Evaluation of Long-term LiDAR Place Recognition

Jukka Peltomäki⋆, Farid Alijani⋆, Jussi Puura†, Heikki Huttunen⋆, Esa Rahtu⋆, and Joni-Kristian Kämäräinen⋆

***Abstract*— We compare a state-of-the-art deep image retrieval and a deep place recognition method for place recognition using LiDAR data. Place recognition aims to detect previously visited locations and thus provides an important tool for navigation, mapping, and localisation. Experimental comparisons are conducted using challenging outdoor and indoor datasets, Oxford RobotCar and COLD, in the "long-term" setting where the test conditions differ substantially from the training and gallery data. Based on our results the image retrieval methods using LiDAR depth images can achieve accurate localization (the single best match recall 80%) within 5.00 m in urban outdoors. In office indoors the comparable accuracy is 50 cm but is more sensitive to changes in the environment.**

I. Introduction

An autonomous robot that operates in an environment should be able to recognize different places when it revisits them after some time (Figure 1, top). This is important to support reliable navigation, mapping, and localisation. Robust place recognition is therefore a crucial capability for an autonomous robot. Due to its importance place recognition is an important research topic in robotics and computer vision community for which Lowry et al. [1] and Zhang et al. [2] survey the past works.

The problem of visual place recognition gets challenging if the visual appearance of places change over time. This usually happens due to changes in the lighting conditions, shadows, different weather conditions, or even different seasons. Also, people moving around and items being moved around change the environment. These factors are particularly addressed in *long-term visual place recognition*. Engineered features can be adjusted to be invariant [3], [4], but the recent deep learning methods are prone to overfitting and therefore their performance depends on suitability of the selected training data [5], [6]. These works lack in one or more terms: they focus either indoor or outdoor place recognition, limited variability, unrealistic navigation data or focus on RGB images only.

In this paper, we study deep place recognition using LiDAR sensor. Compared to typical RGB camera, LiDAR is less rich in details but more robust to various sources of imaging distortions such as illumination change and weather conditions. We perform extensive experiments on two largest indoor and outdoor datasets that include LiDAR measurements and are suitable for robot navigation: Oxford Radar RobotCar [7][8] and COLD [9]. Our findings are that LiDAR is competitive modality to RGB and especially the range (depth map) measurements are robust to long-term changes. We also show that the state-of-the-art image retrieval method by Radenovic et al. [10] performs well in place recognition even without data specific fine-tuning and is not sensitive to backbone network selection. We publish all code and data to facilitate fair comparisons and future works on place recognition for robot navigation.

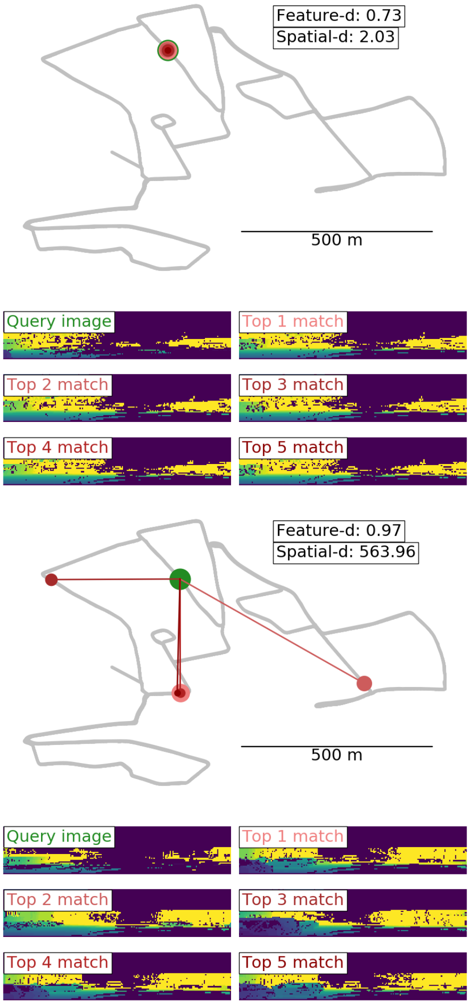

Fig. 1. Examples of successful (top) and failed place recognition (bottom) in the RobotCar dataset. In successful recognition the best five matches (red points) are all near the query place (green point) and within a small feature distance while in the failed case the feature distances are larger and matches are found in random spatial locations.

Related work – Surveys for RGB-based methods can be found from Lowry et al. [1] (engineered features) and

⋆Tampere University, Finland
†Sandvik Mining and Construction Ltd

Zhang et al. [2] (deep learning). Visual localization combines place recognition and refined localization (inc. 3D pose) using RGB images. Hierarchical Localization using SuperPoint and SuperGlue [11], [12] is the top performing system for visual localization in the recent benchmark by Pion et al. [13].

A number of methods have been proposed for point cloud and LiDAR based place recognition. For example, PointNetVLAD [14] uses global features of point cloud data for place recognition. Steder et al. [15] use LiDAR range images. Guo et al. [16] combine the both LiDAR range (depth) and intensity values and use a probabilistic voting scheme. A network architecture utilizing the combination of RGB camera and LiDAR point clouds for place recognition was introduced by Xie et al. [17]. A place recognition system featuring adversarial training and octree mapping was introduced by Yin et al. [18]. Kim and Kim [19] introduced scan contexts, a type of spatial descriptor, to improve results with point cloud place recognition. Our work focus on LiDAR only place recognition to analyze whether LiDAR depth or intensity images work well in long-term place recognition.

A number datasets suitable for place recognition are publicly available: MulRan [20], The Newer College Dataset [21], COLD [9], NCLR [22], Mapillary Street-Level Sequences [23], and Oxford Radar RobotCar [7]. For our experiments we selected COLD and Oxford Radar RobotCar as they represent realistic navigation sequence, include LiDAR and are large long-term datasets.

II. Methods

The two methods experimented in our work are NetVLAD by Arandjelovic et al. [24] and CNN retrieval (CNNRetr) by Radenovic et al. [25]. NetVLAD [24] was selected as it is used in *Hierarchical Localization using SuperPoint and SuperGlue* [11], [12] that won the 2020 Visual Localization Challenge (`https://www.visuallocalization.net/`). On the other hand, the CNNRetr is at the core of the state-of-the-art image retrieval architecture of Radenovic et al. [10], [25], [26]. In the following we briefly introduce these methods and their adaptation to place recognition.

A. Deep place recognition (NetVLAD)

The core idea of NetVLAD [24] is in deep metric learning where the deep architecture learns to produce a representation that encodes the informative content of inputs. The representation is metric in the sense that similarity of inputs, such as RGB or LiDAR range images, can be measured by standard distance functions such as Euclidean distance. In other words, the objective is to learn a function f_θ with its parameters (network weights) defined by θ that maps images I_i to a high (D-)dimensional feature vector space $f_\theta : I \rightarrow \mathbb{R}^D$. The high dimensional representation encodes images from the same place with a small distance value $d_\theta(I_i, I_j) = ||f_\theta(I_i) - f_\theta(I_j)||$ and images from different places with large distance values.

The main building blocks of NetVLAD are 1) the *NetVLAD layer* that implements a differentiable version of the VLAD encoding of SIFT features [27] to replace maximum pooling, 2) *triplet ranking loss*, 3) *Principal Component Analysis (PCA) based dimensionality reduction* and 4) *training procedure using Google Street View Time Machine dataset* that provides multiple close-by images of the same spatial locations captured at different times.

The original VLAD representation is a $K \times D$-dimensional matrix where K denotes cluster centers (visual words) and D is the number of feature dimensions. The SIFT detector provides N descriptors that are VLAD encoded using the following formula:

$$V(j,k) = \sum_{i=1}^{N} a_k(\mathbf{x}_i)(x_i(j) - c_k(j)), \tag{1}$$

where $x_i(j)$ and $c_k(j)$ are the j-th dimensions of the i-th descriptor and k-th cluster centers. The $a_k(\mathbf{x}_i)$ means that the descriptor x_i belongs to the k-th visual word. In other words, $\mathbf{V}$ encodes feature distances from the visual words that are obtained by clustering all features in the training set. This encoding is more powerful than the original Bag-of-Words (BoW) encoding [27], but with the price of much larger feature vectors ($\mathbf{V}$ can be converted to a vector). NetVLAD layer uses a differentiable version of (1)

$$V(j,k) = \sum_{i=1}^{N} \frac{e^{\mathbf{w}_k^T \mathbf{x}_i + b_k}}{\sum_{k'} e^{\mathbf{w}_{k'}^T \mathbf{x}_i + b_{k'}}}(x_i(j) - c_k(j)), \tag{2}$$

where $\{\mathbf{w}\}_k$, $\{b\}_k$ and $\{\mathbf{c}\}_k$ are sets of parameters optimized during training. Eq. 2 is obtained from (1) by applying soft-assignment instead of the original hard assignments. The downside of the NetVLAD features is their high dimensionality and therefore Arandjelovic et al. [24] propose a PCA-based dimensionality reduction as a post-processing step. However, in our experiment we found it unnecessary and therefore used the full NetVLAD feature vectors which for 512-dim deep features and 64 clusters have $512 \times 64 = 32,768$ elements.

The typical formulation of the triplet loss [28] is

$$\max\left(||f_\theta(I_A) - f_\theta(I_P)||^2 - ||f_\theta(I_A) - f_\theta(I_N)||^2 + \alpha, 0\right), \tag{3}$$

where I_A is the "anchor image" (query image from the training set), I_P is a positive example and I_N is a random negative example and α is the margin enforced between the anchor and negative images. Instead of the triplet loss the NetVLAD network is optimized using the triplet ranking loss

$$\max\left(\min_i ||(f_\theta(I_A) - f_\theta(I_P^{(i)})||^2 - ||f_\theta(I_A) - f_\theta(I_N)||^2 + \alpha, 0\right), \tag{4}$$

that can handle multiple positive candidates $I_P^{(i)}$ and select only the distance to the best match. The triplet ranking loss is needed since the Google Street View Time Machine dataset contains panoramic images that are converted to multiple projective images and only the images viewing the same direction are correct matches. However, since in our case the images come from a LiDAR that can be considered as a projective sensor we adopt the standard triplet loss from [28].

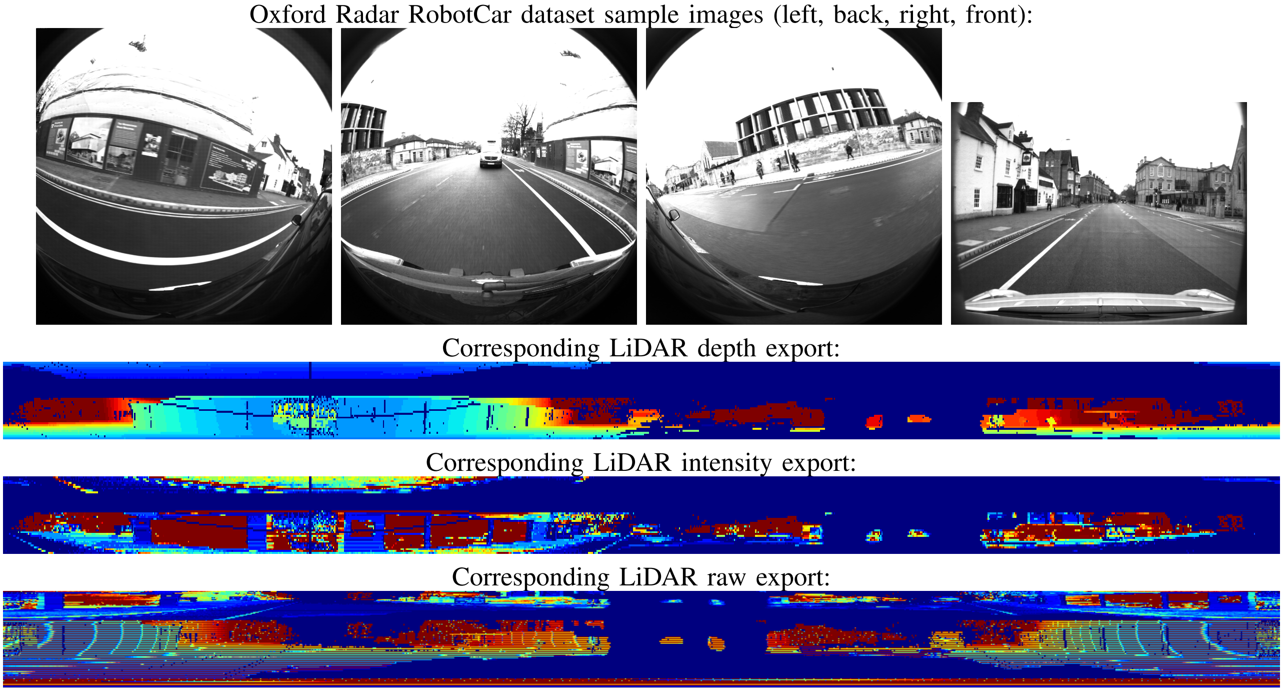
Oxford Radar RobotCar dataset sample images (left, back, right, front):
Corresponding LiDAR depth export:
Corresponding LiDAR intensity export:
Corresponding LiDAR raw export:

Fig. 2. Example from the Oxford Radar RobotCar [7]. The dataset is fully outdoors. The Velodyne LiDAR exports (bottom) are colorized, brightened and cropped for better visualization. The actual images fed to the network are 1024 × 41 pix 8-bit greyscale.

B. Deep image retrieval (CNNRetr)

CNN Retrieval, CNNRetr [10][25], is an image retrieval method. The key components of CNNRetr are a fully convolutional backbone network, generalized-mean pooling layer, siamese architecture with contrastive loss, and whitening with dimension reduction. After training, the image retrieval is done by exhaustively comparing the Euclidean distances between the resulting feature vectors.

The fully convolutional neural network used in CNNRetr [10][25] can be of any convolutional architecture, such as the popular VGG or ResNet. The 3D tensor output χ of size $W*H*K$ from the network is fed into a pooling layer.

CNNRetr [10] employs the usage of generalized-mean (GeM) pooling,

$$\mathbf{f} = [f_1 ... f_k ... f_K]^\top, f_k = \left(\frac{1}{|\chi_k|} \sum_{x \in \chi_k} x^{p_k} \right)^{\frac{1}{p_k}}, \tag{5}$$

where the input χ is pooled into a vector $\mathbf{f}$. χ_k is the set of activations for the feature map $k \in \{1...K\}$. The pooling parameter p_k can be learned, as the GeM layer is differentiable.

The GeM-pooled and l_2-normalized feature vector is used for contrastive loss training in a siamese network architecture. Input images are fed as pairs (i, j) with corresponding labels $Y(i, j) \in \{0, 1\}$, where 1 means that the images are matches, and 0 means that the images are not matches. The contrastive loss,

$$L(i,j) = \begin{cases} \frac{1}{2}\|\mathbf{f}(i) - \mathbf{f}(j)\|^2, & Y(i,j) = 1 \\ \frac{1}{2}(max\{0, \tau - \|\mathbf{f}(i) - \mathbf{f}(j)\|\}^2, & Y(i,j) = 0 \end{cases}, \tag{6}$$

decreases the Euclidean distance between matching images and increases between non-matching images. The parameter τ is the enforced margin between the non-matching examples.

After training, feature vector whitening is performed to improve search precision, and dimension reduction is optionally done to improve performance and resource requirements. CNNRetr learns whitening by employing a structure from motion (SfM) pipeline to reconstruct a scene to extract matching points. They use linear discriminant projections, which involves two phases: whitening and rotation. Here, the first part, whitening, is calculated as the inverse square root of the covariance matrix within each matching class, $C_S^{-\frac{1}{2}}$, where

$$C_S = \sum_{Y(i,j)=1} (\mathbf{f}(i) - \mathbf{f}(j))(\mathbf{f}(i) - \mathbf{f}(j))^\top \ . \tag{7}$$

The second part, rotation, of the linear discriminant projection, is the principal component analysis (PCA) of the covariance matrix of the non-matching pairs in the whitened space $eig(C_S^{-\frac{1}{2}} C_D C_S^{-\frac{1}{2}})$, where C_D is basically the same calculation as C_S (Eq. 7) but for non-matches,

$$C_D = \sum_{Y(i,j)=0} (\mathbf{f}(i) - \mathbf{f}(j))(\mathbf{f}(i) - \mathbf{f}(j))^\top \ . \tag{8}$$

The two linear discriminant projection steps are combined as the projection via multiplication $P = C_S^{-\frac{1}{2}} eig(C_S^{-\frac{1}{2}} C_D C_S^{-\frac{1}{2}})$. To apply the projection, the mean GeM vector to perform centering, μ, is taken into account to get the wanted variance. The applied projection finally is

$P^{\top}(\mathbf{f}(i) - \mu)$, which is also l_2-normalized to get the fully whitened feature vectors to be used in the search process.

III. Experiments

A. Datasets and settings

The experiments were conducted on the two largest and publicly available datasets suitable for outdoor and indoor navigation: Oxford Radar RobotCar [7] (outdoors) and COsy Localization Database (COLD) [9] (indoors).

Radar RobotCar – The Oxford Radar RobotCar dataset [7] is an extension of the original RobotCar dataset [8] and thus follows the original dataset route in Oxford, UK. It consists of 32 traversals in different traffic, weather, and lighting conditions in January 2019. The new dataset contains measurements from three point cloud radars installed on the top of the car and all provide full 360-degree panoramic views around it. In the middle is a Navtech FMCW radar that provides 400 measurements per scan 4 Hz and on its both side two 20 Hz Velodyne LiDARs of 41.3° vertical FoV sensors. We selected the *left Velodyne LiDAR* for our experiments. Note that the route is always to the same direction. See Figure 2 for example images. The velocity of the car is moderate and thus the distance between two measurements is rarely more than 0.5 m. We selected the following sequences for our experiments:

- Train: Jan-10-2019-11:46, Cloudy
- Gallery: Jan-10-2019-12:32, Cloudy
- Query 1: Jan-10-2019-14:50, Cloudy
- Query 2: Jan-11-2019-12:26, Sunny
- Query 3: Jan-16-2019-14:15, Rainy

Train and Gallery set images were used to train the two methods and the Gallery was also used as the place recognition database (gallery set).

COLD – The COsy Localization Database [9] is an indoor navigation dataset. The data has been gathered in 76 sequences across three different locations in Europe. The sequences are varied in lighting conditions such as sunny, cloudy, and night. The sequences also contain dynamic elements such as people moving and rearranged furniture. The room types are annotated, and odometry is used for localization. The sequences are arranged as some being ”standard” or ”extended”. The standard sequences contain rooms that are found in the sequences from the other two locations, as well, and the extended sequences contain location specific room types.

The data is captured with manually driven mobile robots. The robots are equipped with two Videre Design MDCS2 cameras, one in typical perspective mode and the other capturing omnidirectional images. SICK 2D laser scanner is used to capture range information. SICK 2D provides only a single 360-degree line scan of 361 samples that we convert to artificial depth image by expanding it vertically (Figure 3).

For our experiments, we employed datasets similar to the Oxford Radar RobotCar dataset. We had five sequences from the same office. The train set sequence was sunny. The gallery set sequence was cloudy, and the three tested query set sequences represented all the different light categories: sunny, cloudy, and night time:

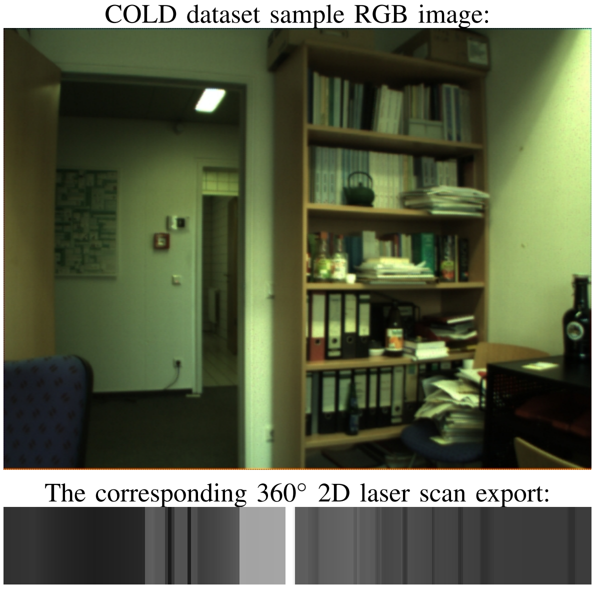
COLD dataset sample RGB image:
The corresponding 360° 2D laser scan export:

Fig. 3. Example from the COsy Localization Database (COLD) [9]. The dataset is fully indoors. The greyscale images below are the expanded SICK laser scans spanning 360-degree around the robot (white encodes distances above 8 m and zero is 0 m). The open door on the left-hand-side can be seen as dark gray region, the opposite wall as light gray and the open doorway as a completely white strip in the middle of the scan.

- Train: Saarbrücken, Part B, Sequence 4, Sunny 3
- Gallery: Saarbrücken, Part B, Sequence 4, Cloudy 1
- Query 1: Saarbrücken, Part B, Sequence 4, Sunny 1
- Query 2: Saarbrücken, Part B, Sequence 4, Cloudy 2
- Query 3: Saarbrücken, Part B, Sequence 4, Night 3

a) Performance measure: All experiments were conducted using the top-1 retrieval results, i.e. only the best matching image was used. Our performance measure is thus Recall@1 i.e. the number of correctly retrieved locations divided by the number of all query images [24].

b) Settings: If not otherwise mentioned the default parameters of CNNRetr and NetVLAD networks from the original authors were used.

For the both methods we used the triplet loss as that was found performing well and makes comparison between the two architectures fair. The positive P and anchor A samples were selected randomly within the given distance threshold used in training. A list of positive matches for each image was generated prior training. The hard negative mining was conducted according to [24], [29].

B. Results

a) Method comparison: The two methods compared were CNNRetr [25] and NetVLAD [24] described in Section II. In the first experiment, we compared using the methods without fine-tuning to our datasets. CNNRetr [10] is used as is, however for NetVLAD we do not perform the PCA-based dimensional reduction as the results in [25] suggest that PCA may degrade the results, which we also

TABLE I

COMPARISON OF THE TWO METHODS WITH THEIR DEFAULT SETTINGS (VGG16 BACKBONE AND WITHOUT FINE-TUNING) AND A TRAINED CNNRETR.

	Outdoor dataset - RobotCar [8]				Indoor dataset - COLD [9]			
Method	Rec@1-25m	Rec@1-10m	Rec@1-5m	Rec@1-2m	Rec@1-100cm	Rec@1-50cm	Rec@1-25cm	Rec@1-10cm
Query 1 (same day, 2h later)					*Query 1 (sunny)*			
NetVLAD [24] (no train)	0.899	0.867	0.728	0.130	0.838	**0.779**	**0.464**	**0.008**
CNNRetr [10] (no train)	0.926	0.901	0.774	0.134	0.783	0.723	0.439	0.004
CNNRetr [10] (trained)	**0.986**	**0.977**	**0.869**	**0.155**	**0.847**	0.774	0.462	0.004
Query 2 (next day, same time)					*Query 2 (cloudy)*			
NetVLAD [24] (no train)	0.895	0.863	0.762	0.454	**0.230**	<0.000	<0.000	<0.000
CNNRetr [10] (no train)	0.887	0.856	0.758	0.468	0.100	<0.000	<0.000	<0.000
CNNRetr [10] (trained)	**0.993**	**0.984**	**0.902**	**0.544**	0.091	<0.000	<0.000	<0.000
Query 3 (after 6 days, 2h later)					*Query 3 (night)*			
NetVLAD [24] (no train)	0.527	0.449	0.325	0.049	0.264	0.005	<0.000	<0.000
CNNRetr [10] (no train)	0.678	0.608	0.465	0.060	**0.267**	0.005	<0.000	<0.000
CNNRetr [10] (trained)	**0.918**	**0.856**	**0.642**	**0.089**	0.263	0.005	**0.001**	<0.000

found out to happen in our experiments. The results for the two dataset without fine-tuning are in Table I.

These results provide the following three findings: 1) there is no substantial performance difference between CNNRetr and NetVLAD; 2) the accuracy of LiDAR-based place recognition is between 2-5 meters with the RobotCar dataset and 25-50 centimeters with the COLD dataset (top-1 recall above 70%); 3) LiDAR-based recognition fails for the indoor dataset query sequences that are substantially different from the gallery dataset (Query 2 and Query 3). This can be explained by the fact that the 360-degree line LiDAR of the COLD dataset does not provide enough information for place recognition. Since COLD is the only indoor dataset for long-term place recognition and including LiDAR there is obvious need for new indoor navigation datasets.

b) Fine-tuning with training data: The best performing method (CNNRetr [25]) was trained with dataset specific training data (gallery sequence and one training sequence). The top-1 recall (Rec@1) values are shown in Table I. The results clearly demonstrate that dataset specific training improves the results by 10-20%. However, the training did not improve the results for Query 2 and 3 images of the indoor dataset that still failed.

c) Backbone network: The typical image retrieval backbone networks are VGG16 and ResNet-50 which were compared during our experiments. The results are shown only for the Radar RobotCar dataset as the indoor results overall were poor for Query 2 and 3 sets. The results are shown in Table II. Clearly, the selection of backbone has only small impact and thus VGG16 is preferable as it is computationally lighter.

d) LiDAR intensity vs. range: LiDAR intensity and depth scan performance are compared in Table III and as functions of the training epochs in Figure 4. While in Query 1 the LiDAR intensity is slightly better, the depth is clearly better in Query 2 and 3 where the conditions are more challenging. The RGB results with the same network are also added to demonstrate that LiDAR-only accuracy is comparable to RGB. Interestingly the "raw" LiDAR data provided with RobotCar data is worse than the depth channel.

TABLE II

BACKBONE COMPARISON USING CNNRETR. THE DETECTION THRESHOLD OF 5.0 M WAS USED IN TRAINING.

	Outdoor dataset - RobotCar [8]			
Method	Rec@1-25m	Rec@1-10m	Rec@1-5m	Rec@1-2m
Query 1 (same day, 2h later)				
VGG16	**0.976**	0.966	**0.866**	**0.154**
ResNet-50	0.970	**0.957**	0.846	**0.154**
Query 2 (next day, same time)				
VGG16	**0.987**	**0.979**	**0.897**	**0.573**
ResNet-50	0.953	0.941	0.869	0.555
Query 3 (after 6 days, 2h later)				
VGG16	**0.832**	**0.777**	0.587	**0.081**
ResNet-50	0.830	0.773	**0.601**	0.080

IV. CONCLUSION

Our experiments provide the following important findings: i) LiDAR is competitive sensor modality (vs. RGB camera) for place recognition, ii) LiDAR depth maps are more robust to long-term changes than LiDAR intensity images, iii) SoTA deep image retrieval architecture "CNNRetr" by Radenovic et al. [10] provides place recognition accuracy of 5 meters urban outdoors and 50 centimeters with recall approx. 80% iv) the backbone network selection is not critical, and v) feature fine-tuning with dataset specific data provides improvement of 10-20%. Two important future directions were also pointed out: a) new indoor navigation datasets with high quality LiDAR are needed and b) complementarity of LiDAR depth, LiDAR intensity and RGB should be further investigated.

REFERENCES

[1] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.

[2] X. Zhang, L. Wang, and Y. Su, "Visual place recognition: A survey from deep learning perspective," *Pattern Recognition*, p. 107760, 2020.

[3] M. Shakeri and H. Zhang, "Illumination invariant representation of natural images for visual place recognition," in *IROS*, 2016.

[4] M. Ullah, A. Pronobis, B. Caputo, J. Luo, P. Jensfelt, and H. Christensen, "Towards robust place recognition for robot localization," in *IROS*, 2008.

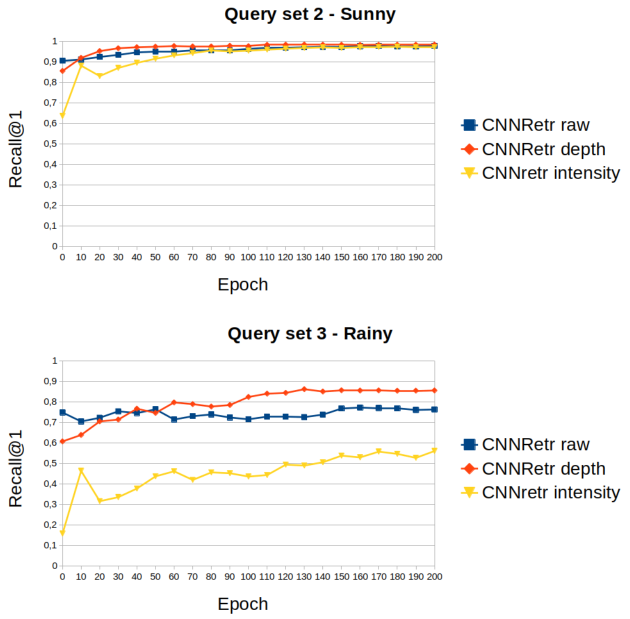

Fig. 4. In sunny weather (top) all the Velodyne LiDAR export modes perform well. In rainy conditions (bottom) the depth mode performs clearly better than the intensity mode and slightly better than the "raw" data.

TABLE III

LiDAR intensity vs. LiDAR range (depth) comparison (CNNRetr, ResNet-50, 5.0m training threshold).

	Outdoor dataset - RobotCar [8]			
Method	Rec@1-25m	Rec@1-10m	Rec@1-5m	Rec@1-2m
Query 1 (same day, 2pm50)				
Intensity	0.974	0.964	0.863	0.142
Depth	0.970	0.957	0.846	0.154
Raw	0.944	0.921	0.786	0.130
RGB	**0.984**	**0.974**	**0.958**	**0.598**
Query 2 (next day, 12pm26)				
Intensity	0.940	0.927	0.855	0.542
Depth	**0.953**	**0.941**	**0.869**	**0.555**
Raw	0.938	0.912	0.809	0.519
RGB	0.951	0.934	**0.869**	0.480
Query 3 (after 6 days, 2pm15)				
Intensity	0.560	0.509	0.381	0.056
Depth	0.830	0.773	0.601	0.080
Raw	0.685	0.627	0.493	0.070
RGB	**0.920**	**0.890**	**0.846**	**0.650**

[5] N. Sunderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford, "On the performance of convnet features for place recognition," in *IROS*, 2015.

[6] Z. Xin, Y. Cai, T. Lu, X. Xing, S. Cai, J. Zhang, Y. Yang, and Y. Wang, "Localizing discriminative visual landmarks for place recognition," in *ICRA*, 2010.

[7] D. Barnes, M. Gadd, P. Murcutt, P. Newman, and I. Posner, "The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset," in *ICRA*, 2020.

[10] F. Radenović, G. Tolias, and O. Chum, "CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples," in *ECCV*,

[8] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 1, pp. 3–15, 2017.

[9] A. Pronobis and B. Caputo, "COLD: COsy Localization Database," *The International Journal of Robotics Research (IJRR)*, vol. 28, no. 5, 2009.

2016.

[11] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *CVPR*, 2019.

[12] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning feature matching with graph neural networks," in *CVPR*, 2020.

[13] N. Pion, M. Humenberger, G. Csurka, Y. Cabon, and T. Sattler, "Benchmarking image retrieval for visual localization," in *Int. Conf. on 3D Vision (3DV)*, 2020.

[14] M. A. Uy and G. H. Lee, "Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4470–4479.

[15] B. Steder, G. Grisetti, and W. Burgard, "Robust place recognition for 3d range data based on point features," in *ICRA*, 2010.

[16] J. Guo, P. V. Borges, C. Park, and A. Gawel, "Local descriptor for robust place recognition using lidar intensity," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1470–1477, 2019.

[17] S. Xie, C. Pan, Y. Peng, K. Liu, and S. Ying, "Large-scale place recognition based on camera-lidar fused descriptor," *Sensors*, vol. 20, no. 10, p. 2870, 2020.

[18] P. Yin, L. Xu, Z. Liu, L. Li, H. Salman, Y. He, W. Xu, H. Wang, and H. Choset, "Stabilize an unsupervised feature learning for lidar-based place recognition," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1162–1167.

[19] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4802–4809.

[20] G. Kim, Y. S. Park, Y. Cho, J. Jeong, and A. Kim, "Mulran: Multimodal range dataset for urban place recognition," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6246–6253.

[21] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The newer college dataset: Handheld lidar, inertial and vision with ground truth," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[22] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of michigan north campus long-term vision and lidar dataset," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2016.

[23] F. Warburg, S. Hauberg, M. Lopex-Antequera, P. Gargallo, Y. Kuang, and J. Civera, "Mapillary street-level sequences: A dataset for lifelong place recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[24] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: Cnn architecture for weakly supervised place recognition," *TPAMI*, 2018.

[25] F. Radenović, G. Tolias, and O. Chum, "Fine-tuning CNN image retrieval with no human annotation," *TPAMI*, 2018.

[26] ——, "Deep shape matching," in *ECCV*, 2018.

[27] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," in *CVPR*, 2010.

[28] F. Schroff, D. Kalenichenko, and 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.

[29] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.