Robotics and Autonomous Systems 143 (2021) 103813

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

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

Learning whole-image descriptors for real-time loop detection and kidnap recovery under large viewpoint difference ☆

Manohar Kuse *, Shaojie Shen

Robotics Institute, Hong Kong University of Science and Technology, Clear Water Bay Road, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 15 April 2019
Received in revised form 13 April 2020
Accepted 18 May 2021
Available online 28 May 2021

Keywords:
Kidnap recovery
Loop closure
VINS
Whole image descriptor

ABSTRACT

We present a real-time stereo visual-inertial-SLAM system which is able to recover from complicated kidnap scenarios and failures online in realtime. We propose to learn the whole-image-descriptor in a weakly supervised manner based on NetVLAD and decoupled convolutions. We analyze the training difficulties in using standard loss formulations and propose an allpairloss and show its effect through extensive experiments. Compared to standard NetVLAD, our network takes an order of magnitude fewer computations and model parameters, as a result runs about three times faster. We evaluate the representation power of our descriptor on standard datasets with precision–recall. Unlike previous loop detection methods which have been evaluated only on fronto-parallel revisits, we evaluate the performance of our method with competing methods on scenarios involving large viewpoint difference. Finally, we present the fully functional system with relative computation and handling of multiple world co-ordinate system which is able to reduce odometry drift, recover from complicated kidnap scenarios and random odometry failures. We open source our fully functional system as an add-on for the popular VINS-Fusion.

1. Introduction

Over the past decade, the SLAM (Simultaneous Localization and Mapping) community has made amazing progress towards increasing the specificity of the odometer and building usable maps of the environment to assist robots in various planning tasks. Systems using visual and inertial information fusion have been a contemporary theme towards reducing drift to less than 0.5% of the trajectory length [1]. Identifying a revisit to a place presents an opportunity to reduce the drift further and also to recover from kidnap scenarios. General place recognition, however, remains an extremely challenging problem [2] due to myriad ways in which visual appearance of a place varies. In our own daily experience, humans describe places to fellow humans as a collection of objects, their color cues, their spatial locations and so on, thereby allowing to disambiguate places even if they approach the place from a very different viewpoint. Ideally, the loop detection module should describe a scene in this context. Humans probably do not rely on corner features (a common technique in use for loop detection in existing SLAM systems) to identify a place, instead we humans, most likely represent the scene as a whole in a semantic sense. The proposed system builds on this motivation.

In this work, we propose the use of a framework which learns whole-image descriptors without explicit human labeling to represent a scene in a high dimensional subspace for detecting place revisits. We lay special emphasis on the real-time performance and evaluation of the system in context of visual-SLAM.

Popular past works have considered loopclosure under fronto-parallel scenarios. However, place revisits can happen at substantial viewpoint difference. The underlying place recognition module in SLAM systems to identify place revisits occurring at widely different viewpoints. Past systems based on bag-of-visual-words (BOVW) are limited by the underlying low-level feature descriptors. The learned vocabulary (for BOVW) also have difficulty generalizing under adversaries like large viewpoint difference, noise, low light, changing exposure, less texture [2]. The proposed method can learn a representation that generalizes well and can identify place revisits under non fronto-parallel viewpoints.

We compare our method's run-time performance with a popular bag-of-words approach, DBOW [5] and ibow-lcd by [6] along with recently proposed CNN based approaches for place recognition [7–9]. On real sequences, our method delivers a similar recognition performance to NetVLAD but at a 3X lower computational time and an order of magnitude fewer training variables. The major advantage of our system is that it has a high place recall rate thus is able to recover live in real-time from long and

☆ The source code is available at https://github.com/HKUST-Aerial-Robotics/VINS-kidnap.

* Corresponding author.
E-mail address: mpkuse@connect.ust.hk (M. Kuse).

https://doi.org/10.1016/j.robot.2021.103813
0921-8890/

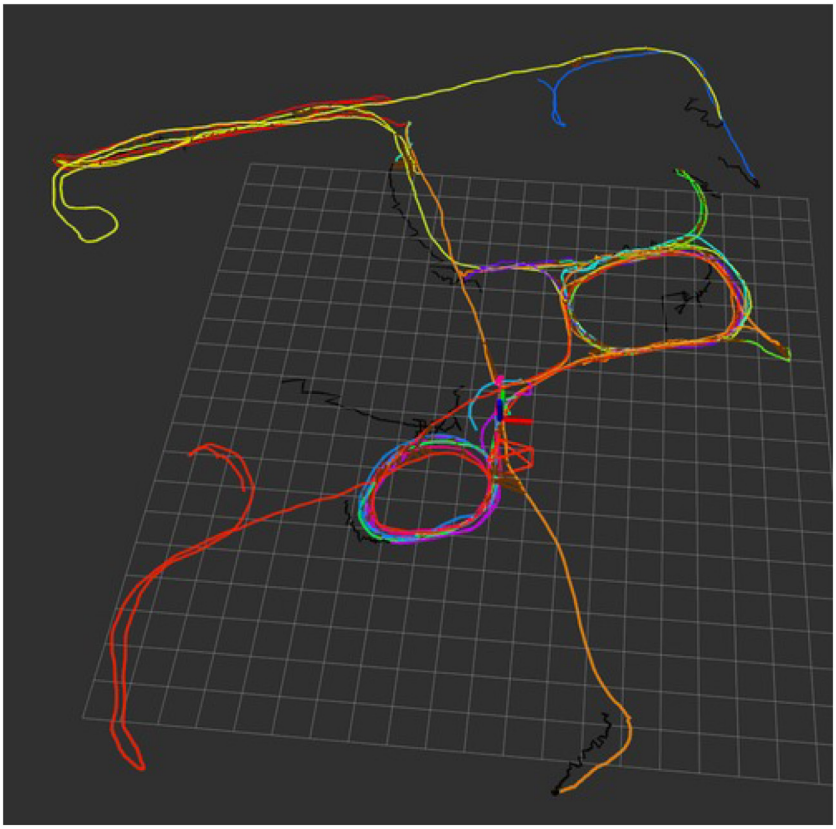


Fig. 1. Shows the corrected trajectories (different colors for different worlds) merged according to the inter-world loop candidates. Note that the merging occurs live (not offline) in real-time as the loop candidates are found. We also note that such cases cannot be handled by Qin et al. [3] which just merges with the world-0 (first world) and ignore any inter-world loop candidates not involving world-0. The *maplab* system [4] provides an online tool, *ROVIOLI* which is essentially a visual-inertial odometry and localization front-end. Although it provides for a console based interface offline for multi-session map merging, it cannot identify kidnaps and recover from them online. This sequence involves multiple kidnaps lasting from 10 s to 30 s. The video for the live run is available at the link: https://youtu.be/3YQF4_v7AEg. Live runs videos are available for more sequences through this link: https://bit.ly/2IkEh3F. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

chained kidnaps by maintaining multiple co-ordinate systems and their relative poses.

Our paper is organized as follows. In Section 2, we start by reviewing approaches in the Visual Place Recognition (VPR) community and some recent loopclosure methods used in Visual-SLAM community. Next, in Section 3, we identify the issue of unstable learning in the original NetVLAD implementation which uses the tripletloss and we propose an allpairloss function to alleviate this issue. In Section 4.1 we present our implementation details for deployment as a visual-SLAM subsystem which includes the place recognition module, the datastructure for handling multiple co-ordinate systems and recovery from kidnap. In Section 5, we presents comparative experiments to this effect. Finally, we present our entire system which is available as a pluggable module to the popular VINS-Fusion [10].

2. Literature review

We recognize that visual place recognition (VPR) and loop-closure detection in SLAM are related problems. Here we first review recent advances from VPR community and then review state-of-the-art loop-closure methods.

In the context of VPR, Sunderhauf et al. [7] pioneered the use of ConvNet features. Compared to SeqSLAM [11] and FAB-MAP [12,13] use of features from pretrained network results in better precision–recall performances on standard VPR datasets (Nordland, Gardens Point, St. Lucia and Campus). In their subsequent work, Sunderhauf et al. [14] proposed to use region proposals and extract ConvNet features on each of the regions. Arandjelovic et al. [15] proposed a trainable feature aggregation layer which mimics the popular VLAD (Vector of Locally Aggregated Descriptor). While impressive performance was obtained, these methods rely on nearest neighbor search for retrieval. The image descriptor being very high dimensional (eg. 32K dimensional for [15], 64K for [7]), these methods perform various dimensionality reduction techniques to make nearest neighbor search feasible in reasonable time with some hit to the retrieval performance. WPCA was used by [15] which involve storage of a 32Kx4K matrix costing about 400 MB.

More recently Khaliq et al. [16] proposed an approach which make use of region-based features from a light-weight CNN architecture and combines them with VLAD aggregation. The approach from Chen et al. [17,18] identifies key landmark regions directly from responses of VGG16 network which was pretrained on image classification task. For regional features encoding, bag-of-words was employed on a separate training dataset to learn the codebook. The approach by Hou et al. [19] is very similar to [17]. The Disadvantage of using pretrained models learned on ImageNet object classification, for example, puts more emphasis on objects rather than the nature of the scene itself. Other works in this context include [20–27]. For a more detailed summary of the works in place recognition we direct the readers to survey on place recognition/instance retrieval [2,28]. We summarize the literature in Table 1.

Although CNN based techniques are considered as state-of-the-art in retrieval and place recognition tasks, they are still

Table 1
A summary of visual place recognition literature. SPF=Sparse Point Features. BOW=Bag-of-words.

Representation	Retrieval	Method description
SPF-real	BOW	[12,13] soft-real-time (can run @5-10hz)
SPF-binary	BOW	[5,29,30] real-time (10 hz or more)
SPF-real	Inc-BOW	[31–33] soft-real-time (1–5 Hz). [34,35] make use of temporal information to form visual words scene representation.
SPF-binary	Inc-BOW	[6,36–38] soft-real-time to 1–5 Hz processing
SPF	graph	[39]
Pretrained-CNN	NN	[7,20,40] provide for real-time descriptor computation (10–15 hz). dimensionality reduction accomplished at 5 hz, NN with 64K dim is really slow, NN after dimensionality reduction (4000d) is about 5–15 hz.
Pretrained-CNN	BOW	[17–19]
Custom-CNN	NN	[15,22–24] provides for real-time descriptor computation. dim-reduction and NN search are bottle necks.
Custom-CNN with region-proposals	regionwise-NN	[14] very slow representation vector computation. [16] region descriptor encoding computation 2–3 Hz. Reported matching times is several seconds.
Unsupervised Learning	NN	[8,9,21] descriptors are not descriptive enough after dim-reduction. real-time desc computation.
Intensity	NN	[11,41]
agnostic	optimization	[42–44] generally slow.

disconnected from overall SLAM and loop-closure detection problems. Commonly employed loop detection methods in state-of-the art SLAM systems rely on sparse point feature descriptors like SIFT, SURF, ORB, BRIEF etc. for representation and an adaptation of BoVW for retrieval. While BoVW provides for an scalable indexed retrieval, the performance of the system is limited by the underlying image representation. Such factors as the quantization in clustering when building vocabulary, occlusions, image noise, repeated structures also affect the retrieval performance.

FAB-MAP [12,13], DBOW2 [5] and others [29,30] rely on a visual vocabulary which is trained offline, while recent methods like OVV [32], IBuILD [38], iBOW-LCD [6], RTAB-MAP [33] and others [31,36,37,45] rely on online constructed visual vocabulary. Authors have also made use of whole-image-descriptors in loop-closure context [11,41,42]. Works in the context of loopclosures in SLAM which make use of learned feature descriptors are: [8, 21]. Merril and Huang [8] learned an auto-encoder from the common HOG descriptors for the whole image. Other miscellaneous work related to our localization system are [46–50].

Some works have also built full SLAM system with multi-session map merging capability. The *maplab* system [4] provides an online tool, *ROVIOLI* which is essentially a visual-inertial odometry and localization front-end. Although it provides for a console based interface offline for multi-session map merging, it cannot identify kidnaps and recover from them online. This is the major distinguishing point of our system. Also the relocalization system by Tong et al. [3] can merge multiple sessions live it only merges with the first co-ordinate frame, any loop connections between co-ordinate systems not involving the first co-ordinate systems are ignored. Our system on the other hand is able to maintain multiple co-ordinate systems and their relative poses, set associations and merge the trajectories online in real-time.

We summarize our contributions:

- A fully functional system, as an add-on for VINS-Fusion, which uses whole-image descriptor for place representation and recovery from odometry drifts, kidnap and failures live and in real-time. Our learning code[1] and VINS-Fusion addon ROS package[2] are open sourced.
- A novel cost function which deals with the gradient issue observed in standard NetVLAD training.
- Decoupled convolutions instead of standard convolutions result in similar performance on precision–recall basis but at a 3X lower computation cost and about 5-7X fewer learnable parameters, making it ideally situated for real-time loopclosure problems.
- Squashing channels of CNN descriptors instead of explicit dimensionality reduction of image descriptor for scalability. Even a 512-D image descriptor gives reasonable performance.

3. Learning place representation

In this section, we describe the training procedure. We start by reviewing VLAD and NetVLAD [15] (Section 3.1). We see these methods as a way for pixelwise featuremap aggregation. Next we describe the learning issues associated with the triplet ranking loss function. To mitigate this issue we propose to use a novel all-pair loss function (Section 3.2). We provide an intuitive explanation along with experimental evidence on why the proposed all-pair loss function leads to faster and stable training.

3.1. Review of VLAD and NetVLAD

Let $\mathbf{h}_{\mathbf{u}}^{(I)}(d)$ be the dth dimension ($d = 1 \cdots D$) of the image feature descriptors for image I (width W and height H) at pixel $\mathbf{u} := (i, j), i = 1, \ldots, W'; j = 1, \ldots, H'$. These per pixel CNN-feature descriptors are assigned to one of the K clusters (K is a fixed parameter, we used 16, 48, 64 in our experiments), with $\mathbf{c}_k \in \Re^D, k = 1 \ldots K$ as cluster centers. The VLAD [15] representation is $D \times K$ matrix, $\mathbf{V} = [\boldsymbol{\vartheta}_1, \boldsymbol{\vartheta}_2, \ldots, \boldsymbol{\vartheta}_K]$, defined as the sum of difference between local descriptor and assigned cluster center,

$$\boldsymbol{\vartheta}_k = \sum_{\forall \mathbf{u}} a_k(\mathbf{h}_{\mathbf{u}}^{(I)}) \times (\mathbf{h}_{\mathbf{u}}^{(I)} - \mathbf{c}_k) \tag{1}$$

where $a_k(.)$ denotes a scalar membership indicator function of the descriptor $\mathbf{h}_{\mathbf{u}}^{(I)}$ in one of the K classes. Arandjelovic et al. [15] proposed to mimic VLAD in a CNN-based framework. In order that the cluster assignment function, $a_k(.)$, be differentiable and hence learnable with back propagation they defined an approximation of the assignment function $a_k(.)$ using the softmax function. For brevity, we write, $\mathbf{h}_{\mathbf{u}}^{(I)}$ as $\mathbf{h}$:

$$\begin{aligned} \hat{a}_k(\mathbf{h}) &= \frac{e^{-\alpha \, \|\mathbf{h}-\mathbf{c}_k\|}}{\sum_{k'=1}^{K} e^{-\alpha \, \|\mathbf{h}-\mathbf{c}_{k'}\|}} \\ &= \sigma(\mathbf{r})_k \end{aligned} \tag{2}$$

[1] https://github.com/mpkuse/cartwheel_train
[2] https://github.com/HKUST-Aerial-Robotics/VINS-kidnap

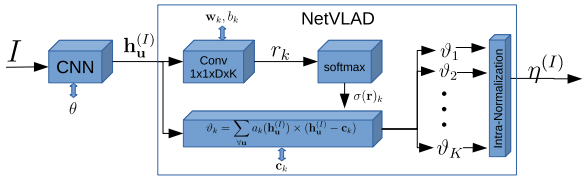


Fig. 2. Notations and computations for the whole-image descriptor. An image is fed into the CNN followed by the NetVLAD layer. We experiment with VGG16 and propose to use decoupled convolution for its speed. Additionally for dimensionality reduction we propose channel-squashing. Our fully convolutional network, with K = 16 produces a 4096-dimensional image descriptor (without channel squashing) and a 512-dimensional image descriptor (with channel squashing). In terms of number of floating point operations (FLOPs) for a 640 × 480 input image our proposed network is about 25X faster, real computational time is about 3X faster. Details in Section 3.

where $r_k = \mathbf{w}_k^T\mathbf{h} + b_k$, $\mathbf{w}_k = 2\alpha\mathbf{c}_k$, $b_k = -\alpha\|\mathbf{c}_k\|^2$. $\sigma(\mathbf{r})_k$ is the softmax function. r_k can be computed with convolutions. $\mathbf{w}_k$, b_k and $\mathbf{c}_k$ are learnable parameters in addition to the CNN parameters θ. Fig. 2 summarizes the computations and notations. Each of the vectors corresponding to K clusters is individually unit normalized and then the whole vector is unit normalized. This is referred in the literature as Intra-normalization which reduce the effect of burstiness of visual features [51]. Thus, scene descriptor $\eta^{(I)}$, of size $D*K$ is produced using a CNN and the NetVLAD layer,

$$\eta^{(I)} = \aleph(\{\mathbf{h}_\mathbf{u}^{(I)}\}) \tag{3}$$

In general any base CNN can be used and the NetVLAD mechanism can be thought of aggregating the CNN pixel-wise descriptors. For experiments, we use the VGG16 network [52]. Additionally, following [53] we propose to use the decoupled convolution, ie. a convolution layer is split into two layers, the first of which does only spatial convolution independently across all the input channels. Second of the two layers does 1 × 1 convolution on the channels. This has been found to boost running time at marginal loss of accuracy for object categorization tasks. We also propose to reduce the dimensions by quashing channels with learned 1 × 1 convolutions rather than reduce the dimensions of the image descriptor as has been done by the original NetVLAD paper. This eliminates the need to store the whitening matrix (as done by [15]). At the run time it eliminates the need for a large matrix–vector multiplication for dimensionality reduction.

3.2. Proposed all-pair loss function

To learn the parameters of the CNN (θ) and of the NetVLAD layer ($\mathbf{w}_k$, b_k and $\mathbf{c}_k$), the cost function needs to be designed such that, in the dot product space, η corresponding to projections of the same scene (under different viewpoints) appear as nearby points (higher dot product value, nearer to 1.0). Let $\eta^{(I_q)}$ be the descriptor of the query image I_q. Similarly, let $\eta^{(P_i)}$ and $\eta^{(N_j)}$ be the descriptors of ith positive and jth negative sample respectively. By positive sample, we refer to a scene which is same as query image scene but imaged from a different perspective. By negative sample, we refer to a scene which is not the same place as the query image. Let the notation, $\langle\eta^{(a)}, \eta^{(b)}\rangle$, denote the dot product of two vectors.

Following [15] we use multiple positive and negative samples $\{I_q, \{P_i\}_{i=1,\dots,m}, \{N_j\}_{j=1,\dots,n}\}$ per training sample, however with a novel all-pair loss function. We provide an intuitive explanation for the superiority of the proposed loss function over the standard triplet loss used by [15] for training. We also provide corroborative experimental evidence towards our claims. The commonly used triplet loss function can be rewritten in our notations as, $L_{triplet-loss}$:

$$\sum_j \max\big(0,\ \langle\eta^{(I_q)}, \eta^{(N_j)}\rangle - \min_i(\langle\eta^{(I_q)}, \eta^{(P_i)}\rangle) + \epsilon\big) \tag{4}$$

where ϵ is a constant margin. Note that [15] preferred to define the loss function in Euclidean space rather than the dot product space. Using any of the spaces is equivalent since for unit vectors, $\mathbf{a}$ and $\mathbf{b}$, the dot product, $\langle\mathbf{a}, \mathbf{b}\rangle$, and the squared Euclidean distance $d(\mathbf{a}, \mathbf{b})$, are related as, $d(a, b) = 2(1-\langle a, b\rangle)$ with a negative correlation. This has been taken care of by flipped sign in our optimization problem compared to the one used by Arandjelovic et al. [15]. This loss function is the difference between the worst positive sample, ie. $\min_i\langle\eta^{(I_q)}, \eta^{(P_i)}\rangle$ and the query with every negative sample.

In an independent study by Bengio et al. [54], it was observed that for faster convergence, it is crucial to select triplets from the training dataset, that violate the triplet constraint, ie. result in as few zero-loss as possible. They demonstrated that these zero-loss scenarios lead to zero gradients which in turn slows the training. They suggested to provide easier samples in early iterations and harder samples in later iteration to speed up the learning process. To this effect, Schroff et al. [55] proposed a strategy to select triplets using recent network checkpoints, every n (say 1000) training iterations. Instead of using a complicated strategy as done by [55], we rely on a well-designed loss function which gives this effect. Thus, we define a novel loss function based on all-pair comparisons of positive and negative samples with the query image. The proposed loss function is relatively harder to satisfy (resulting in fewer zero loss samples), hence its higher discriminatory power compared to the triplet loss (see Fig. 6).

In order to learn highly discriminative descriptors, we want the similarity of query sample ie. $\eta^{(I_q)}$ with positive samples be more than the similarity between query sample and the negative samples. Let us consider two cases (a) $\langle\eta^{(I_q)}, \eta^{(N_j)}\rangle > \langle\eta^{(I_q)}, \eta^{(P_i)}\rangle$; (b) $\langle\eta^{(I_q)}, \eta^{(N_j)}\rangle < \langle\eta^{(I_q)}, \eta^{(P_i)}\rangle$. Case-b is what we prefer so we do not want to have a penalty (want to have zero loss) for its occurrence. Case-a is opposite of what we prefer thus we add a penalty proportional to the magnitude of the dot product to discourage this event. We propose to add a penalty term as above for all the pairs where the conditions do not hold. For effective learning we propose to compute the loss over every pair of positive and negative sample. The final loss function for one learning sample ($\{I_q, \{P_i\}_{i=1,\dots,m}, \{N_j\}_{j=1,\dots,n}\}$) is given as, $L_{proposed}$:

$$L = \sum_{i=1}^{m}\sum_{j=1}^{n} \max(0,\ \langle\eta^{(I_q)}, \eta^{(N_j)}\rangle - \langle\eta^{(I_q)}, \eta^{(P_i)}\rangle + \epsilon) \tag{5}$$

The motivation and the effect of the proposed loss function on termination of learning is summarized in Fig. 3. We further note that the proposed loss function is harder to satisfy (giving a positive penalty) compared to the loss function used by Arandjelovic et al. [15] (ie. Eq. (4)). This has been experimentally observed (see Fig. 6). The major drawback of using an easy to satisfy penalty function is the vanishing gradients problem [54], which slows the speed of learning. This is because a zero-loss sample results in zero-gradient during back-propagation.

3.2.1. Loss function in matrix notation

For fast and efficient training we express the loss function (Eq. (5)) in a matrix notation. We firstly define $\Delta_\mathbf{P}^q$ to represent dot product between the sample query η_q and descriptors of each of the positive samples

$$\Delta_\mathbf{P}^q = \begin{bmatrix} \langle\eta^{(I_q)}, \eta^{(P_1)}\rangle \\ \vdots \\ \langle\eta^{(I_q)}, \eta^{(P_m)}\rangle \end{bmatrix} \tag{6}$$

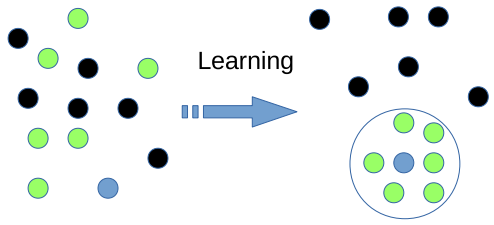


Fig. 3. Illustration of the effect of learning with proposed loss function. Descriptor of query image ($\eta^{(I_q)}$, in blue). Descriptors of positive set ($\eta^{(P_i)}$, in green) and negative set ($\eta^{(N_j)}$, in black). See also Eq. (5). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Next, we define $\Delta^q_{\mathbf{N}}$ to represent dot product between the sample query and descriptors of each of the negative samples.

$$\Delta^q_{\mathbf{N}} = \begin{bmatrix} \langle \eta^{(I_q)}, \eta^{(N_1)} \rangle \\ \vdots \\ \langle \eta^{(I_q)}, \eta^{(N_n)} \rangle \end{bmatrix} \tag{7}$$

Let $\mathbf{1}_n$ denote a column-vector of size n and $\mathbf{1}_m$ denote a column-vector of size m with all entries as 1s. Also define $\mathbf{0}_{m\times n}$ as null-matrix of dimensions $m \times n$. The max(.) operator is a point-wise operator. Now we note that Eq. (5) can be expressed in matrix notation as:

$$\mathbf{L} = \max\left(\mathbf{0}_{m\times n}, \mathbf{1}_m(\Delta^q_{\mathbf{N}})^T - \Delta^q_{\mathbf{P}}\mathbf{1}_n^T + \epsilon \mathbf{1}_m\mathbf{1}_n^T\right) \tag{8}$$

3.3. Training data

In order to train the scene descriptor, only requirement on the data is that we be able to draw positive sample images (views of the same physical scenes) and negative sample images (images of different scenes). One possible way is to bootstrap a video sequence with existing methods for loopclosure detection. Such a preprocessed sequence might not be useful for localization but can provide enough information to draw positive and negative samples for learning a whole-image-descriptor with the proposed method. Several walking, driving, drone videos etc. available on video sharing websites can be used for learning. Another way could be to use 3D mesh-models and render views with nearby virtual-camera locations to obtain positive samples. With the advent of crowd sourcing street scenes and availability of services like mappilary,[3] it is easily possible to assemble a much larger dataset for training. Faster and discriminative learning is even more crucial when making use of such larger training datasets. We differ this until our future work.

For our experiments be comparable with existing methods, we use the Pittsburgh (Pitts250k) [56] dataset which contains 250k images from Google's street-view engine. It provides multiple street-level panoramic images for about 5000 unique locations in Pittsburg, Pennsylvania over several years. Multiple panoramas are available at a particular place ($\approx$ 10m apart) sampled approximately 30 degrees apart along the azimuth. Another similar dataset is the TokyoTM dataset [56].

3.4. Training hyperparameters

The CNN-learnable parameters are initialized with the Xavier initialization [57]. We initialize NetVLAD parameter $\mathbf{c}_k$ as unit vectors drawn randomly from a surface of a hypersphere. b_k and $\mathbf{w}_k$ are coupled with $\mathbf{c}_k$ at initialization. However, as learning progresses these variables are decoupled. We use the AdaDelta solver [58] with batch size of $b = 4$ (each batch with $m = 6$ positive samples and $n = 6$ negative samples). This configuration takes about 9GB of GPU memory during training. We stop the training at 1200 epochs. Our 1 epoch is 500 randomly drawn tuples from the entire dataset. The learning rate is reduced by a factor of 0.7 if loss function does not decrease in 50 epochs and a regularization constant is set to 0.001 (to make regularization loss about 1% of fitting loss). Data augmentation (rotation, affine scale, random cropping, random intensity variation) is used for robust learning, which we begin after 400 epochs. This explains the rise in the loss function values in our experiments in Figs. 8, 9, 10.

The output descriptor size is $K \times D$. K is the number of clusters in NetVLAD, we use K = 16,32,64. D is the number of channels of the output CNN. For both VGG16 and the decoupled network it is 512. Some other authors notably Arandjejovic [15] and Sunderhauf [7] have used whitening-PCA and gaussian random projections respectively to reduce the dimensions of the image descriptor from about 64K to 4K. We suggest to use learnable squashing channels (to say 32) with channelwise convolutions before feeding the pixelwise descriptors to the NetVLAD layer rather than reduce the dimensionality of the image descriptor. We have experimentally compared the effect of this channel squashing in Figs. 14 and 15.

4. Deployment

Our system, which is able to correct drifts of VIOs, recover from long kidnaps and system failures online in real-time, is available as a add-on to the popular VINS-Fusion. In general, our system can work with any VIO system. It is worth nothing that the *maplab* system's [4] *ROVIOLI* cannot identify kidnaps and recover from them online, although it can merge multiple sequences offline (when all the sequences are known).

In this section we describe our multi-threaded software architecture (Fig. 4). We use the consumer-producer paradigm for real-time deployment of our system. We start by describing the image level descriptor extraction and comparison. After that, we describe the details of pose computation at the loopcandidate image pair. Next we describe how our system is able to recover from kidnaps by keeping track of the disjoint co-ordinate systems and switching off the visual-inertial odometry when no features can be reliably tracked.

4.1. Descriptor extraction and comparison

We propose a pluggable system to the popular VINS-Fusion[4] by Qin et al. [10]. Our system receives keyframes from the visual-inertial odometry sub-system to produce loopclosure candidates. We use a naive store-and-compare strategy to find loopcandidates. The descriptors at all previous keyframes, ie. $\eta^{(I_t)}$ $t = 1, \ldots, t$ are stored indexed with time. When a new keyframe arrives, say $\eta^{(I_{t+1})}$, we perform $\langle \eta^{(I_{t_i})}, \eta^{(I_{t+1})} \rangle$ $i = 1, \ldots, t - T$. T is typically 150, ie. ignore latest 150 frames (or 15 s) for loopclosure candidates. These are a measure of likelihoods for loopclosure at each of the keyframe timestamps. We accept the loopclosure hypothesis if the query score is above a set threshold (fixed for all the sequences) and if three consecutive queries retrieve descriptors within six keyframes of the first of the three queries. In a real implementation, the threshold can be set a little lower and wrong hypothesis can be eliminated with geometric verifications heuristics. We note that loopcandidates with large viewpoint difference provide a formidable challenge for tracked

[3] https://www.mapillary.com/

[4] https://github.com/HKUST-Aerial-Robotics/VINS-Fusion

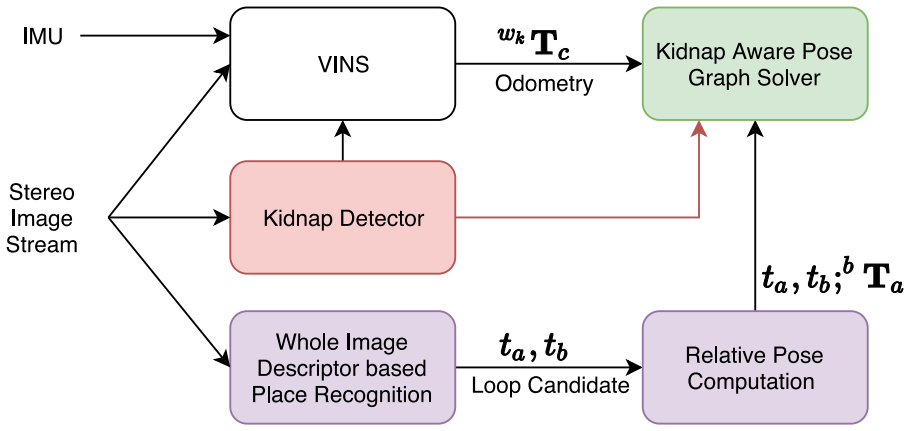


Fig. 4. System overview.

feature matching between the two views. In this work for comparison of descriptor's precision–recall, we do not perform any geometric verification.

Our naive comparison (matrix–vector multiplication) takes about 50 ms for comparison with 4000 keyframes (about 10 min sequence) on a desktop CPU with image descriptor dimension of 8192 and about 10 ms for 512 dimensional image descriptor. While the comparison times grow unbounded as number of keyframes increase, the objective of this paper is to demonstrate the representation power of learned whole image descriptors over the traditional BOVW on sparse feature descriptor loopclosure detection framework and the recently proposed CNN-based image descriptors in terms of detection under large viewpoint difference. Dealing with scalability could be a future research direction. In our opinion scalability can be achieved by sophisticated product quantization approaches similar to Johnson et al. [59] or by maintaining a marginalized set of scene descriptors, along with scene object labels and dot product comparison on this smaller subset.

4.2. Feature matching and pose computation

In this section we describe the task of metric scale relative pose computation given a loopcandidate pair. For computation of reliable pose estimates we make use of the GMS-Matcher [60] as a robust correspondence engine. The GMS-matcher propose a simple grid based voting scheme for finding fast correspondences. This matcher provides reliable matches even under large viewpoint difference. This can be attributed to the grid-based voting scheme that it implements, has the effect of eliminating point correspondences if nearby points in one view do not go to nearby points in the second view. Computationally it takes about 150–200 ms per image pair (image size 640 × 480). Since we only need to compute these correspondence on loopcandidates and also since we use a multi-threaded implementation, this higher computational time does not stall our computational pipeline. Note that in the event of loop detections under large viewpoint difference, the tracked features in a SLAM system do not provide enough information for feature computation. Some other related recent works on CNN based feature correspondence include InLoc [61], Neighborhood consensus network [62]. Although these approaches present impressive results, they involve storage of allpair dense pixel-level features which currently cannot be accomplished in realtime.

Once the feature correspondences are produced we make use of direct perspective-n-point (PNP) method by Hesch and Roumeliotis [63] for pose computation.[5] We reject a loopcandidate if insufficient number of correspondences are produces (we use a threshold of 200 correspondences). We use random sampling and consensus (RANSAC) to make the PNP robust under spurious feature correspondences that may occasionally occur. For about 10 RANSAC iteration it takes on average 5 ms to compute the pose from approximately 2000 feature correspondences. We make use of stereo geometry for computing 3D points at loopclosures.

In cases where the queue size is small and computational resources sufficient we also compute the pose by iterative closest point (ICP) method in addition to the PNP. The 3D points for both the images of the loopcandidates is obtained from stereo geometry in this case. If the pose computations are not consistent from the two methods we reject the loopcandidate.

4.3. Kidnap detection and recovery

In our system we also deal with a kidnap recovery mechanism. By kidnap we refer to the camera's view being blocked and the camera teleported to another location several 10 s to 100 s of meters away in 10 s to 100 s of seconds. The teleported location may or may not be a previously seen location. We make use of a simple criterion like the current number of tracked features falling to a very low value (like less than 10) to determine if the camera system is kidnapped. Once we determine that the camera system has been kidnapped we stop the visual inertial odometry subsystem. When sufficient number of features are again being tracked we reinitialize the odometry/sensor fusion system. It is to be noted that in such a case it starts with a new co-ordinate reference. Hence forth we refer to the new co-ordinate systems as world-0 (w_0), world-1 (w_1), world-2 (w_2) and world-k (w_k) in general. We denote the nodes n by a superscript to identify the world that it is in. For instance $n_i^{(k)}$, means the ith node (i is the global index of the node) is in the kth world. The odometry pose of the node is denoted as ${}^{(k)}\mathbf{T}_i$.

The incoming loopcandidate pair can be categorized into two kinds (refer to Fig. 5 for a visual explanation). (a) Intra worlds (eg. $n_i^{(k)} \leftrightarrow n_j^{(k)}$) and (b) Inter worlds ($n_i^{(k)} \leftrightarrow n_j^{(k')}$).

4.3.1. Direct pose computation between w_k and $w_{k'}$

The inter-world loopcandidates $n_i^{(k)} \leftrightarrow n_j^{(k')}$ can be used to infer the relative poses between the co-ordinate system w_k and $w_{k'}$. In this section, we describe the computation of the relative pose between the worlds k and k', ie. ${}^{(k)}\mathbf{T}_{(k')}$ from the relative pose between the two nodes (${}^i\mathbf{T}_j$) and the odometry poses of the nodes in their respective worlds ie. ${}^{(k)}\mathbf{T}_i$ and ${}^{(k')}\mathbf{T}_j$ respectively.

$$ {}^{(k)}\mathbf{T}_{(k')} = {}^{(k)}\mathbf{T}_i \times {}^i\mathbf{T}_j \times \left({}^{(k')}\mathbf{T}_j\right)^{-1} \tag{9} $$

[5] Implementation from *theia-sfm* (http://www.theia-sfm.org/).

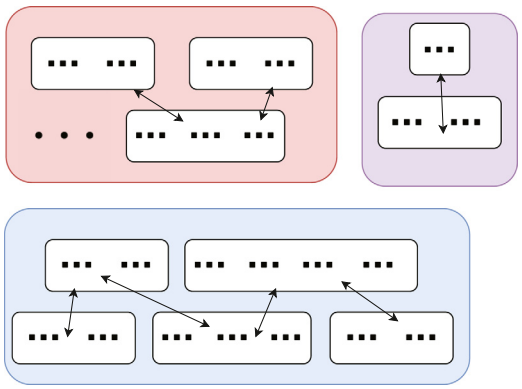


Fig. 5. The solid black squares represent the nodes in the pose graph. Arrows show the loopcandidates. The white rectangles show each of the individual worlds. The colored rectangular enclosures are the worlds belonging to the same set. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

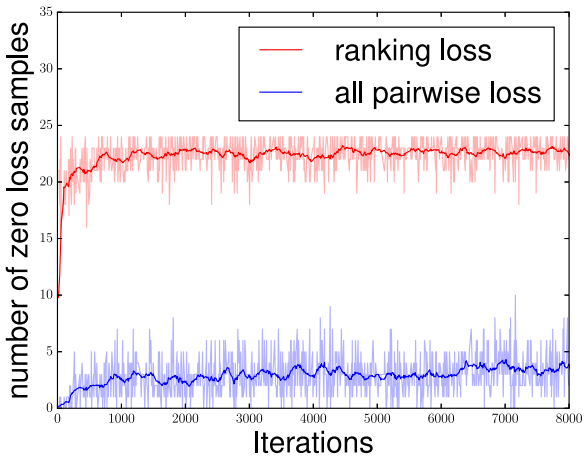


Fig. 6. The number of batches with zero loss as iterations progress for learning with proposed cost function (in blue, Eq. (5)) compared to using the triplet ranking loss [15] (in red, Eq. (4)). This experiments used a batch size of 24 with gradient commutation. Having a higher count for zero-loss samples is detrimental to learning as it leads to zero-valued gradients. Best viewed in color. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.3.2. Indirect pose computation between w_k and w_{k_1}

It is easy to see that if we have two inter-world loopcandidates like: $n_i^{(k)} \leftrightarrow n_j^{(k')}$ and $n_{i_1}^{(k_1)} \leftrightarrow n_{j_1}^{(k')}$, the three worlds w_k, $w_{k'}$ and w_{k_1} are said to be in same set. It is also possible to indirectly infer the relative pose between worlds w_k and w_{k1} even though no loopcandidate exists between these two sets. This estimate is needed for correctly initializing the poses to solve the pose graph optimization problem.

We make use of the data structure disjoint sets [64] to maintain the world information that are in the same set. The advantage of the disjoint set datastructure is it provides for a constant time set-union and sub-linear time set-association query. Each world starts in its own set, everytime we encounter the inter-world looppair we merge these two sets of the worlds into a single set.

When we assert that two different worlds are in the same set, we imply that a relative transform between these two worlds can be determined. However that a loopcandidate between these two pairs of worlds may or may not exist. In case no loopcandidate exists between the two worlds but these two worlds are in the same set, the relative poses between the worlds can be determined by finding a graph-path between the two worlds and chaining the relative pose estimates between the adjacent world pairs in the path.

In a general scenario, this can be accomplished by constructing a directed graph of the worlds with nodes being the worlds in the same set and edges being the relative poses between these two worlds, ie. ${}^{(k)}\mathbf{T}_{(k')}$. A breadth-first search on this graph is sufficient to determine an estimate of relative poses between arbitrary pairs of worlds by chaining the relative poses of the path generated by the graph search.

4.4. Implementation details

Our full system employs multiple threads. It uses the producer–consumer programming paradigm for managing and processing the data. In our system, thread-1 produces image descriptors of all incoming keyframe images. Thread-2 consumes the image descriptors to produce candidate matches. Thread-3 consumes the candidate matches to produce feature correspondences. Thread-4 uses the feature correspondence to produce the relative pose ${}^{i}\mathbf{T}_{j}$. Thread-5 monitors the number of tracked features to know if the system has been kidnap. Thread-6 uses the loopcandidates and their relative poses to construct the disjoint set data structure and maintain the relative poses between multiple co-ordinate systems as detailed in Sections 4.3.1 and 4.3.2. Thread-7 incrementally constructs and solves the pose graph optimization problem while carefully initializing the initial poses and making use of poses between the worlds. Our pose graph solver is based upon the work of Sunderhauf et al. [65]. A separate thread is used for visualizing the poses. Even though we use 7 threads, the effective load factor on the system is about 2.0. This means about two cores are occupied by our system (this does not include the processing for VINS-Fusion Odometry System).

We demonstrate the working of our full system (see Fig. 1). It is worth noting that in addition to reducing the drift on account of loopclosures, our implementation can reliably identify and recover from kidnap scenarios lasting longer than a minute online in realtime. We attribute such robustness to the high recall rates of the NetVLAD based image descriptor engine. The results in regard to the operation of the full system can be found in the attached video. We record our own data for demonstrating the kidnap cases. Some of the kidnap cases are labeled hard which include the need for indirect inference which is not available in the previous slam systems including the relocalization system from Qin et al. [3].

5. Experiments

In this section we describe our experiments. We evaluate the effect of using the allpair loss function compared to the commonly used triplet loss function (Section 5.1), while keeping the same backend CNNs. Next the effect on running time and memory consumption by the use of decoupled convolutions are tabulated (Section 5.2). In Section 5.3 we evaluate the precision–recall performance of the proposed algorithm with other competing methods in the SLAM community and in the visual place recognition community. Finally in Section 5.4 we evaluate the performance of the proposed method on a real world SLAM sequences captured under challenging conditions especially revisits occurring with non-fronto parallel configurations and in-plane rotations. For demonstration of our full system on real world sequences refer to the video attachment.

5.1. Evaluation metrics for loss function

We evaluate the effect of the proposed loss function on NetVLAD compared to the original triplet loss which was used

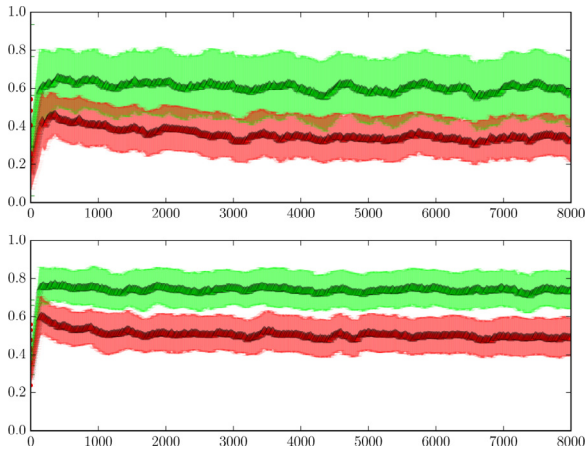


Fig. 7. Showing spreads ($\mu \pm \sigma$) of $\langle \eta_q, \eta_{P_i} \rangle$ (in green) and spreads of $\langle \eta_q, \eta_{N_i} \rangle$ (in red) as the learning progresses. Fig. 7 (top) corresponds to [15], with the triplet loss function. Fig. 7 (bottom) corresponds to the proposed allpair loss function. We observe a lower spread amongst positive samples and larger separation between positive and negative samples. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

in [15]. We evaluate our proposed loss function using (a) relative loss declines (b) number of correctly identified pairs from the validation tuple. We also plot the variance dot products of the positive samples amongst themselves. For validation we use the Pitts30K dataset and for training we use the TokyoTM dataset.

Similar to the training tuple, a validation tuple contains a query image, nN (= 6) negative samples and nP (= 6) positive samples. For evaluation, we propose to count the number of correctly identified image pairs which were actually similar to query images (ground truth) and identified as similar based on the image descriptor dot product scores. Ideally, the evaluation metric should be the SLAM sequences however it becomes infeasible to evaluate with SLAM sequence every say 20 epochs so we stick to this workaround. We plot these metrics for the training data and the validation data as the iterations progress. See Figs. 8, 9, 10 for the plots. The summary of the observation:

- Using the same backend convolutional network, the same parameters of the netvlad layer, and same learning hyperparameters, the network trained with the proposed allpair loss function performs better as evaluated against the validation metric, count of pairs correctly identified.
- It can also be inferred that the gradients obtained from the use of proposed all-pair loss function are more stable, hence the faster convergence.
- Our all pair loss function was found to perform better even when using the decoupled convolutions, decoupled convolution with channel quashing vs the original VGG16 network.

5.1.1. Number of zero loss tuples

In this experiment, we train with batch size 24. But since this will not fit in the GPU memory we use gradient commutation. We plot the number of zero loss sample as iterations progress in Fig. 6. When using the triplet loss, we get more number of zero loss samples. This results in zero-gradient updates and hence slow learning compared to proposed allpairloss. This can be attributed to allpairloss function being harder to satisfy resulting in better gradients during training.

5.1.2. Spreads of positive and negative samples

As experimentally observed in Fig. 7, the use of proposed allpair loss function results in a more discriminative image descriptor as compared to the network trained with the triplet loss. We observe a lower spread amongst the positive samples and a larger separation in positive and negative samples. This has the effect that the deployment as loopclosure module being less sensitive to slight changes in dot product thresholds.

5.1.3. Tripletloss vs allpairloss on decoupled net

We compare the effect of different loss function when using the decoupled network. The network trained with allpair loss is able to correctly identify almost 60% of the pairs from the tuples drawn from the validation data, compared to when network was trained with tripletloss which is able to identify about 35% correctly under identical conditions. See Fig. 8.

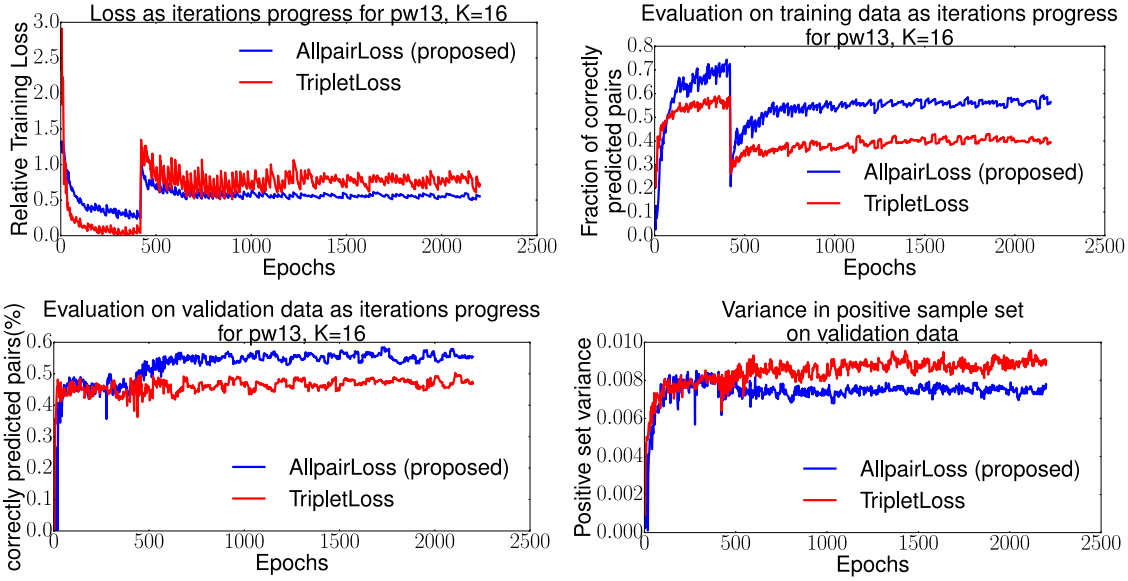


Fig. 8. Comparing the effect of using allpairloss and tripletloss for training with decoupled net (deepest layer) with K = 16. (a) Shows the relative training loss as iterations progress (lower is better). We show the evaluation metric, ie. the count of correctly identified pairs in (b) and (c) for training data and a separate validation data (higher is better). (d) show the variance in the positive set in dot product space as iterations progress (lower is better). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

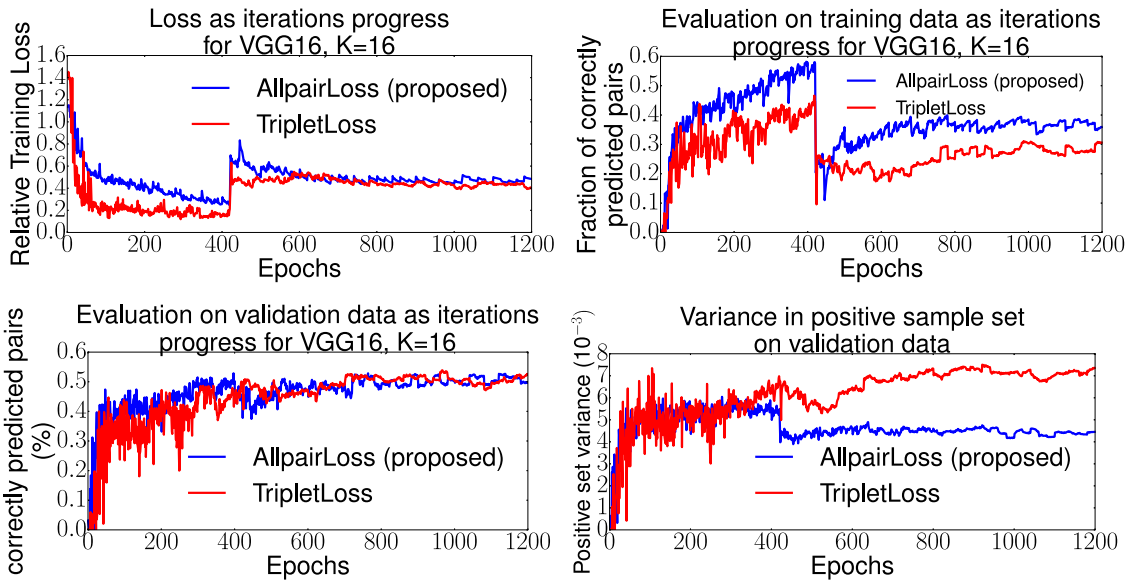


Fig. 9. Effect of tripletloss and allpairloss with backend CNN as the VGG16 net with K = 16. (a) shows the relative training loss as iterations progress (lower is better). (b) and (c) shows the training and validation evaluation metric (higher is better). Evaluation metric is the percentage of pairs correctly identified. (d) shows the variance of positive set descriptors in dot product space.

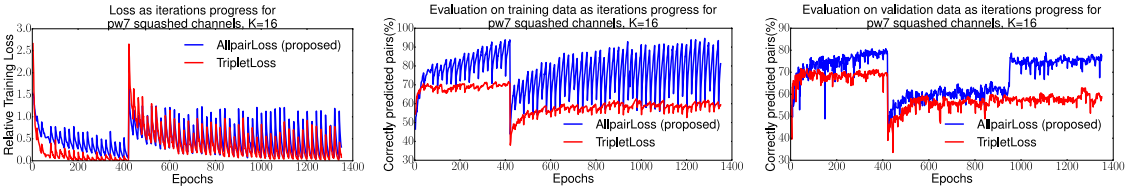


Fig. 10. Effect of tripletloss vs allpairloss for decoupled net, K = 16 with channel squashing. The descriptor size in this case was just 512. Arguably the learning in this case can be improved with lower learning data due to the oscillating losses we observe. (a) shows the relative training loss. (b) and (c) shows the percentage of pairs correctly identified for training and a separate validation data.

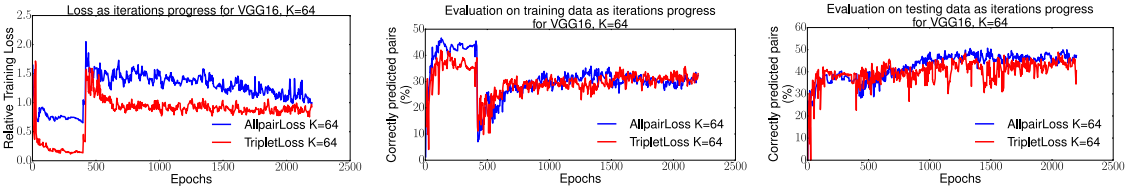


Fig. 11. Effect of tripletloss vs allpairloss for VGG16 net, K = 64. (a) shows the relative training loss (ratio of loss at *i*th and 0th iteration). (b) and (c) shows the percentage of pairs correctly identified for training and a separate validation data.

5.1.4. Tripletloss vs allpairloss on VGG16 net

Even when trained with VGG16 as the backend CNN, allpairloss performed better than the tripletloss under identical training conditions. See Fig. 9.

5.1.5. Tripletloss vs allpairloss on decoupled net with channel squashing

When using decoupled network with channels squashing (for dimensionality reduction) we observe a better performance when trained with allpairloss. In this configuration the descriptor size is just 512 per image. The training was arguably more unstable in this case (we observe oscillations). Possibly with a lower learning rate this effect can be reduced. See Fig. 10 (see Fig. 11).

5.2. Running times

Currently there is rapid progress in compute-capabilities of GPUs. Under such circumstances it is much more appropriate to report the number of floating point operations (FLOPs) for the networks rather than absolute running times in seconds (or milli-seconds). We tabulate in Table 2 the Giga-FLOPs of the networks under various parameter settings. For reference, the forward pass with VGG base network can be computed in about 40–50 ms and with decoupled nets in about 10–15 ms for 640 × 480 3 channel images on Titan X (Pascal). VGG16 with $K = 64$ is the recommended configuration from Arandjelovic et al. [15], which results in a 32K-dimensional descriptor which is reduced to 4096-dimensional by a linear transformation. This linear transformation takes about 400 MB of memory.

On the other hand, our proposed network which uses a network with decoupled convolutions as the base CNN with channel squashing and $K = 16$ results in 512-dimensional descriptor (4096-dimensional if not using channel quashing). It is able to run 3-4x faster than NetVLAD [15] while having about 20x fewer floating point operations for a 640 × 480 image and 5x fewer number of learnable parameters. See Table 2. It is worth noting that most of the computational load is in the computation of per

Table 2
Tabulation of run time memory requirements, learnable parameters (# L), descriptor size (D-Size), model size in Mega-bytes, giga floating point operation (GFLOPs) for various configurations. We note that *block5_pool* for VGG16 network is equal in depth to $pw13$ for decoupled network. *block4_pool* and pw10 have equal depth; *block3_pool* and pw7 have equal depth. K (eg. K16, K64) refers to the number of clusters in NetVLAD layer. We report data for input image size 320 × 240 and 640 × 480. We conclude that our proposed decoupled network is 20X faster computationally with an order of magnitude less number of parameters, while delivering about the same performance as the original NetVLAD. Our squashed channel network 'decoup_K16_r' gives a descriptor size of 512 with about 5% additional forward pass memory and 2% increase in parameter size with hardly noticeable computation time increase. NetVLAD [15] uses a whitening PCA for reducing descriptor dimensionality which needs to store a matrix of size 32Kx4K that takes about 400 MB.

CNN Layer	# L	D-Size	Model (MB)	Fwd pass memory (MB)		GFLOPS	
VGG16_K16				**320 × 240**	**640 × 480**	**320 × 240**	**640 × 480**
block5_pool	14.7M	8192	56.19	234.94	767.56	47.04	188.08
block4_pool	7.6M	8192	29.19	174.06	725.32	47.05	188.117
block3_pool	1.7M	4096	6.65	165.47	641.86	47.05	188.167
VGG16_K64				**320 × 240**	**640 × 480**	**320 × 240**	**640 × 480**
block5_pool	14.78M	32768	56.38	234.32	711.46	47.05	188.11
block4_pool	7.70M	32768	29.38	203.53	696.23	47.08	188.26
block3_pool	1.76M	16384	6.75	158.86	635.26	47.13	188.46
decoup_K16				**320 × 240**	**640 × 480**	**320 × 240**	**640 × 480**
pw13	3.2M	8192	12	197.97	792.21	1.742	7.01
pw10	1.36M	8192	5	189.1	734.48	1.749	7.03
pw7	554K	4096	2	164.9	652.25	1.749	7.04
decoup_K16_r				**320 × 240**	**640 × 480**	**320 × 240**	**640 × 480**
pw13	3.5M	512	12	211.58	793.46	1.742	7.01
pw10	1.49M	512	5	193.86	739.73	1.749	7.03
pw7	686K	512	2	167.67	657.97	1.749	7.04
decoup_K64				**320 × 240**	**640 × 480**	**320 × 240**	**640 × 480**
pw13	3.33M	32768	12	210.98	805.21	1.76	7.08
pw10	1.40M	32768	5	189.38	734.58	1.78	7.18
pw7	600K	16384	2	162.86	652.35	1.78	7.186

pixel descriptors and the NetVLAD layer itself takes negligible computations compared to base CNN.

For training the networks, we use Intel i7-6800 CPU with Titan X (Pascal), 12GB GPU RAM. It takes about 1 s per iteration for forward and backward pass. Note that one iteration with batchsize 4, involve 52 images $((6 + 6 + 1) \times 4)$.

5.3. Precision–recall comparison

We evaluate the performance on the following datasets: (a) **GardensPoint** dataset, (b) **CampusLoop** dataset [8], (c) our **CampusConcourse** dataset. Each of the datasets contains two sequences, 'live' and 'memory'. Note that every image in live sequence has a corresponding image in the memory sequence. For evaluation, we load the memory sequence in the database and compare this database with each of the images in live sequence using a basic nearest neighbor search. Further we also evaluate our performance for the mappilary Berlin streetview dataset [14] (i) **berlin-kundamm** (ii) **berlin-halenseeestrasse** and (iii) **berlin-A100** as has been common amongst visual place recognition community.

We compare the proposed method with the recently proposed learning based loop detection methods **CALC** by Merril and Huang [8]. Additionally we also compare with **DBOW** [5] (Bag-of-visual words). We evaluate with prominent approaches amongst visual place recognition community, **AlexNet** by Sunderhauf et al. [7], **LA-Net**, by Lopez-Antequera et al. [9]; **original NetVLAD** [15]; Chen et al. [18].

The main idea of this evaluation is to gauge the recall rates and discriminative performance of various methods. We take the matches as correctly identified if match's index is within six indices of itself. For our evaluation, we use the total number of positive matches as the length of the sequence, since every image in the live sequence has a correspondence in the memory-sequence. Total number of accepted matches are those which satisfy the loop hypothesis. The precision–recall curves are formed by sweeping through all the thresholds within the full range of thresholds. The results presented here differ from those presented in [8] as it is not exactly clear how recall = 1 was achieved by them, how the DBoW was used to generate these results and any heuristics, if any, was used to identify false positives.

As noted by Merril and Huang [8], and by Sunderhauf et al. [66] superior precision–recall does not fully prove the superiority of a method in real loop-closure of a SLAM system. Such factors as repeated objects in scenes, similar looking scenes, invariance to rotation & scale, computation time are important when considering a place recognition system for SLAM's loop closure. Another issue about such an evaluation is that it cannot gauge a method's ability to return 'no matches' in case the query scene is not found in the database. Also all these datasets are rather small (about 80–200 frames) and we cannot evaluate the generalizability of the scene description by each of the methods. For example a dataset with multiple similar looking scene is needed to thoroughly evaluate a method's performance. Rotation and scale variance of the method cannot be evaluated with these datasets. So for a better perspective of the usability of the methods we also evaluate them on live SLAM sequences with manually marked loopclosure detections for evaluation (details in Section 5.4).

5.3.1. Walking-apart sequence

For precision–recall curves see Fig. 12. The *CampusLoop sequence* contains appearance variation due to changing weather condition. Our method does not explicitly deal with this kind of variation as it is primarily based on color cues. The method CALC, for example, is based on scene structure. Our method delivers comparable performance in this sequence. For the other two testing sequences, viz. *GardenPoint* and *UniConcourse*, our method performs better than previous methods. This is attributed to the fact that the descriptors learned by our method are able to generalize well into identifying place revisits at large viewpoint difference and rotational variance, which is the case in these two sequences. When compared to NetVLAD [15] which uses the triplet loss and VGG network, we observe a slight boost of recall rates. Since these sequences are very small, the higher capacity of the proposed method is not observable in this case.

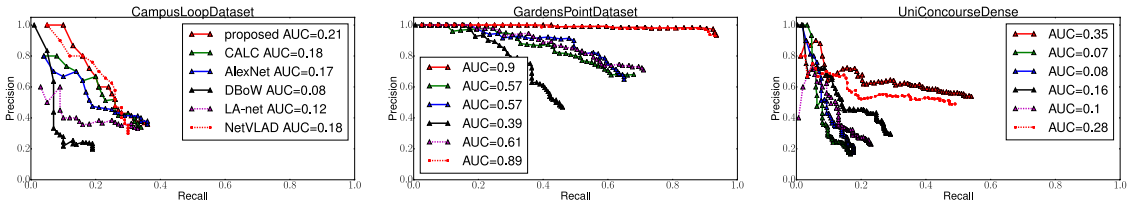


Fig. 12. Precision–recall curves for various methods for loop detection. Our method using the decoupled net as the backend CNN gives comparable performance in CampusLoop dataset which contains appearance changes due to snowy weather. Our method gives a comparable performance to the NetVLAD in other two datasets which has only large viewpoint and in-plane rotational changes. Which is far better than other relevant methods. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

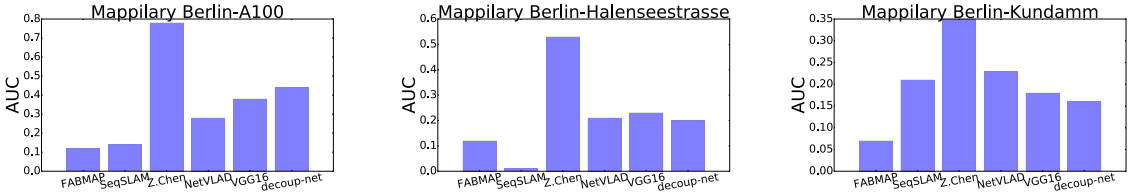


Fig. 13. Comparing the methods with area under the curve (AUC) of the precision–recall plots for the mappilary dataset. The following methods were compared: FABMAP [13], SeqSLAM [11], Z. Chen [18], NetVLAD [15], proposed with VGG16 backend net, proposed with decoupled net as backend net.

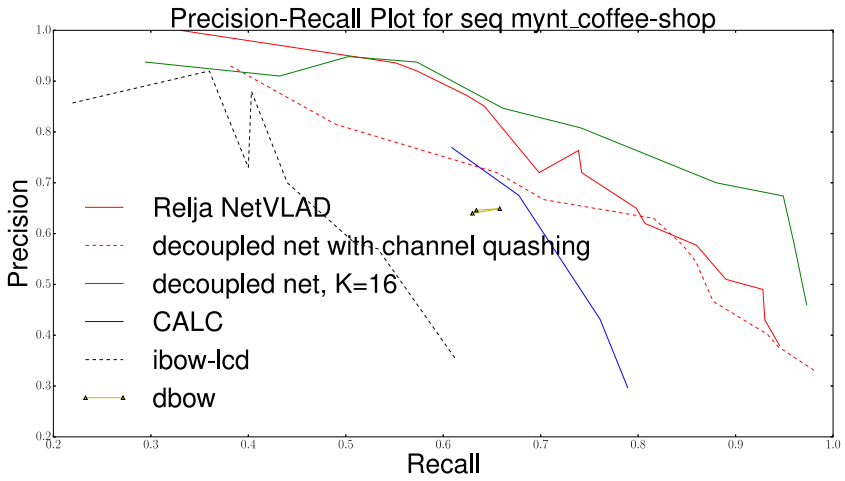


Fig. 14. Precision–recall plot for the sequence 'mynt_coffee-shop' when compared to manual annotations of loop candidates and threshold varied. We compare the following methods: Relja NetVLAD [15], decoupled net with channel squashing (proposed), decoupled net without channel squashing, CALC [8], ibow-lcd [6] and DBOW [5].

5.3.2. AUC performance on mappilary dataset

We evaluate our method with other state of the art methods with area-under-the-curve (AUC) of the precision–recall plot on the mappilary berlin-streets dataset in Fig. 13. Each of the three test sequences contain two sets of images. Note that each image in second of the two sets has a pre-image in the first set. These datasets are 80–200 frames each. Although considerable viewpoint and light variation exists amongst the two sets, there is no rotation variation. We test our method with VGG16 backend CNN and with decoupled-net backend CNN. We compare with FAB-MAP [13], SEQSLAM [11], Chen et al. [18], NetVLAD [15]. In this case, Chen et al.'s method outperform. It is worth noting that Chen's method [18] makes use of region proposal and is not a real-time (or near real-time) method.

5.4. Online loop detections

We compare the performance of the descriptors produced from the proposed method to some of the relevant methods with real world sequences. We introduce three sequences and refer them as 'Live Walks Dataset', each is about 10min of walking. The main differentiating point compared to the standard KITTI dataset is that ours contains adversaries like revisits under large viewpoint difference, moving objects (people), noise, lighting changes, in-plane rotation to name a few. Two of which were captured with a gray scale camera and one of it was captured with a color camera. We also provide manually marked ground truth labels for loop detections along with odometry of the poses for visualization. The odometry was not used for identifying loops. Note that for a real sequence with N keyframes there are a little less than N^2 pair of loop-frames. The human was shown every pair and asked to mark the pairs which were the same place. Using these manual annotations, there are 3 kinds of pairs. (a) pairs not detected by the algorithm, ie. missed pairs (b) wrongly detected pairs, ie. pairs which were in reality different places but algorithm identified it as the same place. (c) pairs correctly identified, ie. pairs which were marked by the algorithm as same places and were in reality same places.

We compare our method with some of the relevant methods on our real world datasets. We plot the precision–recall under various threshold settings. We define precision as the fraction of candidate loops which were actually loops. By recall we mean the

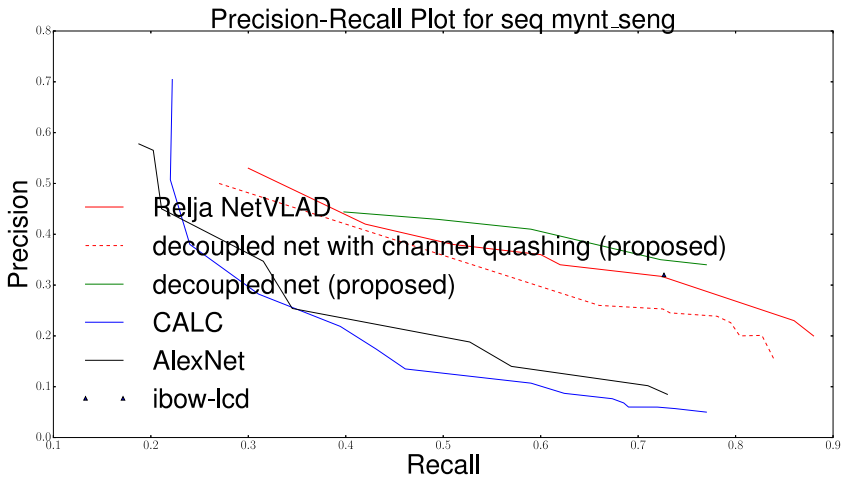


Fig. 15. Similar to Fig. 14 but for sequence 'mynt_seng'.

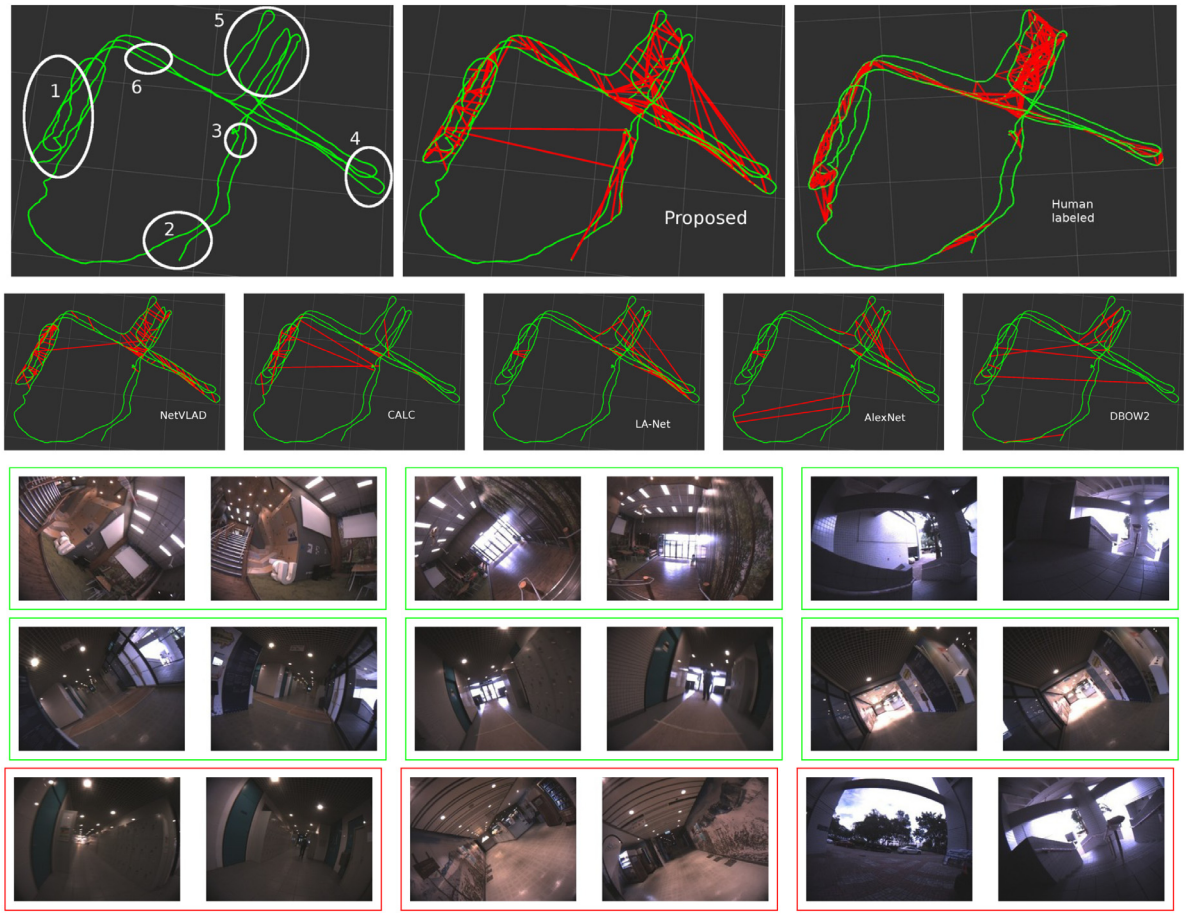


Fig. 16. Top row: Plot of visual-inertial odometry of the sequence 'base-2'; Loop candidates by our proposed method; human marked loop candidates; **2nd row**: NetVLAD [15]; CALC [8]; LA-Net [9]; Alexnet [14]; DBOW2 [5]. **Row 3 and 4**: Examples of correct detections by the proposed method in each of the regions. **Row 5**: Examples of wrong detections.

fraction of actual loops identified. We do not use any geometric verification step to boost our precision, the results shown in this section are from raw image descriptor comparison. With geometric verification, precision of almost 100% can be easily accomplished.

We use our method in various configurations (a) decoupled net as base CNN, K = 16 (descriptor size of 4096), (b) VGG16 as base CNN, K = 16, (c) decoupled net with squashed channels, K = 16 (descriptor size of 512). We compare with (i) NetVLAD [15], (ii) Merril and Huang [8], (iii) Sunderhauf et al. [7], (iv) DBOW [5] and (v) ibow-lcd [6]. We acknowledge the superior performance of Z. Chen et al. [18] method and possibly also of Sunderhauf et al. [14] on the mappilary dataset. However, it was not practical to test it on our datasets which are an order of magnitude larger than those dataset. It takes about 1–1.5 sec/frame for descriptor computation and about 800 ms–1.2 sec/pair for descriptor comparison. For our dataset of 5000 keyframes the provided MATLAB implementation would take almost 100 days (number of comparisons would be

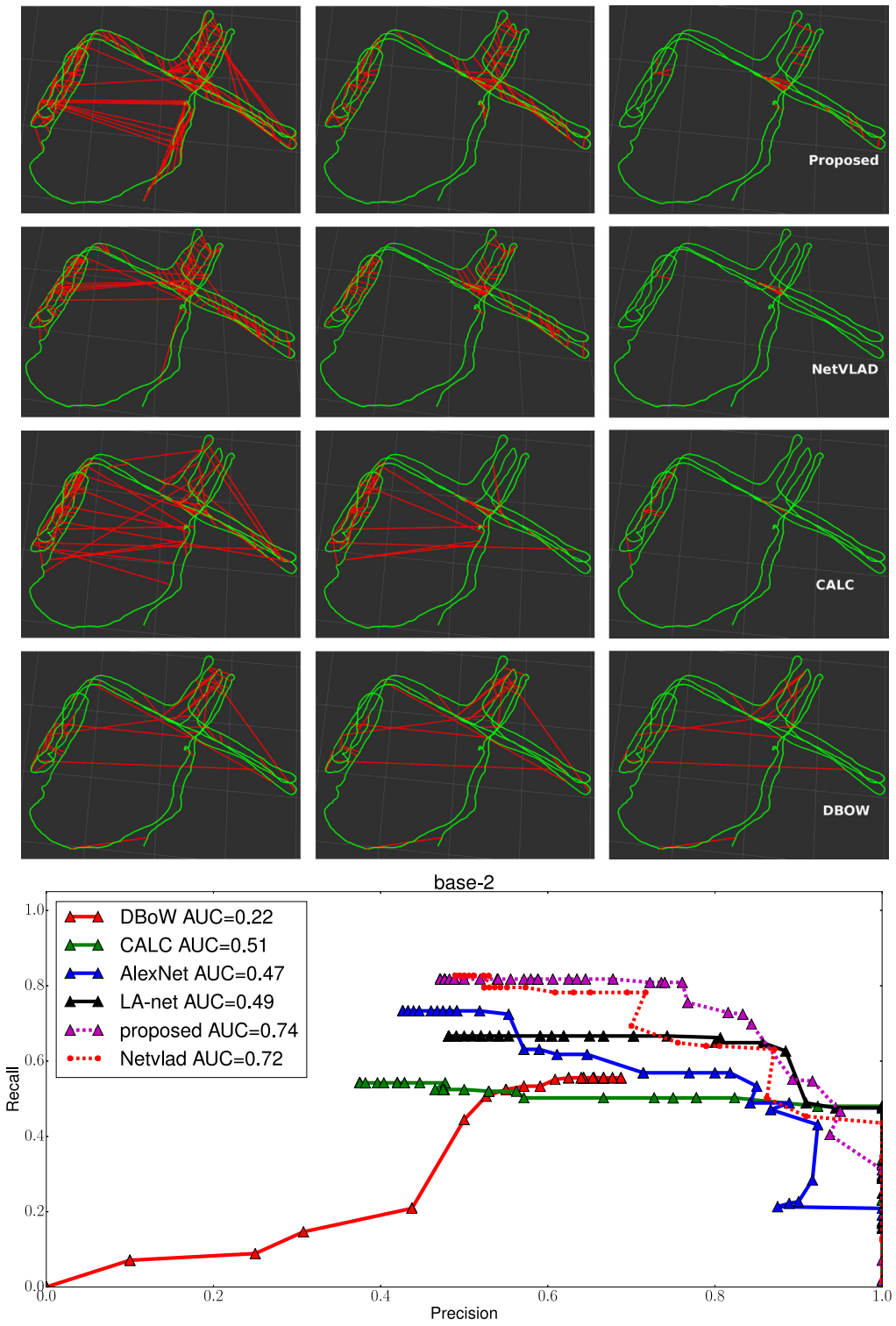


Fig. 17. Loop closure candidates (in red) as we vary the thresholds on VIO (green) for sequence 'base-2' for the proposed method (in row-1); NetVLAD [15] (in row-2); CALC [8] (in row-3) and DBOW [5] (in row-4). Along the columns are various thresholds. Leftmost is for loosest, rightmost is for tightest. Row-5 shows the PR-curve for each method where compared to human marked loop-candidates. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$5000 + 4999 + 4998 + \cdots$). Arguably a faster implementation could accomplish the task in about a day or two for a 5000 frame or 15 min walking video. Thus this method is no where close to being real-time. Hence it was not compared. We also note that the running time for Sunderhauf et al. [14] is in similar range to Z. Chen et al.'s method.

The proposed method is able to detect revisits in all the regions for this test sequence. We attribute this to the NetVLAD

architecture which cumulates the descriptors so as to become somewhat invariant to in-plane rotations. Other learning based methods for example CALC totally miss the revisits occurring under dim to moderate lighting. This can be attributed to the fact that it is based on HOG which under dim lighting do not provide enough spread for the histogram to generate meaningful descriptors. CALC and DBOW as expected are able to work in situations with in-plane rotations (in region 1 of Fig. 16). AlexNet and LA-Net being essentially off-the-shelf parameters learned for object classification task are not invariant to rotations. DBOW however is easily confused if two scene share similar looking textures, for example, the texture of the ceiling in otherwise different looking scenes. DBOW also perform poorly under low-contrast scenes. Compared to the original NetVLAD descriptor we observe a boost in recall rates for the proposed method. A precision–recall curve when comparing the methods to human marked loop candidates is presented in Fig. 17.

5.5. Full system experiments

We also experiment with our entire system involving relative pose computations at the loopcandidates and pose graph solver with kidnap recovery mechanism. Our experimental setup involves just the 'MYNT EYE D'[6] camera. It includes a stereo camera pair and an 200 Hz IMU with frame and IMU sync of about 1 ms. We kidnap the camera by blocking the view of the camera and transporting it to another location. Additionally, we also experiment with the EuRoC MAV dataset [67] which also have stereo camera data and IMU (see Fig. 18).

A representative live real-time run of the system is shown in Fig. 1. Our system can identify and recover from kidnaps online and in realtime. A comparison of the loop detections with the VINSFusion system and our proposed system is shown in Fig. 19. This sequence repeatedly traverse a hall at non-fronto-parallel views and with in-plane rotations. Our system is able to correctly recognize and compute relative poses at loopclosures involving large viewpoint differences and in-plane rotations.

We highlight the distinguishing points of our system compared to *colmap* [68] and *maplab* [4]. *Colmap* is a general 3D reconstruction system and involves offline processing of unordered image sets. The *maplab* system provides an online tool, *ROVIOLI* which is essentially a visual-inertial odometry and localization front-end. Although it provides a console based interface for multi-session map merging, it cannot identify kidnaps and recover from them online. Our system essentially fills in this gap.

6. Conclusion and future work

We proposed a data-driven, weakly supervised approach to learn a scene representation for use in loopclosure module of a SLAM system. Additionally we demonstrated the use of the disjoint dataset structure to maintain set associations of multiple coordinate systems for online merging of multiple pose graphs.

Unstable learning was observed for the original NetVLAD [15] which made use of tripletloss for training. This was observed to be especially prominent when trained with smaller number of clusters. The issue was mitigated with use of the proposed allpairloss function. This resulted in higher performance even with a smaller number of cluster in the NetVLAD layer. For realtime performance we made use the decoupled convolutional layer instead of the standard convolutions for speed. The network with decoupled convolutions are almost 3X faster in computation time with 5-7X fewer learnable parameters.

[6] https://www.mynteye.com

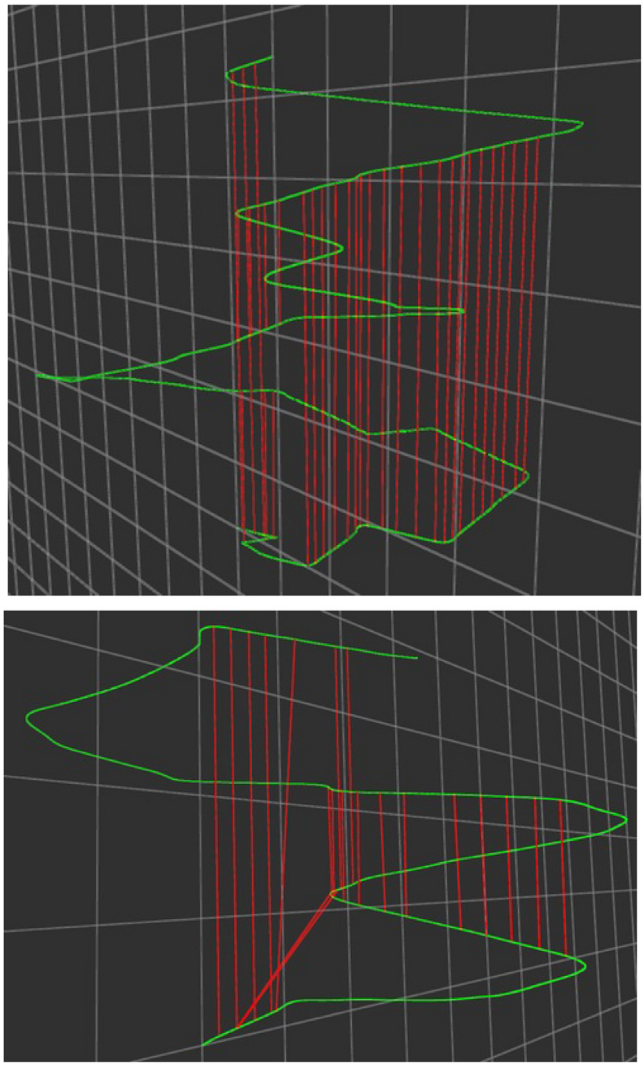


Fig. 18. The results of the proposed method on KITTI00 and KITTI05. The XY plane is the 2d location of the trajectory. z-axis represents the frame number. In this dataset the revisits occur at similar viewpoints, the performance of all the compared methods is almost the same.

To evaluate precision–recall performance for loopclosure detection in a real SLAM system, we compare our method with popular BOVW-based methods along with state-of-the-art CNN-based methods on real world sequences. Qualitative and quantitative experiments on standard datasets as well as self-captured challenging sequences with adversaries including revisits at large viewpoint difference, in-plane rotation, dim lighting etc. suggest that proposed method can identify loopcandidates under substantial viewpoint difference. We also observe a boost in recall rates when compared to training with original NetVLAD. Also our descriptors are found to be fairly invariant to rotation and lighting changes.

In addition to the precision–recall we also demonstrate the real-time working of our method as a pluggable module for VINS-Fusion. Our system is not only able to reduce drift but also identify and recover from complicated kidnap scenarios and failures.

A robust place recognition module is a critical element for the SLAM dependent fields: long-term autonomy and augmented reality. In addition to a robust and discriminative image representation, the use of text and object information to further

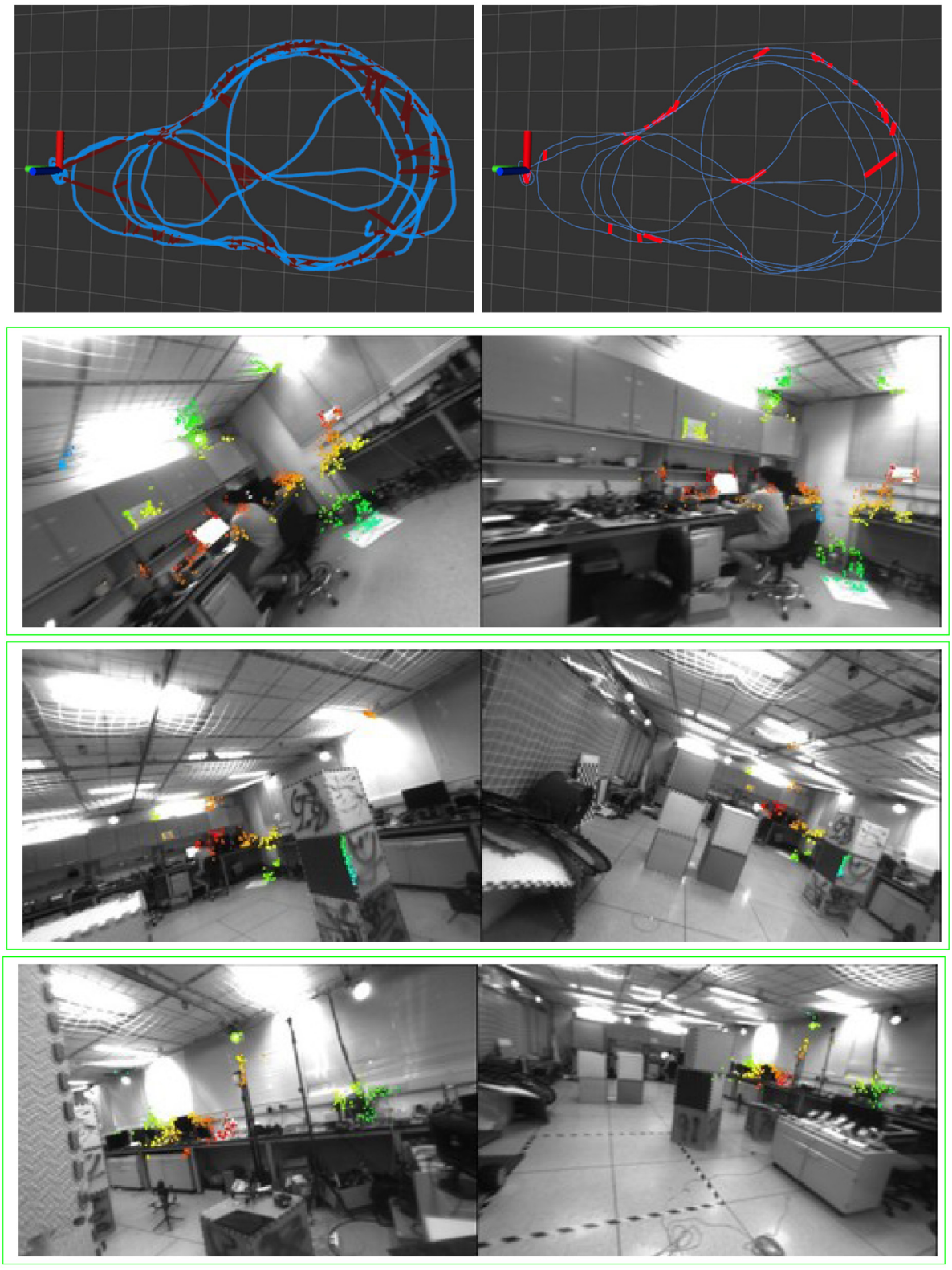


Fig. 19. Comparing revisit detections of the proposed method (top) and VINS-Fusion, which uses DBOW2 (2nd row). This sequence contains repeated traversal in a hall of 15mx5m at various rotations and viewpoints. Although bag-og-words based method perform well under fronto-parallel view it has very low recall compared to our method on larger viewpoint difference. A side-by-side live run of this sequence is available at https://youtu.be/dbzN4mKeNTQ. Row-3 to row-5 shows some representative looppairs which we identified by our methods as loops but were missed out by DBOW2 in VINS-Fusion.

disambiguate similar looking places and provide semantic cues to underlying planning methods could be a way forward for a truly intelligent and scalable place recognition system. To aid the development of such a system, we opensource our implementation and our dataset along with human annotated looppairs to the research community.

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.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.robot.2021.103813.

References

[1] 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.

[2] 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.

[3] T. Qin, P. Li, S. Shen, Relocalization, global optimization and map merging for monocular visual-inertial SLAM, in: 2018 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2018, pp. 1197–1204.

[4] T. Schneider, M.T. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, R. Siegwart, maplab: An open framework for research in visual-inertial mapping and localization, IEEE Robot. Autom. Lett. (2018) http://dx.doi.org/10.1109/LRA.2018.2800113.

[5] 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.

[6] E. Garcia-Fidalgo, A. Ortiz, iBoW-LCD: An appearance-based loop closure detection approach using incremental bags of binary words, 2018, arXiv preprint arXiv:1802.05909.

[7] N. Sünderhauf, S. Shirazi, F. Dayoub, B. Upcroft, M. Milford, On the performance of convnet features for place recognition, in: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, IEEE, 2015, pp. 4297–4304.

[8] N. Merrill, G. Huang, Lightweight unsupervised deep loop closure, 2018, arXiv preprint arXiv:1805.07703.

[9] M. Lopez-Antequera, R. Gomez-Ojeda, N. Petkov, J. Gonzalez-Jimenez, Appearance-invariant place recognition by discriminatively training a convolutional neural network, Pattern Recognit. Lett. 92 (2017) 89–95, http://dx.doi.org/10.1016/j.patrec.2017.04.017.

[10] T. Qin, P. Li, S. Shen, VINS-mono: A robust and versatile monocular visual-inertial state estimator, IEEE Trans. Robot. 34 (4) (2018) 1004–1020.

[11] M.J. Milford, G.F. Wyeth, SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights, in: Robotics and Automation (ICRA), 2012 IEEE International Conference on, IEEE, 2012, pp. 1643–1649.

[12] M. Cummins, P. Newman, FAB-MAP: Probabilistic localization and mapping in the space of appearance, Int. J. Robot. Res. 27 (6) (2008) 647–665.

[13] M. Cummins, P. Newman, Appearance-only SLAM at large scale with FAB-MAP 2.0, Int. J. Robot. Res. 30 (9) (2011) 1100–1123.

[14] 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.

[15] 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.

[16] 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.

[17] Z. Chen, L. Liu, I. Sa, Z. Ge, M. Chli, Learning context flexible attention model for long-term visual place recognition, IEEE Robot. Autom. Lett. 3 (4) (2018) 4015–4022.

[18] 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.

[19] Y. Hou, H. Zhang, S. Zhou, BoCNF: Efficient image matching with bag of ConvNet features for scalable and robust visual place recognition, Auton. Robots 42 (6) (2018) 1169–1185.

[20] D. Bai, C. Wang, B. Zhang, X. Yi, X. Yang, Sequence searching with cnn features for robust and fast visual place recognition, Comput. Graph. 70 (2018) 270–280.

[21] X. Gao, T. Zhang, Unsupervised learning to detect loops using deep neural networks for visual SLAM system, Auton. Robots 41 (1) (2017) 1–18.

[22] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, T. Pajdla, 24/7 place recognition by view synthesis, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2015, pp. 1808–1817, http://dx.doi.org/10.1109/CVPR.2015.7298790.

[23] 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: 2017 IEEE International Conference on Robotics and Automation, ICRA, 2017, pp. 3223–3230, http://dx.doi.org/10.1109/ICRA.2017.7989366.

[24] A. Loquercio, M. Dymczyk, B. Zeisl, S. Lynen, I. Gilitschenski, R. Siegwart, Efficient descriptor learning for large scale localization, in: 2017 IEEE International Conference on Robotics and Automation, ICRA, 2017, pp. 3170–3177, http://dx.doi.org/10.1109/ICRA.2017.7989359.

[25] 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.

[26] R. Arandjelović, A. Zisserman, DisLocation: Scalable descriptor distinctiveness for location recognition, in: Asian Conference on Computer Vision, Springer, 2014, pp. 188–204.

[27] 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.

[28] L. Zheng, Y. Yang, Q. Tian, SIFT meets CNN: A decade survey of instance retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 40 (5) (2018) 1224–1244.

[29] R. Mur-Artal, J.D. Tardós, Fast relocalisation and loop closing in keyframe-based SLAM, in: Robotics and Automation (ICRA), 2014 IEEE International Conference on, IEEE, 2014, pp. 846–853.

[30] L. Bampis, A. Amanatiadis, A. Gasteratos, High order visual words for structure-aware and viewpoint-invariant loop closure detection, in: Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on, IEEE, 2017, pp. 4268–4275.

[31] A. Angeli, D. Filliat, S. Doncieux, J.-A. Meyer, Fast and incremental method for loop-closure detection using bags of visual words, IEEE Trans. Robot. 24 (5) (2008) 1027–1037.

[32] T. Nicosevici, R. Garcia, Automatic visual bag-of-words for online robot navigation and mapping, IEEE Trans. Robot. 28 (4) (2012) 886–898.

[33] M. Labbe, F. Michaud, Appearance-based loop closure detection for online large-scale and long-term operation, IEEE Trans. Robot. 29 (3) (2013) 734–745.

[34] K.A. Tsintotas, L. Bampis, A. Gasteratos, Assigning visual words to places for loop closure detection, in: 2018 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2018, pp. 1–7.

[35] L. Bampis, A. Amanatiadis, A. Gasteratos, Fast loop-closure detection using visual-word-vectors from image sequences, Int. J. Robot. Res. 37 (1) (2018) 62–82.

[36] E. Garcia-Fidalgo, A. Ortiz, On the use of binary feature descriptors for loop closure detection, in: Emerging Technology and Factory Automation (ETFA), 2014 IEEE, IEEE, 2014, pp. 1–8.

[37] G. Zhang, M.J. Lilly, P.A. Vela, Learning binary features online from motion dynamics for incremental loop-closure detection and place recognition, in: Robotics and Automation (ICRA), 2016 IEEE International Conference on, IEEE, 2016, pp. 765–772.

[38] S. Khan, D. Wollherr, Ibuild: Incremental bag of binary words for appearance based loop closure detection, in: Robotics and Automation (ICRA), 2015 IEEE International Conference on, IEEE, 2015, pp. 5441–5447.

[39] E. Stumm, C. Mei, S. Lacroix, J. Nieto, M. Hutter, R. Siegwart, Robust visual place recognition with graph kernels, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4535–4544.

[40] R. Arroyo, P.F. Alcantarilla, L.M. Bergasa, E. Romera, Fusion and binarization of CNN features for robust topological localization across seasons, in: Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on, IEEE, 2016, pp. 4656–4663.

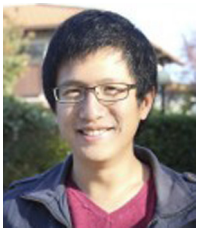
[41] E. Pepperell, P.I. Corke, M.J. Milford, All-environment visual place recognition with SMART, in: 2014 IEEE International Conference on Robotics and Automation, ICRA, 2014, pp. 1612–1618, http://dx.doi.org/10.1109/ICRA.2014.6907067.

[42] H. Zhang, F. Han, H. Wang, Robust multimodal sequence-based loop closure detection via structured sparsity, in: Robotics: Science and Systems, 2016.

[43] Y. Latif, G. Huang, J.J. Leonard, J. Neira, An online sparsity-cognizant loop-closure algorithm for visual navigation, in: Robotics: Science and Systems, 2014.

[44] F. Han, H. Wang, G. Huang, H. Zhang, Sequence-based sparse optimization methods for long-term loop closure detection in visual SLAM, Auton. Robots (2018) 1–13.
[45] E.S. Stumm, C. Mei, S. Lacroix, Building location models for visual place recognition, Int. J. Robot. Res. 35 (4) (2016) 334–356, http://dx.doi.org/10.1177/0278364915570140.
[46] C. Kenshimov, L. Bampis, B. Amirgaliyev, M. Arslanov, A. Gasteratos, Deep learning features exception for cross-season visual place recognition, Pattern Recognit. Lett. 100 (2017) 124–130.
[47] X. Fei, K. Tsotsos, S. Soatto, A simple hierarchical pooling data structure for loop closure, 2015, arXiv:1511.06489, CoRR abs/1511.06489, URL http://arxiv.org/abs/1511.06489.
[48] E. Sizikova, V.K. Singh, B. Georgescu, M. Halber, K. Ma, T. Chen, Enhancing place recognition using joint intensity - depth analysis and synthetic data, in: European Conference on Computer Vision (ECCV) Workshop on Virtual/Augmented Reality for Visual Artificial Intelligence, VARVAI, 2016.
[49] T. Cieslewski, D. Scaramuzza, Efficient decentralized visual place recognition using a distributed inverted index, IEEE Robot. Autom. Lett. 2 (2) (2017) 640–647, http://dx.doi.org/10.1109/LRA.2017.2650153.
[50] T. Cieslewski, D. Scaramuzza, Efficient decentralized visual place recognition from full-image descriptors, in: 2017 International Symposium on Multi-Robot and Multi-Agent Systems, MRS, 2017, pp. 78–82, http://dx.doi.org/10.1109/MRS.2017.8250934.
[51] H. Jégou, M. Douze, C. Schmid, On the burstiness of visual elements, in: Computer Vision and Pattern Recognition, 2009. IEEE Conference on, CVPR 2009, IEEE, 2009, pp. 1169–1176.
[52] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, arXiv preprint arXiv:1409.1556.
[53] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017, arXiv preprint arXiv:1704.04861.
[54] Y. Bengio, J. Louradour, R. Collobert, J. Weston, Curriculum learning, in: Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 2009, pp. 41–48.
[55] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: A unified embedding for face recognition and clustering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 815–823.
[56] A. Torii, J. Sivic, T. Pajdla, M. Okutomi, Visual place recognition with repetitive structures, in: Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, IEEE, 2013, pp. 883–890.
[57] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, 2010, pp. 249–256.
[58] M.D. Zeiler, ADADELTA: an adaptive learning rate method, 2012, arXiv preprint arXiv:1212.5701.
[59] J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with GPUs, 2017, arXiv preprint arXiv:1702.08734.
[60] J. Bian, W.-Y. Lin, Y. Matsushita, S.-K. Yeung, T.-D. Nguyen, M.-M. Cheng, Gms: Grid-based motion statistics for fast, ultra-robust feature correspondence, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, 2017, pp. 2828–2837.
[61] 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.
[62] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, J. Sivic, Neighbourhood consensus networks, in: Advances in Neural Information Processing Systems, 2018, pp. 1651–1662.
[63] J.A. Hesch, S.I. Roumeliotis, A direct least-squares (DLS) method for PnP, in: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE, 2011, pp. 383–390.
[64] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, MIT Press, 2009.
[65] N. Sünderhauf, P. Protzel, Switchable constraints for robust pose graph SLAM, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2012, pp. 1879–1884.
[66] N. Sünderhauf, O. Brock, W. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford, et al., The limits and potentials of deep learning for robotics, Int. J. Robot. Res. 37 (4–5) (2018) 405–420.
[67] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M.W. Achtelik, R. Siegwart, The EuRoC micro aerial vehicle datasets, Int. J. Robot. Res. (2016) http://dx.doi.org/10.1177/0278364915620033, arXiv:http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.full.pdf+html, URL http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.abstract.
[68] J.L. Schönberger, J.-M. Frahm, Structure-from-motion revisited, in: Conference on Computer Vision and Pattern Recognition, CVPR, 2016.

Manohar Kuse is from Mumbai, India. He is currently a Ph.D. candidate at Robotics Institute (RI) of Hong Kong University of Science and Technology (HKUST) in Hong Kong. His research focus on Visual Navigation of UAVs. Before joining his Ph.D. studies he was a Research Assistant at European Organization for Nuclear Research (CERN) in Geneva, Switzerland.



Shaojie Shen received his B.Eng. degree in Electronic Engineering (Honors Research Option) from the Hong Kong University of Science and Technology in 2009. He received his M.S. in Robotics and Ph.D. in Electrical and Systems Engineering in 2011 and 2014, respectively, all from the University of Pennsylvania. He joined the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology in September 2014 as an Assistant Professor. His research interests are in the areas of robotics and unmanned aerial vehicles, with focus on state estimation, sensor fusion, localization and mapping, and autonomous navigation in complex environments.