) Words (BoW) models [10,19–21] or the closely related VLAD [22–24] and Fisher vectors [25,26]. These approaches aggregate robust image features into previously trained dictionaries of visual words, leveraging the resulting compact representations for the creation and fast query of image databases. Nevertheless, these models are content-based and do not typically take into account spatial relationships between image features. It is for this reason that they are usually fairly robust to viewpoint changes but suffer from issues such as *perceptual aliasing* [27].

To go around this problem, some improvements of BoW have successfully taken into account geometric information. For instance, [28] improved the recognition rate of natural scene categories by dividing images into increasingly fine sub-regions and employing the BoW model in each of them, creating the so-called *spatial pyramids*. Philbin et al. [29] introduced a spatial verification step where the top-ranked results from the standard BoW model were re-ranked using spatial constraints. Similarly, the authors of [30] identified regions susceptible to perceptual aliasing and down-weighted their influence in the image representation, performing geometric re-ranking afterwords. Even with the implementation of spatial verification, some problems may arise due for instance to *geometric bursts*, defined in [31] as sets of visual elements with a consistent spatial configuration in unrelated database images. The authors showed that geometric bursts can considerably reduce the precision of a VPR system and proposed a simple approach to down-weight their influence, dramatically improving recognition results.

Previous to the resurgence of CNN models [32], most common approaches in VPR employed handcrafted features such as SIFT, SURF, ORB, etc. [33] to represent images. In recent years, however, CNN-based descriptors have shown their superiority in many computer vision tasks [34–37], including VPR. Thus, the latest body of literature in this field is largely based on CNN image representations. The works in [38,39] are some of the first examples utilizing CNNs for the VPR problem. The authors looked at different depths of pre-trained convolutional networks and observed that later layers held more semantic information and therefore were more robust under conditions of large viewpoint variance. On the other hand, middle layers were less affected by appearance changes such as illumination. Their findings paved the way for future research where several authors have demonstrated improved performance when using CNN features as compared with handcrafted ones [40–44]. A common characteristic of these works is that they utilized features extracted from pre-trained networks, whose training datasets were not necessarily related to the VPR problem.

In contrast, some authors have trained their networks specifically for VPR, demonstrating the benefits brought by this approach. For instance, [45] created a large dataset of places and trained a network that interpreted VPR as a classification problem, achieving an average 10% increase in performance over other approaches. Highly influential in VPR is the work of Arandjelović et al. [22]. They designed NetVLAD, a CNN architecture that incorporated a VLAD [46] layer and could be trained in an end-to-end fashion specifically for the place recognition task. Their results on some very challenging datasets were remarkable, significantly outperforming published results based on pre-trained CNNs. This has very recently been confirmed in [47], where a comprehensive comparison of 10 VPR systems identified NetVLAD as the best overall performing one.

In line with our two-stage proposed methodology, several authors [29–31,48,49] have explicitly considered geometric post-verification of shortlisted locations, showing that recognition can be significantly improved in this fashion. They focused however on handcrafted features. On the other hand, some recent work has concentrated on using CNNs for geometric matching of images. For example, [50] designed an end-to-end trainable CNN architecture that allowed them to learn the parameters of the transformation model between images. In [51], the authors developed a multi-step methodology to find the pose of a query photograph given a large indoor 3D map. They used the output of NetVLAD for fast initial retrieval of images and several layers of the VGG16 architecture to geometrically match them against a query. The overall structure of our pipeline resembles that of [51], although we do not perform pose estimation and focus on outdoors, long-term visual place recognition, devising our own initial image retrieval and geometrical matching schemes.

3. Methodology

The general methodology of the system presented in this paper was originally introduced by us in [18] and dubbed SSM-VPR (Semantic and Spatial Matching Visual Place Recognition). In the current work, we improve, extend and thoroughly optimize it, raising its performance to levels that, on the benchmark datasets considered and to the best of our knowledge, sets the state of the art in VPR.

As it is common in the literature [22,29,31,52], we approach the problem of VPR as an *image retrieval* task, which is usually divided into an image database filtering stage (stage I) and a re-ranking stage (stage II). Fig. 1 summarizes the workflow of the proposed recognition system. It is separated into offline and online parts. During the former, a reference image dataset of

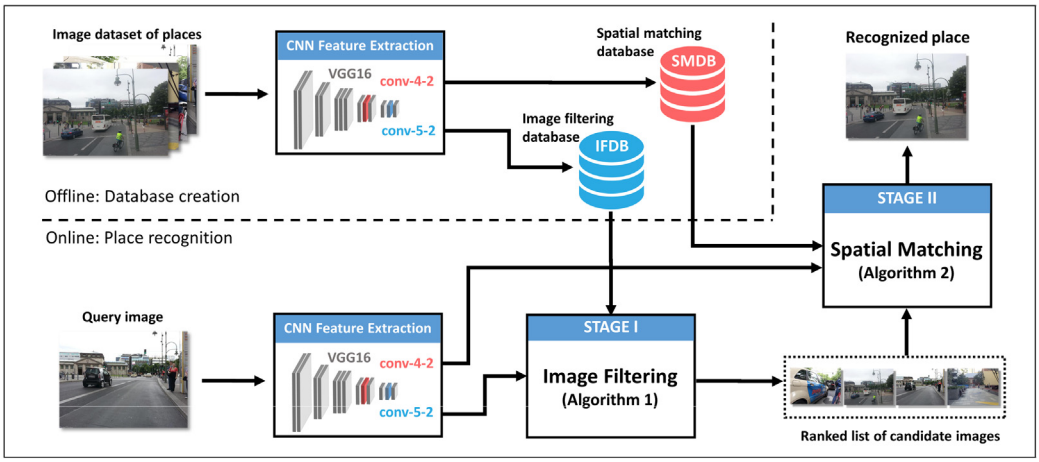

Fig. 1. Workflow of the proposed two-stage visual place recognition system.

places is passed through a VGG16 CNN architecture [35], pre-trained on the *Places365* dataset [53]. CNN features are then extracted from layers *conv 4-2* and *conv 5-2* and stored in separate databases, which are denoted, respectively, as SMDB (spatial matching database) and IFDB (image filtering database). The extraction process is described in detail in Section 3.1.

During the online part, a query image representing the place to be recognized is also passed through the network and encoded as before. Recognition then proceeds in two stages. In stage I, features extracted from layer *conv 5-2* are compared with those stored in the IFDB and a list of N candidates images, ranked by similarity with the query, selected (filtered) from it. In stage II, query image features obtained from layer *conv 4-2* are compared with those stored in the SMDB for each candidate. Based on geometrical considerations, the best matching candidate is then deemed as the recognized place. The two stages are thoroughly explained in the following sections. For convenience, we have shortened the name of our method to SSM throughout the paper. A GUI to test the methodology along with source code is freely available in our group's website.[1].

3.1. Feature extraction

The process of extracting CNN features from images varies according to the stage where they are used during recognition, which also determines the target layer and the storage database. The following discussion refers to a set of default optimized parameters, such as the image size, the size of feature map regions considered during the extraction, their number, or the final feature's dimensionality. They are experimentally determined later in this work and summarized in Table 2.

Features used in stage I are created from layer *conv 5-2*. The layer is schematically represented in the left part of Fig. 2. For the employed image size of $S_1{=}224\times224$ pixels (3 color channels), the 512 feature maps contained in this layer have an activation spatial resolution of 14×14. To create the features, activations are grouped into several convolutional 3D tensors such as the one represented by a blue cube in the figure, whose dimensions are $h\times w\times D = 9\times9\times512$. A total of $n_{c_1}{=}36$ cubes are formed by sliding along the layer's horizontal (W) and vertical (H) directions, using a stride $s=1$ activation. Taking each generated cube individually, activations at each spatial location (denoted by the grid cells in the figure) are normalized along the dimension of the feature maps, D. The resulting normalized activation sequences are

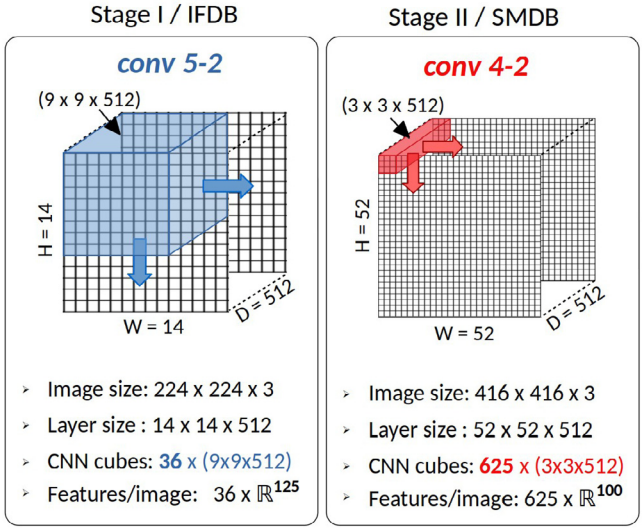

Fig. 2. Encoding details and schematic representation of the convolutional layers considered in this work. Left: encoding for stage I and the image filtering database (IFDB). Right: encoding for stage II and the spatial matching database (SMDB). Image size is in pixels. Layer size in activations.

then concatenated into a vector of size $9\times9\times512 = 41\,472$. Since working in such a high dimensional space is expensive in terms of memory and complexity, we perform dimensionality reduction through Principal Component Analysis (PCA), a common practice when working with CNN features [54]. PCA models are trained by uniformly sampling features from images in the dataset, up to a maximum of 5000 features. Their dimensionality is finally reduced to $d_1 = 125$ and the resulting vectors, 36 per image, stored in the IFDB during the offline part. All information about the spatial location of the vectors in the feature maps is discarded. During recognition, the same processing is performed on any entering query image, although on this occasion the extracted features are compared against the IFDB rather than saved.

Features used during stage II are extracted from layer *conv 4-2*. The layer is depicted in the right part of Fig. 2. We used an image size of $S_2{=}416\times416$ pixels, which as it will be discussed later in this paper it is required to maximize the performance of the spatial matching stage. This layer also contains 512 feature maps and for the chosen image size, each of them accommodates 52×52 activations. On this occasion, activations are grouped into cubes of size $3\times3\times512 = 4608$ (red cube), normalizing and

[1] http://imr.ciirc.cvut.cz/Software/Ssm-vpr.

Algorithm 1 Image filtering stage

1: $\{\mathbf{q}_i\} \leftarrow$ Compute query image vectors ($36\times\mathbb{R}^{100}$)
2: *cand_hist* ←Initialize histogram of candidate images
3: **for** $i = 1$ to 36 **do**
4: Find closest N matches of $\mathbf{q}_i$ in database IFDB
5: Increment the corresponding N bins in *cand_hist*
6: **end for**
7: Candidate list ←Select N highest bins from *cand_hist*

concatenating as before. As for PCA compression, the optimal number of dimensions was set to $d_2 = 100$ (Section 4.3). After applying a stride of $s = 2$ activations, each image is finally represented by an array of $25 \times 25 = 625$ vectors in $\mathbb{R}^{100}$. For dataset reference images, they are stored in the SMDB along with their spatial location within the array. For query images, vectors and locations are both used to compare against the SMDB during the spatial matching stage.

3.2. Stage I: Image filtering

The goal of stage I is to retrieve from a database of image features those candidates that are most similar to a query. The retrieval process should exhibit some robustness to typical image changes such as illumination or those caused by viewpoint differences. By choosing one of the latest layers in the network, features with high semantic content may provide this invariance. However, using full-image feature map representations can rapidly degrade recognition performance under changes in viewpoint, as large parts of reference images may be missing in the query or vice versa. Several authors have shown that selecting regions of interest can boost recognition performance [42,43,55]. Rather than trying to find those regions, we take a brute force approach and spatially scan the feature maps by dividing them into smaller, heavily overlapping square regions, as described in Section 3.1.

Stage I is summarized in Algorithm 1. Given a query image, feature vectors are extracted from layer *conv 5-2* and compared one by one against the IFDB. For each vector and based on the nearest neighbor distance criterion, the closest N candidates are added to a histogram of places. After considering all query vectors, the resulting accumulated histogram is used to extract a list of N top-ranked candidate images.

3.3. Stage II: Spatial matching

As discussed in the introduction, the spatial layout and geometric constrains among semantic items in images can be of vital importance in the recognition of a place. We have devised a novel, simple yet powerful geometric verification procedure that compares images by (1) finding high-level matching features between them and (2) measuring how similar their surroundings are to each other.

The matching procedure is described in Algorithm 2. Given a query image, let us denote by $\{\mathbf{q}_{k,l}\}$ the array of spatially-aware feature vectors extracted from convolutional layer *conv 4-2*. For each candidate in the list returned by Algorithm 1 for that particular query, the corresponding array of spatially-aware vectors, denoted as $\{\mathbf{c}_{i,j}\}$, is retrieved from the SMDB. Geometric verification is then performed between the two arrays as follows. For each location (i, j) in the candidate array, we find the location (k, l) in the query array that contains the best matching (closest) vector of $\mathbf{c}_{i,j}$, using the nearest neighbor criterion. We call these two locations *anchor points*. A square patch is then defined around the candidate's anchor point. Depending on the size of the patch

Algorithm 2 Spatial matching stage

1: *candidates* ← Apply Algorithm 1 on query image
2: $\{\mathbf{q}_{k,l}\} \leftarrow$ Compute spatially-aware vectors from query image
3: *scores_hist* ← Initialize score histogram for candidates
for each: *candidate* in *candidates*
4: $\{\mathbf{c}_{i,j}\} \leftarrow$ Get spatially-aware candidate vectors from SMDB
5: **for all** $i, j \in \{1 \dots 25\}$ **do**
6: $\mathbf{q}_{k,l} \leftarrow$Find closest match of $\mathbf{c}_{i,j}$ in query array
7: Set (i, j) and (k, l) as anchor points
8: Define patch around location (i, j). It may be truncated
9: Define identical (possibly truncated) patch around (k, l)
10: **for all** $k', l' \in$ {query's patch} **do**
11: $\mathbf{c}_{i',j'} \leftarrow$Find closest match of $\mathbf{q}_{k',l'}$ in candidate array
12: **if** $(i', j') - (i, j) = (k', l') - (k, l)$ **then**
13: Increment bin in *scores_hist* for *candidate*
14: **end if**
15: **end for**
16: **end for**
17: Select the candidate with the highest bin in *scores_hist*

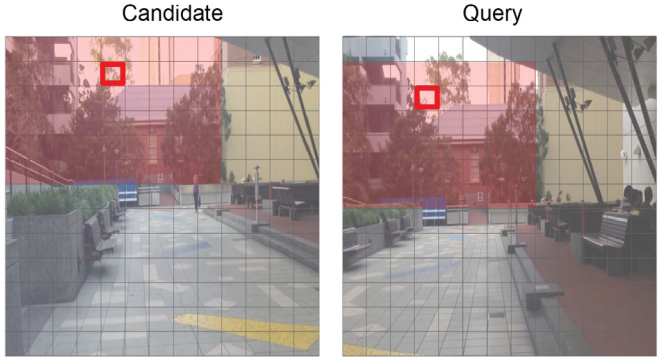

Fig. 3. Illustration of patch truncation caused by the location of anchor points and patch size. The small squares defined by the grid represent the locations within the arrays of feature vectors $\{\mathbf{c}_{i,j}\}$ and $\{\mathbf{q}_{i,j}\}$ in the candidate and query images, respectively. Images are also shown in the background for convenience. Left: A 9×9 patch is defined around an anchor location (small red square) in the candidate array. The patch is truncated on the top due to the anchor's proximity to the edge. Right: The associated anchor (closest match) in the query image array is slightly lower and closer to the left edge of the array. Consequently, the application of the candidate's patch to this location causes a further truncation in the query patch, this time on its left-hand side.

and the anchor's proximity to the edges of the array, the patch may be truncated. An exact copy of this (possibly truncated) patch is then applied to the anchor point in the query array, (k, l). The location of the latter will again determine whether the new patch must be further truncated or not. This is schematically illustrated in Fig. 3, where an initial 9×9 patch applied to the candidate array results in a final 6×8 patch in the query array.

Once the query image patch has been defined, we look at each of its containing vectors $\{\mathbf{q}_{k',l'}\}$ and find their closest matches, $\{\mathbf{c}_{i',j'}\}$, in the candidate image. For any given (k', l'), if the relative position of (i', j') with respect to anchor (i, j) is the same as the relative position of (k', l') with respect to its own anchor (k, l), then a location match between the images has been found and a score is increased for the current candidate. After searching for potential matches at all locations (k', l') within the patch, the procedure is repeated by looking at all remaining locations (i, j) in the candidate and their associated anchors in the query. An accumulated score for that candidate is created in this fashion, repeating the process for all candidates. In a final step, the candidate with the greatest score is selected as the matched place for the query image.

We note that this approach is different, more exhaustive and ultimately better than the one presented by us in [18], where we

looked at each location within $\{\mathbf{c}_{i,j}\}$ but instead of patches, we only considered each location's row and column when comparing it with $\{\mathbf{q}_{k,l}\}$.

3.4. Ground truth and frame tolerance

The datasets employed throughout this work consist of query and reference image sequences, both belonging to the same route but taken at different times and under changing conditions (see Section 4.2).

Given a query image of a place and its assigned ground truth in the reference sequence, we define *frame tolerance* or *tol* as the number of frames departing from the ground truth within which the output of the recognition system for that query is still considered a true positive. Rather than setting a fixed frame tolerance, we propose to find its optimal value for each considered dataset. We do this by directly looking at the recognition performance of our system as the tolerance is varied. Furthermore, we have adopted from [22,23,56] a maximum distance threshold for a place, where a query image can be deemed as a true positive only when its ground truth location is less than approximately 25 m from the guessed location. An analysis of performance vs. *tol* is carried out in Section 4.3.

3.5. Frame correlation (FC)

The system presented so far is based on the visual attributes of a single image. However, in a mobile robotics environment, it is almost certain that some time correlation will exist between frames, a fact that can be exploited to improve the overall recognition performance. We have implemented this idea in our system by introducing some prior knowledge into the spatial matching stage. In particular, when comparing candidate and query images, candidates that are consistent with recently recognized places are favored.

To describe how prior recognition knowledge is introduced and since the image sequences employed in this work correspond to forward movement, let us focus on frames rather than distances. We define n_p, a positive integer representing the total number of consecutive, past recognized places being considered. For $n_p = 1$, only the most recent recognition event is taken into account, denoted by $t = -1$ and associated with a particular frame number $f_{t=-1}$ in the reference sequence. In general, for any n_p, the recognized frames are $f_{t=-1}, \ldots f_{t=-n_p}$. We also define the relative score of individual past recognition events, $\hat{s_t}$, as the ratio between their score as given by *scores_hist* in Algorithm 2 and the maximum score among all n_p.

We propose the following expression to describe the contribution a_t of each past recognition to the candidate being matched with the query:

$$a_t = \begin{cases} \hat{s}_t \dfrac{(\Delta f_{max} - |f_t - f_c|)}{\Delta f_{max}}, & \text{if } |f_t - f_c| < \Delta f_{max} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

In Eq. (1), f_c is the frame number of the candidate being matched and Δf_{max} a threshold that restricts the range of frames between f_t and the candidate for which the latter is affected by past recognition events (default $\Delta f_{max} = 15$). When this is the case and subjected to $\hat{s_t}$, we see that candidates that are closer to the recognized place f_t contribute more to a_t, whereas the contribution tends to zero as the distance approaches Δf_{max}.

In order to add knowledge from the past, we define a candidate weighting factor, w_c, that will multiply the corresponding candidate bin in *scores_hist*:

$$w_c = 1 + C_p \max \{a_t\}_{t=-1}^{t=-n_p} \quad (2)$$

Eq. (2) tells us that from all past recognition events being considered, only the one that contributes the most to the current candidate is selected. The term C_p is a frame correlation parameter that can be obtained by experimentally analyzing recognition performance over a set of values and then looking at a consistent maximum across all datasets.

As it will be shown in Section 4.3.2.5, the approach described in this section can significantly improve recognition with respect to the single-frame approach.

4. Experiments

4.1. Current approaches in VPR

We have compared our recognition results with various existing VPR methodologies. In particular, we considered NetVLAD [22] and the recent methods developed in [42] (Region-BoW) and [43] (Region-VLAD) as well as some other approaches compared in those works. They are summarized below.

1. *NetVLAD* [22]. A CNN architecture that is trainable in an end-to-end manner directly for the place recognition task. It introduces a generalized VLAD [57] layer that can be trained via backpropagation.
2. *FAB-MAP* [20]. A probabilistic approach that learns a generative model of place appearance, using hand-crafted features (SURF) and a BoW model to encode images.
3. *SeqSLAM* [58]. It leverages the correlation between images and instead of calculating the most likely location given a single image, it uses short sequences of them to obtain the best match for a location.
4. *AlexNet* [39]. Uses a pre-trained AlexNet CNN architecture to represent a whole image and finds the best match through single-image nearest neighbor search based on the cosine distance. The best performance is obtained for the *conv3* layer.
5. SUMPOOL [59]. This method employs a simple sum-pooling aggregation scheme to create robust descriptors from deep CNN activations on the VGG16 architecture.
6. MAXPOOL [60]. Utilizes spatial max-pooling as a form of aggregation.
7. CROSSPOOL [61]. In this encoding method, image representations are created from the activations of two consecutive convolutional layers of a pre-trained CNN.
8. Region-BoW [42]. Identifies salient regions in images by looking at distinctive patterns created by the activations of different CNN layers. It uses one layer to discover relevant regions and another layer to match those regions among images.
9. Region-VLAD [43]. This method combines novel lightweight CNN-based regional features with VLAD [46] encoding.

4.2. Datasets

The five main datasets used in this paper are the same or very similar to those in [18,42,43]. As discussed earlier, they consist of pairs of image sequences along the same route. One sequence is used as reference to create the IFDB and SMDB databases and the other as query to test and evaluate the performance of the recognition system. The differences between the sequences span a wide range of condition changes and viewpoint variations, whereas the datasets themselves cover several environments (e.g. urban, university campus, and train journey), making them suitable for the evaluation of the overall performance of a VPR system. Examples of the same place for both reference and test traverses can be visualized in Fig. 4.

Fig. 4. Same-place examples of reference (left) and query (right) images for the main datasets considered in this work. Rows correspond, in descending order, to *Gardens Point*, *Berlin A100*, *Berlin Halenseestrasse*, *Berlin Kudamm* and *Synthesized Nordland* datasets.

In addition to these datasets, we have created three complementary larger datasets for the purposes of analyzing the influence of dataset size on recognition. All of them are summarized in Table 1 and described below.

4.2.1. Gardens point

This dataset consists of footage recorded with a hand-held, forward-facing mobile phone through the Gardens Point Campus at QUT university in Brisbane and includes outdoor, semi-indoor and indoor images. We employed as reference the traverse recorded from the left-hand side of the path at daytime whereas the one recorded from the right-hand side at night was used for testing. Changes in viewpoint and conditions are considered, respectively, strong and very strong for these sequences, which are also prone to important perceptual aliasing due to the man-made nature of the environment.

4.2.2. Berlin A100

It contains day-time sequences of frames along the same urban route, with each traverse recorded from a different car and by different users of Mapillary [62], a service for sharing crowdsourced geotagged photos. Viewpoint changes are strong and condition variations moderate for this dataset.

4.2.3. Berlin Halenseestrasse

Also downloaded from Mapillary, this dataset contains day-time traverses taken by a car user (reference) and by a bicycle user (query) in an urban area. Viewpoint changes are even stronger than for the previous dataset since query images were taken from the bicycle's lane. An extra difficulty comes from the fact that such a lane is sometimes surrounded by vegetation and it is not possible to find much resemblance with reference images (see example in Fig. 9). It is also worth noting that as many as 10 query images appear rotated from the horizontal, either by 90° clockwise/anti-clockwise or completely upside down, negatively affecting recognition.

4.2.4. Berlin Kudamm

This dataset, whose source is again Mapillary, is commonly regarded as the most challenging from all five considered. Set in a busy urban environment, reference frames were taken from the top deck of a bus and query ones from a bicycle, making for very strong viewpoint changes and moderate condition variations. In addition to these changes, there exist multiple dynamic objects such as moving and parked cars, pedestrians, and bicycles. There is also abundant vegetation that is present most of the time, occluding more static and differentiating features such as buildings. Ground truth correspondences in this dataset as well as in the two previous ones were made manually by carefully parsing the frames.

4.2.5. Synthesized Nordland

This is the largest of the main datasets, containing a total of 1415 frames per sequence. The footage was taken from a train traveling between two Norwegian cities, with the reference traverse recorded in summer and the query one in winter. Condition variation is very strong mainly due to snow-covered landscapes during the winter traverse. In order to introduce some moderate viewpoint variation, frames from the summer traverse were chosen so as to keep 75% viewpoint resemblance with the winter ones.

4.2.6. Large datasets

As seen in Table 1, the size of the first four datasets is considerably small, with *Synthesized Nordland* having moderate size. In order to investigate how dataset size can affect our recognition system, we created three extra datasets, each of them containing 8000 images per traverse. We considered urban and train journey environments.

For urban environments, we used routes through 8 different major cities around the world and downloaded from *Google Street View* a total of 8000 images. The two traverses were generated by randomly changing the panoramic view of each image as follows. Firstly, the horizontal rotation of the camera, θ_{init}, was calculated so that it faced the direction of the route (estimated by using the relative position of the previous place and the current one). From this initial angle, the reference and query images were generated by uniformly sampling two headings, θ_{left} and θ_{right}, in the range between 3 and 25 degrees. Headings for reference and query images were then obtained as $\theta_{init} - \theta_{left}$ and $\theta_{init} + \theta_{right}$, respectively. Similarly, a difference in the camera's pitch was introduced by applying two uniformly sampled angles between -5 and 20 degrees. Since the resulting images came from different parts of the same panorama, this dataset, dubbed *Cities-8000*, only takes into account differences in viewpoint but not condition changes. Fig. 5 shows a same-place example of reference and query images.

The two remaining datasets in Table 1 were created by downloading from [63] the first 8000 images of the Nordland dataset. We used summer traverse as reference and then winter and fall as query sequences. Sets were named *Nord-8000-SW* and *Nord-8000-SF*, respectively. The results of our experiments on these datasets are described in Section 4.4.3.

Table 1
Summary of the datasets used throughout this work.

Dataset	Environment	Traverse		Variation in	
		Reference	Query	Condition	Viewpoint
Gardens Point	Campus	200 (day-left)	200 (night-right)	Very strong	Strong
Berlin A100	Urban	85 (car)	81 (car)	Moderate	Strong
Berlin Halenseestr.	Urban	67 (car)	157 (bicycle)	Moderate	Strong
Berlin Kudamm	Urban	201 (bus)	221 (bicycle)	Moderate	Very strong
Synth. Nordland	Train journey	1415 (summer)	1415 (winter)	Very strong	Moderate
Cities-8000	Urban	8000 (left)	8000 (right)	No variation	Strong
Nord-8000-SW	Train journey	8000 (summer)	8000 (winter)	Very strong	Moderate
Nord-8000-SF	Train journey	8000 (summer)	8000 (fall)	Moderate	Moderate

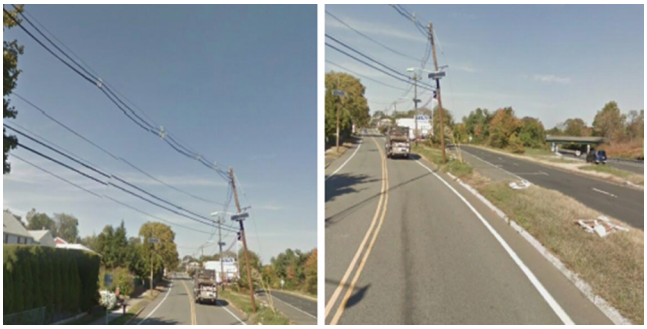

Fig. 5. Same-place example of reference (left) and query (right) images for the *Cities-8000* dataset.

4.3. Parameter optimization

In this section, we investigate the influence of several important parameters on the recognition performance of our system and find those that are optimal, on average, on the considered datasets. Results are presented below and in Table 2. By assuming a set of initial values, each parameter is sequentially optimized through stages I and II. Table 3 summarizes the followed optimization sequence in matrix form. Entries below the main diagonal correspond to parameters that have already been optimized, whereas entries above the diagonal are the initially used values. The last row in the table represents the full set after optimization.

4.3.1. Image filtering stage

Optimization in stage I proceeded by measuring the recall@N for different values of a given parameter. The recall@N is defined as the percentage of successfully retrieved places when considering a list of candidate images of size N. A place is deemed as successfully retrieved if at least one of the images in the list corresponds to a place that is close enough to the ground truth, given some distance or frame tolerance. An arbitrary, reasonably small number of candidates $N = 5$ (recall@5) was chosen in most cases to monitor search performance while avoiding reaching recalls of 100%, which otherwise would not provide much insight into the behavior of the parameter. Series of 5 runs were performed for each parameter and dataset. The results presented in Figs. 6 and 7 show the averages over the runs.

Parameters considered during this stage were (a) the image size S_1, (b) the spatial dimensions, $h \times w$, of convolutional cubes, (c) the number of cubes n_{c_1} and (d) the number of components, d_1, after PCA compression. Results and optimization details are discussed below.

4.3.1.1. Image size. Images were scaled to square dimensions in all datasets and sequences, bringing them to the smallest side (the image height in all cases). Results are shown in Fig. 6(a), where the average recall among all datasets (orange curve) suggest an optimal image size of 244×244 pixels.

4.3.1.2. CNN cube size. Recall as a function of the spatial dimensions of convolutional cubes is depicted in Fig. 6(b). Results indicate that taking cubes of size $9 \times 9 \times 512$ activations would provide, on average, the best recall for the image filtering stage.

4.3.1.3. Number of CNN cubes. Another parameter that can help to maximize recall is the number of convolutional cubes used during the search. For the already optimized parameters ($h \times w = 9 \times 9$ and $S_1 = 244 \times 244$ px), the maximum possible number of cubes is $n_{c_1} = 36$. Thus, we considered the range $5 \leq n_{c_1} \leq 36$. Experimental results presented in Fig. 6(c) reveal that the average recall continuously improves with n_{c_1} and therefore $n_{c_1} = 36$ was chosen as the default value. Note that even taking $n_{c_1} = 20$ would still provide reasonable recalls, with the advantage of reducing the storage requirements for stage I by almost 50%.

4.3.1.4. PCA components. An analysis on the effect of PCA components on recall is presented in Fig. 6(d). On this occasion, we chose $N = 50$ candidates because this is roughly the threshold we set in terms of achieving near real-time place recognition. One can see in the figure that on average, the recall peaks at $d_1 = 125$ and therefore this is the default value adopted throughout this work.

4.3.2. Spatial matching stage

The parameters considered for optimization in stage II are (a) the image size S_2, (b) the number of PCA components d_2, (c) the frame tolerance *tol*, (d) the anchor patch size S_{patch}, (e) the frame correlation factor C_p in Eq. (2), and (f) the number of previously recognized places, n_p, taken into account when exploiting frame correlation.

4.3.2.1. Image size. We proceeded similarly to the first stage and scaled images to square dimensions. The size of convolutional cubes was not considered for optimization in stage II and kept constant at $3 \times 3 \times 512$ activations. Thus, increasingly larger images produced more cubes, which were associated with smaller areas of the original image and resulted in more spatial resolution during the spatial matching process.

An analysis of precision as a function of image size is presented in Fig. 7(a). In general, an increase in precision is observed as the size of the images grows towards their original size, which was 360×360 px for the *Synthesized Nordland* dataset and a slightly larger 480×480 px for the rest. This suggests that the spatial matching stage requires as much image resolution as possible to perform adequately. Nevertheless, scaling up images above their original sizes does not provide any additional information and

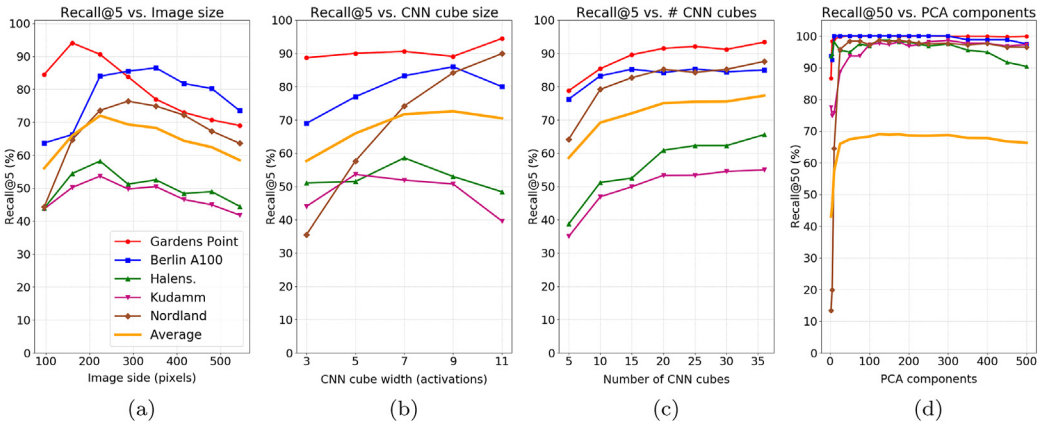

Fig. 6. Parameter optimization for stage I. (a) Recall@5 as a function of image size. (b) Recall@5 as a function of the spatial dimensions of the CNN cubes. (c) Recall@5 as a function of the employed number of CNN cubes. (d) Recall@50 as a function of the number of PCA components (average shifted down by 30% for ease of visualization).

this fact is reflected in the figure by no further improvement or even a slight reduction in precision above those sizes (note how this happens at smaller sizes for *Synthesized Nordland*). From the above discussion and as long as complexity allows for it, using image sizes that are as close as possible to the original size of reference images is recommended. Here, taking the average precision in all datasets, images of size 416×416 px should provide the best results as a whole.

Recognition times per image are also depicted in Fig. 7 (a) for two of the datasets at each size and referred to a standard CPU configuration (see Section 4.4.4). The selected size of 416×416 pixels allows for recognition times of around 1.5 s. With slightly more powerful hardware, real-time recognition can be easily achieved.

4.3.2.2. PCA components. Using 416×416 images, we investigated precision as a function of the number of PCA components and found out that no significant gain was achieved above $d_2 = 100$ dimensions, which is the default value set for the rest of this paper.

4.3.2.3. Frame tolerance. Results for the analysis discussed in Section 3.4 are presented in Fig. 7 (b), which shows recognition precision when considering several frame tolerances in the assignation of true positives. A zero tolerance signifies that only those query images whose guessed location is exactly the ground truth are considered true positives. As can be seen, relaxing the tolerance from $tol = 0$ to larger values quickly translates into higher recognition precision. This is due to the existence of several query images for which the guessed place is not exactly the ground truth but nonetheless in its close proximity. We also see that above certain tolerances the performance gain rate becomes both small and constant. This is consistent with the random inclusion of images as true positives due exclusively to larger tolerances but otherwise uncorrelated with the correct recognition of places. The fact that the slope of the gain is steeper in the smallest datasets also supports this idea.

We considered those frame tolerances where the rate becomes approximately constant as an intrinsic indicator of the spatial limits of a given place. In accordance with this criterion, we see that a tolerance of $tol = \pm 3$ should be chosen for *Gardens Point* whereas for *Halenssestrasse*, $tol = \pm 1$ appears to be more appropriate. For the rest of the datasets, a tolerance of $tol = \pm 2$ was found suitable based on the figure. We note that in all cases, these tolerances meet the 25 m threshold discussed in Section 3.4.

4.3.2.4. Anchor patch size. The size of the patch around the anchor points during stage II, S_{patch}, is an important parameter to be optimized. Small patches will speed up recognition but might not capture enough geometric information. On the other hand, patches that reach over long distances from the anchors may be beneficial when comparing image sequences with small viewpoint differences between them. When this is not the case, however, large patches may not be very useful because feature locations with respect to a particular anchor can quickly decorrelate between two images of the same place.

We looked at square patches that, when referred to the image width, covered from 12% to 200% of it. By using values up to 200% we intentionally included situations where the entire image is covered by the patch, even when the anchor points were located near the edges. Results in Fig. 7 (c) strongly suggest that patches covering 75% of the image width may be the best choice on average. When referred to the feature vector array, this corresponds to a 19×19 patch. Note how precision in *Synthesized Nordland* keeps improving even for increasingly large patches. Since changes in viewpoint are rather small in this dataset, it seems reasonable that we can extend the reach of the patches and still benefit from it.

4.3.2.5. Frame correlation. The frame correlation factor, C_p, appearing in Eq. (2) acts as a scaling factor of the most contributing past recognition event. Too small C_p values will cause little knowledge from past recognitions being added. On the contrary, a value that is too large may lead to problems such as cascade errors of multiple frames, where the system might 'get stuck' in a sequence of false positives. For these reasons, a value of $C_p = 0.4$ was determined experimentally by analyzing recognition performance over a set of values and then looking at a consistent maximum across all datasets. Only the last recognition event was considered in such an analysis.

Using the optimized parameter, precision was monitored as a function of the number of recognition events, up to $n_p = 10$. Results are depicted in Fig. 7 (d). Starting from $n_p = 0$ (no past information added), a clear improvement in performance is observed when the latest recognition output is considered. This can be as large as 15% for the *Kudamm* dataset. Adding a second event from the past still improves recognition in most datasets, although the gain is not so significant. The average precision across all datasets suggests that the optimal performance is reached at $n_p = 2$, with an average improvement of around 10% with respect to the single-frame approach. We, therefore, use this optimized value as default.

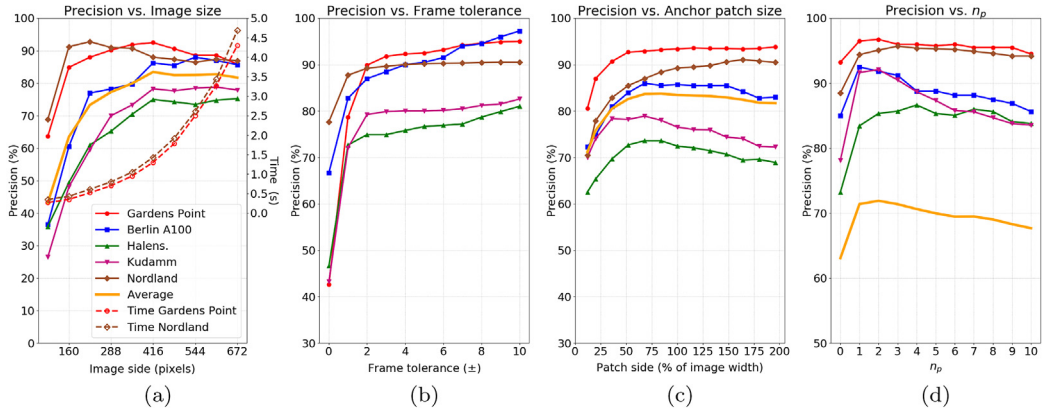

Fig. 7. Parameter optimization for the spatial matching stage. (a) Precision of the SSM_{NRT} system as a function of image size. Recognition times per image are also depicted for the *Gardens Point* and *Synthesized Nordland* datasets. (b) Precision of SSM_{NRT} as a function of the ground truth frame tolerance. (c) Precision as a function of the anchor patch side length (expressed as a percentage of image width). (d) Precision of the SSM_{NRT} system as a function of the number of previously recognized places considered, n_p. Average has been shifted down by 20% for better visualization.

Table 2
Optimized parameters used throughout this work and adjusted for near real-time recognition (e.g $N = 50$).

Parameter	Optimized	Stage	Description
S_1	224×224	I	Image size (in pixels)
$h \times w$	$9 \times 9 \times 512$	I	CNN cube spatial dimensions (in activations)
n_{c_1}	36	I	Number of CNN cubes
d_1	125	I	Number of PCA components
N	50	I	Number of candidates (for near real-time recognition)
S_2	416×416	II	Image size in pixels
d_2	100	II	Number of PCA components
tol	± 1	II	Frame tolerance for Halenseestrasse dataset
tol	± 2	I, II	Frame tolerance for A100, Kudamm, Nordland and large datasets Also used to evaluate recall in stage I
tol	± 3	II	Frame tolerance for Gardens Point dataset
S_{patch}	19×19	II	Default patch size around anchor points in feature vector array
C_p	0.4	II	Frame correlation factor
n_p	2	II	Number of past recognition outputs considered in frame correlation

Table 3
Parameter optimization sequence (top to bottom) followed in this work. Entries below the main diagonal correspond to optimized parameters. Entries above the diagonal refer to initial parameters. The last row shows the full set of optimized parameters.

Optimization sequence	Stage	Optimized parameters\Initial parameters									
		S_1	$h \times w$	n_{c_1}	d_1	S_2	d_2	*tol*	S_{patch}	C_p	n_p
S_1	I	–	$H/2 \times W/2$	16	100			2			
$h \times w$	I	224×224	–	16	100			2			
n_{c_1}	I	224×224	9×9	–	100			2			
d_1	I	224×224	9×9	36	–			2			
S_2	II	224×224	9×9	36	125	–	100	2	25×25		
d_2	II	224×224	9×9	36	125	416×416	–	2	25×25		
tol	II	224×224	9×9	36	125	416×416	100	–	25×25		
S_{patch}	II	224×224	9×9	36	125	416×416	100	1, 2, 3	–		
C_p	II	224×224	9×9	36	125	416×416	100	1, 2, 3	19×19	–	
n_p	II	224×224	9×9	36	125	416×416	100	1, 2, 3	19×19	0.4	–
All		$\mathbf{224 \times 224}$	$\mathbf{9 \times 9}$	**36**	**125**	$\mathbf{416 \times 416}$	**100**	**1, 2, 3**	$\mathbf{19 \times 19}$	**0.4**	**2**

4.4. Results

This section presents and discusses our recognition results, comparing them with relevant published approaches [42,43] that employ the same datasets as we do in this paper. We also perform a thorough comparison of our method with NetVLAD [22], regarded as the state of the art in VPR [47], and test its capabilities as an alternative for the image filtering stage in our system. In addition, using the complementary large datasets described in

Section 4.2.6, we provide an analysis and discussion on the effect of dataset size on recognition.

4.4.1. Main recognition results

Recognition performance was assessed by evaluating precision–recall (PR) curves. The recall was varied by using several thresholds for the accumulated score during the spatial matching stage (the bin height in *scores_hist*, Algorithm 2). In order to compare with other approaches, we utilized the values reported for the Area Under the Curve (AUC) of the PR curves. An exception was NetVLAD, for which we calculated the PR curves by using the freely available Python/Tensorflow implementation found in [64]. The AUC was computed with a simple trapezoidal rule, given by Eq. (3):

$$\text{AUC} = \sum_i (r_i - r_{i-1}) \times \frac{p(r_i) + p(r_{i-1})}{2}, \tag{3}$$

where $p(r_i)$ is the precision at recall r_i and i is an index over the number of calculated points in the PR-curve.

We provide recognition results for five different variants of our pipeline:

- SSM_0: Image filtering stage as an independent recognition system, without any geometric verification taking place and selecting the top candidate as the recognition output. It basically provides the performance of layer *conv 5-2* in the VGG16 architecture. Thus, it is conceptually close to the *AlexNet* method considered in [43,47], with the difference that we used several image regions instead of the full image.
- SSM [18]. Our own approach previous to the improvements and optimizations addressed in this paper.
- SSM_{NRT}: This is the two-stage system as explained in Sections 3.1, 3.2 and 3.3. The set of parameters are those in Table 2, which allow for recognition in near real-time (NRT).
- SSM_{NRT}+FC: two-stage system tuned for near real-time recognition and boosted by introducing knowledge about previously recognized places (Section 3.5).
- SSM_{max}+FC: No recognition time restriction is imposed and therefore represents the maximum performance that can be achieved by the system in its current implementation. In this version, all reference images are fed into the spatial matching stage (i.e. no image filtering is performed), with the exception of *Synthesized Nordland*, where a maximum of 200 candidates was considered.

AUC values for the five main datasets are presented in Fig. 8(a) as bar plots. We included all the methods discussed in Section 4.1 and the five versions of our system. To make a more faithful comparison of our approach with NetVLAD, we investigated the relationship between the recognition performance of the latter and image size. Although not shown, results suggested an optimal image size of 480 × 480 pixels and therefore, all NetVLAD results from now on will be referred to this particular size.

As can be seen in Fig. 8(a) and except for our own results, NetVLAD outperformed all previously published works in all datasets but *Synthesized Nordland*, where it was significantly surpassed by AlexNet and Region-VLAD methods only. Its superiority is especially apparent in the *Kudamm* dataset, a result in agreement with [47], where the authors have recently shown that from a total of 10 VPR approaches, the NetVLAD architecture performed significantly better on the Kudamm dataset. Based on [47] and our calculations shown in the figure, NetVLAD has been selected as the main baseline methodology against which to compare our approach throughout this paper.

Fig. 8(a) also shows that any of the two-stage optimized versions of our system (last three bars on each plot) is superior to all other methods considered. The system reported by us in [18] behaves similarly or slightly worse than NetVLAD on the urban datasets, but it is clearly better on the rest. A more detailed comparison of our approach with the baseline is provided by the PR curves presented in Fig. 8(b). As can be seen, precision at 100% for the SSM_{NRT} version largely surpasses it in all datasets, making clear the advantage of using robust spatial information in the visual place recognition task. In the following, we discuss results for each dataset separately.

4.4.1.1. Gardens point. A first look at Fig. 8(b) for this dataset reveals that precision at 100% for NetVLAD is lower than that of SSM_0 at almost all recalls. NetVLAD model was trained in an end-to-end manner on the *Pittsburgh 30K* dataset [30], which contains daytime outdoor images only. Since the query sequence in the *Gardens Point* dataset consists of semi-indoors images taken at night, we presume NetVLAD features are not particularly advantageous on this dataset, hence the lower performance when compared to even the basic SSM_0. This behavior, which has also been noted elsewhere [65], highlights the importance of choosing training datasets that are representative of the query images.

Even more interesting is the dramatic gain in recognition performance when the spatial matching stage is introduced. Without using any knowledge from previously recognized places, the precision of SSM_{NRT} at 100% recall escalates from 70% to 92%. Even at this high recognition rate, introducing frame correlation into the spatial matching stage resulted in an extra 5%. With a precision of 97%, this brings the system to nearly perfect performance.

As mentioned earlier, the query sequence for this dataset consists of night footage and contains images of rather bad quality that exhibit a large amount of noise. Some of the images are totally cluttered with this noise, as can be seen in the example shown in Fig. 9. Most recognition mistakes on this dataset are associated with this kind of query images. Otherwise, our system exhibits superb performance under very strong condition variations due to day–night cycles.

4.4.1.2. Berlin A100, Halenseestrasse and Kudamm. We discuss these three datasets together since they represent fairly similar urban environments in the city of Berlin. AUC values show that NetVLAD's performance is very similar to SSM [18], clearly surpassing SSM_0 and the rest of published methods in all three cases. This is most likely due to similarities in appearance between the current datasets and the also urban *Pittsburgh* dataset. For instance, for *Kudamm*, AUC is almost 50% higher than the next best method (SSM_0), corroborating the power of this architecture when the training set is representative of the testing set.

Nevertheless, Fig. 8(b) shows that adding the spatial matching stage to our system results in an outstanding recognition improvement, with precision at 100% recall surpassing NetVLAD by almost 60% in the case of Kudamm. For *Berlin A100* and *Halenseestrasse*, the differences are not so large, with precision going above the baseline by 25% and 21%, respectively. However, with the introduction of frame correlation, performance soars to levels around 90% precision in all cases, with *Berlin A100* and *Kudamm* reaching, respectively, an impressive 94% and 92%. The *Halenseestrasse* datasets only achieved a still acceptable 86%, but as discussed in Section 4.2, there are some issues with this dataset that lowers the precision. An example can be seen at the bottom of Fig. 9, which shows a query where recognition is inevitably unsuccessful due to total occlusion by vegetation.

Including all reference images as candidates (SSM_{max}+FC) did not make any difference in *Berlin A100* and *Halenseestrasse*, mostly because of the small size of the reference sequence, which for the default $N = 50$ candidates already included the majority of images. However, for *Kudamm*, we were able to further increase precision to 98%. With an AUC = 0.998, we reached almost perfect

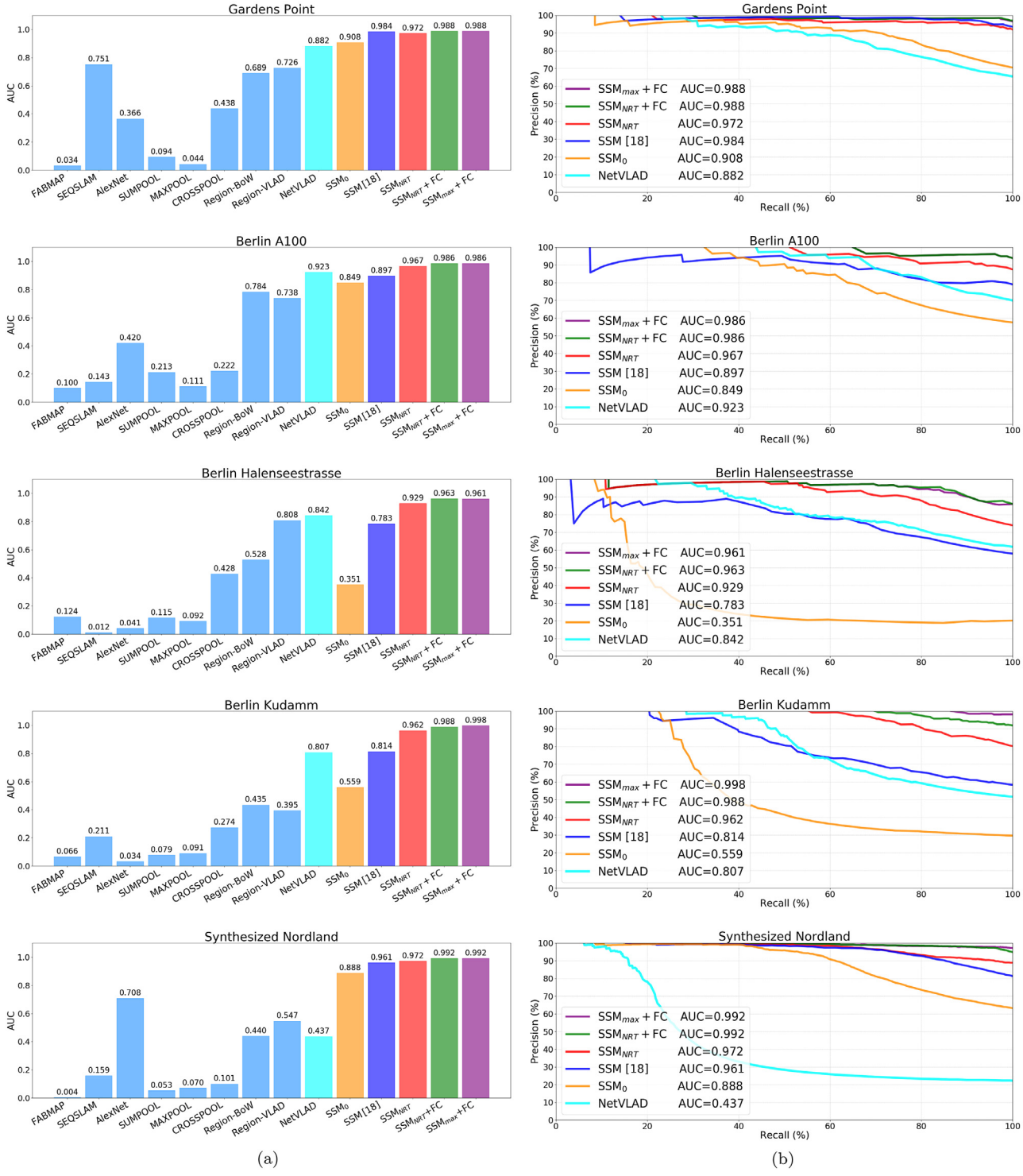

Fig. 8. Area Under the Curve (AUC) of the precision–recall curves for the methods discussed in Section 4.1 and the five variants of our system.

performance in a dataset for which all other VPR approaches consistently perform rather poorly. Recognition results on this dataset represent a milestone of our work so far, showing unprecedented performance of our system in the task of visual place recognition under very strong viewpoint variations as well as changes due to dynamic objects and vegetation.

4.4.1.3. Synthesized Nordland. On this dataset, the performance of all third party methods considered is remarkably low, with the exception of AlexNet, whose AUC was still worse than SSM_0. Note that in previous datasets, the differences between these two methods were rather larger, with SSM_0 clearly outperforming AlexNet. Since the latter uses entire activation feature maps to represent images, our SSM_0 approach based on multiple smaller regions seems to be more robust to changes in viewpoint. Thus, the discrepancies are more noticeable on the first 4 datasets and less pronounced in *Nordland*, where changes in viewpoint are only moderate.

Other than AlexNet, the generally low performance is almost certainly due to the drastic change in appearance that snowfalls cause on landscapes. In particular, this is the dataset for which

Fig. 9. Top: *Gardens Point* dataset. Example of very poor quality query image taken at night (right) and its corresponding ground truth in the reference sequence (left). Bottom: *Berlin Halenseestrasse* dataset. Problematic query image (right) where practically no resemblance can be found with the reference image (left) due to occlusions from vegetation.

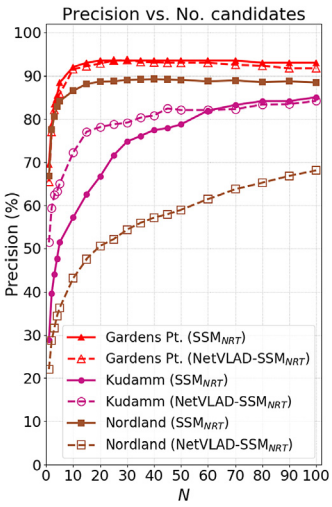

Fig. 10. Comparison of recognition performance of the two-stage pipeline as a function of the number of candidates when considering two different approaches in stage I: (1) the original implementation based on layer *conv 5-2* (SSM$_{NRT}$) and (2) NetVLAD (NetVLAD-SSM$_{NRT}$).

NetVLAD performed the worst, with AUC = 0.437 and a precision at 100% recall of 22%. In line with the observations made for *Gardens Point*, the *Synthesized Nordland* dataset is strictly non-urban, with the query sequence characterized by snow-covered scenes. Again, we believe that the features learned by NetVLAD on the *Pittsburgh* dataset may not appropriately cover this type of environment, hence the lower performance. The more general features extracted from the pre-trained VGG16 architecture seem to work much better, as seen by the relatively good performance of the SSM$_0$ system.

Once the spatial matching stage is incorporated for this dataset, precision at 100% recall rises to from 63% to 89%. Furthermore, considering frame correlation and increasing the candidate's list to $N = 200$ brings about 94.8% precision and nearly perfect recognition performance (97.1%), respectively. Although viewpoint variations are not so strong for this dataset, the unmatched precision reveals the robustness of our approach under very strong variation in conditions due to different seasons of the year.

To summarize the results in this section, we have shown that using our pipeline, a dramatic improvement in place recognition precision can be achieved with respect to state-of-the-art approaches, obtaining nearly perfect performance on most of the datasets considered. The power of our approach lies mainly in the addition of the spatial matching stage, but also in exploiting time correlation between frames in a sequence and an exhaustive optimization of hyper-parameters.

We want to emphasize the fact that our spatial matching procedure can be applied to any existing recognition system whose output is a list of candidates. A clear example can be found in the following Sections 4.4.2 and 4.4.3, where the NetVLAD methodology is extended by our geometric verification scheme.

4.4.2. Image filtering with NetVLAD

We have analyzed the database image filtering capabilities of NetVLAD as compared to our original filtering scheme based on layer *conv 5-2* of the VGG16 architecture. Fig. 10 shows recognition precision of the two-stage system when both approaches were used in stage I and for several choices in the number of candidates N. We only considered *Gardens Point*, *Kudamm* and *Synthesized Nordland*, as *Berlin A100* and *Halenseestrasse* are expected to behave similarly to *Kudamm*. For *Gardens Point*, we see in the figure that both SSM$_{NRT}$ and NetVLAD-SSM$_{NRT}$ (the system using NetVLAD in stage I) behave similarly. For the default number of candidates $N = 50$, precision is in both cases around 93%. Recognition times per image were also comparable so there is no real advantage of using one system over the other.

On the *Kudamm* dataset, however, performance is significantly better for NetVLAD-SSM$_{NRT}$ at low number of candidates. For instance, a precision of 80% is reached at around $N = 30$, whereas using our original approach required $N = 55$. This suggests that NetVLAD could be employed in this case to reduce the number of candidates while maintaining relatively good performance, with the advantage of cutting recognition times down.

In the case of the *Nordland* dataset, our approach outperformed NetVLAD-SSM$_{NRT}$ at all N and reached top performance at around $N = 20$, a rather positive result considering that the dataset contains a moderately large number of images. NetVLAD's low performance for image filtering on this dataset evidences once again the unsuitability of the employed model on scenarios affected by heavy snowfalls.

4.4.3. Dataset size

To complete our results, we have investigated recognition as the size of datasets grows. Thus, we calculated recognition performance as a function of the number of images on the large sets presented in Section 4.2.6. We considered NetVLAD as a stand-alone recognition system, NetVLAD-SSM$_{NRT}$, and SSM$_{NRT}$. Frame tolerance was set to ± 2 frames in all cases and the number of images varied from 50 to 8000. Results are depicted in Fig. 11 and separately discussed below for each dataset.

4.4.3.1. Cities-8000. A first conclusion drawn from the figure is that on these larger datasets, recognition tends to deteriorate as the number of images grows. On the *Cities-8000* dataset, SSM$_{NRT}$ sees a decrease in precision from 100% to 44%. The extensive drop is due to the low performance of the image filtering stage. A proof for this is that the NetVLAD-SSM$_{NRT}$ system keeps precision above 95% at all dataset sizes. This is certainly an indication that NetVLAD does on this urban image collection a very good job at selecting the right candidates, a fact that was extensively demonstrated in the original paper [22]. Even the stand-alone NetVLAD system surpasses our approach significantly. The results in Fig. 11 give first evidence that our image filtering approach may not be the most suitable on large datasets containing strong

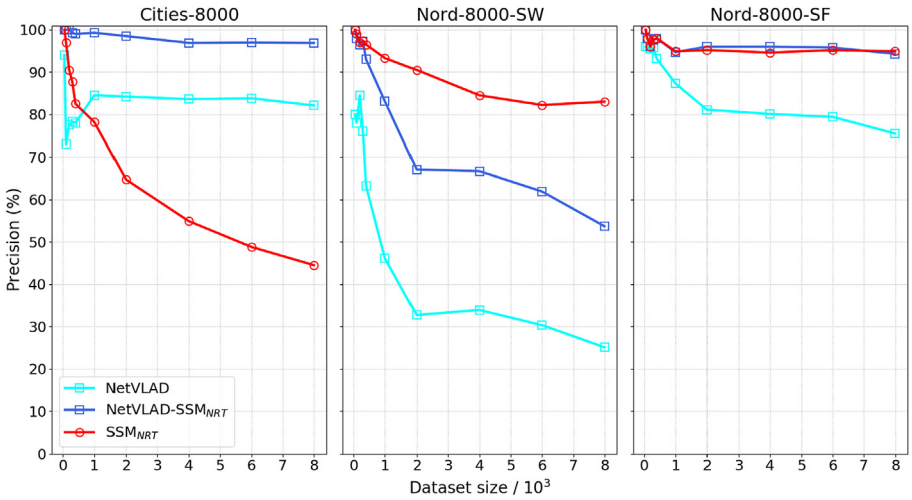

Fig. 11. Recognition performance as a function of dataset size for NetVLAD, SSM_{NRT} and NetVLAD-SSM_{NRT} (SSM_{NRT} using NetVLAD in the image filtering stage).

viewpoint variations. On the other hand, it demonstrates the power of our spatial matching scheme when implemented on top of an efficient image filtering methodology, as is the case for NetVLAD when trained on a suitable corpus of images.

4.4.3.2. Nord-8000-SW. On this train journey dataset, characterized by very strong condition changes due to snow-covered landscapes, our original system SSM_{NRT} is superior to both NetVLAD and NetVLAD-SSM_{NRT}. The inability of NetVLAD's model to robustly encode features in this kind of environment leads to the observed poor recognition performance, both as a stand-alone system and as a provider of candidates. Still, note the large improvement in precision (around 30% on average) when geometrical verification is added. Our original system stayed above 80% precision at all dataset sizes, showing increased robustness to extreme condition changes when viewpoint differences are not too large.

4.4.3.3. Nord-8000-SF. Just as the reference sequence, the query sequence in this dataset, recorded during fall, has a total lack of snow. Since changes in viewpoint are not so large, images of the same place in both sequences tend to look very similar to each other. This explains the good performance of both SSM_{NRT} and NetVLAD-SSM_{NRT}, which stays around 95% precision for all dataset sizes. It also shows that the NetVLAD model learned in an urban environment can also be applied to non-urban ones as long as more dramatic changes such as those caused by snow are not present.

To summarize, results in Fig. 11 suggest that for small to medium-sized datasets, our original system, which is entirely based on pre-trained, off-the-shelf CNN features, can provide excellent recognition results. In situations where the number of images is considerably larger than the list of candidates, we obtained somewhat mixed results. Heavy changes in viewpoint may require more powerful image filtering approaches, such as NetVLAD trained on a suitable dataset. Strong appearance variations, on the other hand, seem to be handled reasonably well with pre-trained features, as it is also the case for slight condition and viewpoint changes.

4.4.4. Runtime and memory considerations

Average recognition times per image were presented in Fig. 7(a) for the near real-time system (SSM_{NRT}) on an Intel(R) Core(TM) i7-7700 CPU @ 3.60 GHz and Python 3.6. For the image sizes shown in Table 2, it takes around 1.5 s to recognize a place. Currently, we are using 50 candidate images. A reduction to 25 candidates brings down recognition time to 950 ms and 600 ms for a candidate list of 10 images. By running the system on an Nvidia GeForce RTX 2080 Ti without any code optimization, recognition times per image were reduced to 810/440/250 ms. Regarding storage requirements, each image in stage I and II takes approximately 18 kB and 250 kB of disk space, respectively.

5. Conclusions and future work

In this paper, we have shown that image database filtering combined with robust and exhaustive spatial matching of images can be a very successful two-stage approach to the problem of visual place recognition. By adding a geometric verification stage to an initial image retrieval stage, both based on pre-trained CNN features, we were able to improve by a very large margin the state of the art, reaching nearly perfect recognition on most of the datasets considered.

In order to obtain such a level of performance, we took several paths. Firstly, based on the methodology originally proposed by us in [18], we performed an in-depth analysis and optimization of the system's hyperparameters. Secondly, we revised and improved the most important stage of our approach, where the geometric verification of query and candidate images takes place. Thirdly, we exploited the time correlation existing between frames in a sequence.

During the retrieval or image filtering stage, our results indicate that for small to medium datasets, our initial approach based on pre-trained CNN features performs fairly well, with the advantage of not requiring any additional training. However, as the datasets grow in size, and especially under conditions of large viewpoint changes, it may be necessary to consider other approaches such as end-to-end trainable architectures. The performance of the latter may depend nonetheless on whether a representative set of training samples is used. When this is not the case, recognition precision can be significantly lower than in systems utilizing more general, pre-trained features.

In the future, we are planning to explore even larger datasets and expand their number in order to cover a wider set of changing environments. We will also look at alternative, more recent CNN architectures such as ResNet, Inception, or DenseNet, which have shown better performance and lower complexity in the task of image recognition compared to VGG16. It is also our intention to integrate our recognition system in real robotic applications, with a view on FPGA real-time implementations. As an example, we are currently working [66] on a very robust teach-and-repeat navigation system that uses our methodology to localize a robot and to guide it by visual servoing during the repeat phase.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 688117, by the Technology Agency of the Czech Republic under the project no. TE01020197 "Centre for Applied Cybernetics", and by the European Regional Development Fund under the project Robotics for Industry 4.0 (reg. no. CZ.02.1.01/0.0/0.0/15 003/0000470)

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.robot.2020.103625.

References

[1] L. Kunze, N. Hawes, T. Duckett, M. Hanheide, T. Krajník, Artificial intelligence for long-term robot autonomy: a survey, IEEE Robot. Autom. Lett. 3 (4) (2018) 4023–4030.
[2] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, J.J. Leonard, Past, present, and future of simultaneous localization and mapping: toward the robust-perception age, IEEE Trans. Robot. 32 (6) (2016) 1309–1332.
[3] C. Valgren, A.J. Lilienthal, SIFT, SURF & seasons: Appearance-based long-term localization in outdoor environments, Robot. Auton. Syst. 58 (2) (2010) 149–156.
[4] W. Churchill, P. Newman, Practice makes perfect? managing and leveraging visual experiences for lifelong navigation, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 4525–4532.
[5] A.J. Davison, Real-time simultaneous localisation and mapping with a single camera, in: Iccv, Vol. 3, 2003, pp. 1403–1410.
[6] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, et al., FastSLAM: A factored solution to the simultaneous localization and mapping problem, Aaai/iaai 593598, 2002.
[7] P. Newman, K. Ho, SLAM-loop closing with visually salient features, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, IEEE, 2005, pp. 635–642.
[8] M. Labbe, F. Michaud, Online global loop closure detection for large-scale multi-session graph-based SLAM, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 2661–2666.
[9] K.L. Ho, P. Newman, Detecting loop closure with scene sequences, Int. J. Comput. Vis. 74 (3) (2007) 261–286.
[10] A. Angeli, D. Filliat, S. Doncieux, J.-A. Meyer, A fast and incremental method for loop-closure detection using bags of visual words, IEEE Trans. Robot. (2008) 1027–1037.
[11] S. Lowry, N. Sünderhauf, P. Newman, J.J. Leonard, D. Cox, P. Corke, M.J. Milford, Visual place recognition: A survey, IEEE Trans. Robot. 32 (1) (2016) 1–19.
[12] D.L. Yamins, H. Hong, C.F. Cadieu, E.A. Solomon, D. Seibert, J.J. DiCarlo, Performance-optimized hierarchical models predict neural responses in higher visual cortex, Proc. Natl. Acad. Sci. 111 (23) (2014) 8619–8624.
[13] S.-M. Khaligh-Razavi, N. Kriegeskorte, Deep supervised, but not unsupervised, models may explain IT cortical representation, PLoS Comput. Biol. 10 (11) (2014) e1003915.
[14] R.M. Cichy, A. Khosla, D. Pantazis, A. Oliva, Dynamics of scene representations in the human brain revealed by magnetoencephalography and deep neural networks, Neuroimage 153 (2017) 346–358.
[15] R.A. Epstein, E. Zita Patai, J.B. Julian, H.J. Spiers, The cognitive map in humans: spatial navigation and beyond, Nature Neurosci. 20 (11) (2017) 1504–1513.
[16] R.A. Epstein, L.K. Vass, Neural systems for landmark-based wayfinding in humans, Philos. Trans. R. Soc. B 369 (1635) (2014).
[17] K. Lynch, The Image of the City, Vol. 11, MIT Press, 1960.
[18] L.G. Camara, L. Přeučil, Spatio-semantic convnet-based visual place recognition, in: 2019 European Conference on Mobile Robots, IEEE, 2019, pp. 1–8.
[19] D. Gálvez-López, J.D. Tardos, Bags of binary words for fast place recognition in image sequences, IEEE Trans. Robot. 28 (5) (2012) 1188–1197.
[20] M. Cummins, P. Newman, FAB-MAP: Probabilistic localization and mapping in the space of appearance, Int. J. Robot. Res. 27 (6) (2008) 647–665.
[21] E. Eade, T. Drummond, Unified loop closing and recovery for real time monocular SLAM, in: BMVC, Vol. 13, Citeseer, 2008, p. 136.
[22] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, J. Sivic, NetVLAD: CNN architecture for weakly supervised place recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5297–5307.
[23] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, T. Pajdla, 24/7 place recognition by view synthesis, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1808–1817.
[24] S. Lowry, H. Andreasson, Lightweight, viewpoint-invariant visual place recognition in changing environments, IEEE Robot. Autom. Lett. 3 (2) (2018) 957–964.
[25] F. Perronnin, Y. Liu, J. Sánchez, H. Poirier, Large-scale image retrieval with compressed fisher vectors, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 3384–3391.
[26] M. Douze, A. Ramisa, C. Schmid, Combining attributes and fisher vectors for efficient image retrieval, in: CVPR 2011, IEEE, 2011, pp. 745–752.
[27] N. Kejriwal, S. Kumar, T. Shibata, High performance loop closure detection using bag of word pairs, Robot. Auton. Syst. 77 (2016) 55–65.
[28] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2, CVPR'06, IEEE, 2006, pp. 2169–2178.
[29] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Object retrieval with large vocabularies and fast spatial matching, in: 2007 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2007, pp. 1–8.
[30] A. Torii, J. Sivic, T. Pajdla, M. Okutomi, Visual place recognition with repetitive structures, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 883–890.
[31] T. Sattler, M. Havlena, K. Schindler, M. Pollefeys, Large-scale location recognition and the geometric burstiness problem, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1582–1590.
[32] S. Srinivas, R.K. Sarvadevabhatla, K.R. Mopuri, N. Prabhu, S.S. Kruthiventi, R.V. Babu, A taxonomy of deep convolutional neural nets for computer vision, Front. Robot. AI 2 (2016) 36.
[33] E. Karami, S. Prasad, M. Shehata, Image matching using SIFT, SURF, BRIEF and ORB: performance comparison for distorted images, 2017, arXiv preprint arXiv:1710.02726.
[34] A. Sharif Razavian, H. Azizpour, J. Sullivan, S. Carlsson, CNN features off-the-shelf: an astounding baseline for recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2014, pp. 806–813.
[35] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, arXiv preprint arXiv:1409.1556.
[36] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587.
[37] V. Badrinarayanan, A. Kendall, R. Cipolla, Segnet: A deep convolutional encoder-decoder architecture for image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 39 (12) (2017) 2481–2495.
[38] Z. Chen, O. Lam, A. Jacobson, M. Milford, Convolutional neural network-based place recognition, 2014, arXiv preprint arXiv:1411.1509.
[39] N. Sünderhauf, F. Dayoub, S. Shirazi, B. Upcroft, M. Milford, On the performance of convnet features for place recognition, 2015, arXiv preprint arXiv:1501.04158.
[40] Y. Hou, H. Zhang, S. Zhou, Convolutional neural network-based image representation for visual loop closure detection, in: 2015 IEEE International Conference on Information and Automation, IEEE, 2015, pp. 2238–2245.
[41] R. Arroyo, P.F. Alcantarilla, L.M. Bergasa, E. Romera, Fusion and binarization of CNN features for robust topological localization across seasons, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2016, pp. 4656–4663.
[42] Z. Chen, F. Maffra, I. Sa, M. Chli, Only look once, mining distinctive landmarks from convnet for visual place recognition, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2017, pp. 9–16.
[43] A. Khaliq, S. Ehsan, M. Milford, K. McDonald-Maier, A holistic visual place recognition approach using lightweight CNNs for severe viewpoint and appearance changes, 2018, arXiv preprint arXiv:1811.03032.
[44] P. Panphattarasap, A. Calway, Visual place recognition using landmark distribution descriptors, in: Asian Conference on Computer Vision, Springer, 2016, pp. 487–502.
[45] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, M. Milford, Deep learning features at scale for visual place recognition, in: Robotics and Automation (ICRA), 2017 IEEE International Conference on, IEEE, 2017, pp. 3223–3230.

[46] H. Jegou, F. Perronnin, M. Douze, J. Sánchez, P. Perez, C. Schmid, Aggregating local image descriptors into compact codes, IEEE Trans. Pattern Anal. Mach. Intell. 34 (9) (2012) 1704–1716.
[47] M. Zaffar, A. Khaliq, S. Ehsan, M. Milford, K. McDonald-Maier, Levelling the playing field: A comprehensive comparison of visual place recognition approaches under changing conditions, 2019, arXiv preprint arXiv:1903.09107.
[48] M. Cummins, P. Newman, Appearance-only SLAM at large scale with FAB-MAP 2.0, Int. J. Robot. Res. 30 (9) (2011) 1100–1123.
[49] C. Cadena, D. Gálvez-López, J.D. Tardós, J. Neira, Robust place recognition with stereo sequences, IEEE Trans. Robot. 28 (4) (2012) 871–885.
[50] I. Rocco, R. Arandjelovic, J. Sivic, Convolutional neural network architecture for geometric matching, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6148–6157.
[51] H. Taira, M. Okutomi, T. Sattler, M. Cimpoi, M. Pollefeys, J. Sivic, T. Pajdla, A. Torii, InLoc: Indoor visual localization with dense matching and view synthesis, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7199–7209.
[52] E. Mohedano, K. McGuinness, N.E. O'Connor, A. Salvador, F. Marqués, X. Giro-i Nieto, Bags of local convolutional features for scalable instance search, in: Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, ACM, 2016, pp. 327–331.
[53] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, A. Torralba, Places: A 10 million image database for scene recognition, IEEE Trans. Pattern Anal. Mach. Intell. 40 (6) (2017) 1452–1464.
[54] K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the devil in the details: Delving deep into convolutional nets, 2014, arXiv preprint arXiv:1405.3531.
[55] N. Sünderhauf, S. Shirazi, A. Jacobson, F. Dayoub, E. Pepperell, B. Upcroft, M. Milford, Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free, in: Proceedings of Robotics: Science and Systems XII, 2015.
[56] R. Arandjelović, A. Zisserman, Dislocation: Scalable descriptor distinctiveness for location recognition, in: Asian Conference on Computer Vision, Springer, 2014, pp. 188–204.
[57] H. Jégou, M. Douze, C. Schmid, P. Pérez, Aggregating local descriptors into a compact image representation, in: CVPR 2010-23rd IEEE Conference on Computer Vision & Pattern Recognition, IEEE Computer Society, 2010, pp. 3304–3311.
[58] M.J. Milford, G.F. Wyeth, Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 1643–1649.
[59] A. Babenko, V. Lempitsky, Aggregating local deep features for image retrieval, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1269–1277.
[60] G. Tolias, R. Sicre, H. Jégou, Particular object retrieval with integral max-pooling of CNN activations, 2015, arXiv preprint arXiv:1511.05879.
[61] L. Liu, C. Shen, A. van den Hengel, Cross-convolutional-layer pooling for image recognition, IEEE Trans. Pattern Anal. Mach. Intell. 39 (11) (2017) 2305–2313.
[62] https://www.mapillary.com.
[63] D. Olid, J.M. Fácil, J. Civera, Single-view place recognition under seasonal changes, in: PPNIV Workshop At IROS 2018, 2018.
[64] T. Cieslewski, S. Choudhary, D. Scaramuzza, Data-efficient decentralized visual SLAM, in: 2018 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2018, pp. 2466–2473.
[65] M. Wang, E. Zhu, Q. Liu, Y. Ye, Y. Ming, J. Yin, Intensity filtering and group fusion for accurate mobile place recognition, IEEE Access 6 (2018) 31088–31098.
[66] Camara, Luis G. and Pivoňka, Tomáš and Jílek, Martin and Gäbert, Carl and Košnar, Karel and Přeučil, Libor, Accurate and Robust Teach and Repeat Navigation by Visual Place Recognition: A CNN Approach, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, forthcoming.

Dr. Luis Gomez Camara obtained his Ph.D (2007, London) in Computational Chemistry from Imperial College, an M.Sc. (2006, London) in DSP from Queen Mary and an M.Sc. (2016, Madrid) in Computer Vision from Rey Juan Carlos.

He has worked in the area of automatic music recognition for more than 6 years and as a researcher in computer vision at Airbus in Germany for more than 2.

Since 2018, he has been an active researcher in the IMR lab at CIIRC CTU. His main research activities focus on life-long autonomy of mobile robots, visual place recognition and deep neural networks.

Dr. LiborPřeučil obtained his MEE (1986) and Ph.D (1992) in Technical Cybernetics - Robotics and Computer Vision, both from the Czech Technical University in Prague (CTU). He is an Assistant Professor in Robotics and AI at the Czech Institute of Informatics, Cybernetics, and Robotics (CIIRC CTU).

Libor Přeučil founded and currently heads the Intelligent and Mobile Robotics (IMR) lab at CIIRC (http://imr.ciirc.cvut.cz) and co-founded the Center for Advanced Field Robotics (http://imr.ciirc.cvut.cz/cafr) His primary research interests cover the area of intelligent mobile robotics: namely outdoor robotics, HRI systems, mobile robots in uncontrolled environments and life-long autonomy.