OverlapNet: Loop Closing for LiDAR-based SLAM

Xieyuanli Chen* Thomas Läbe* Andres Milioto* Timo Röhling*,‡
Olga Vysotska†,* Alexandre Haag† Jens Behley* Cyrill Stachniss*

*Photogrammetry & Robotics Lab, University of Bonn, Germany
‡Fraunhofer FKIE, Wachtberg, Germany
†Autonomous Intelligent Driving GmbH, Munich, Germany

***Abstract*—Simultaneous localization and mapping (SLAM) is a fundamental capability required by most autonomous systems. In this paper, we address the problem of loop closing for SLAM based on 3D laser scans recorded by autonomous cars. Our approach utilizes a deep neural network exploiting different cues generated from LiDAR data for finding loop closures. It estimates an image overlap generalized to range images and provides a relative yaw angle estimate between pairs of scans. Based on such predictions, we tackle loop closure detection and integrate our approach into an existing SLAM system to improve its mapping results. We evaluate our approach on sequences of the KITTI odometry benchmark and the Ford campus dataset. We show that our method can effectively detect loop closures surpassing the detection performance of state-of-the-art methods. To highlight the generalization capabilities of our approach, we evaluate our model on the Ford campus dataset while using only KITTI for training. The experiments show that the learned representation is able to provide reliable loop closure candidates, also in unseen environments.**

I. Introduction

Simultaneous localization and mapping or SLAM [1, 29] is an integral part of most robots and autonomous cars. Graph-based SLAM often relies on (i) pose estimation relative to a recent history, which is called odometry or incremental scan matching, and (ii) loop closure detection, which is needed for data association on a global scale. Loop closures enable SLAM approaches to correct accumulated drift resulting in a globally consistent map.

In this paper, we propose a new method to loop closing for laser range scans produced by a rotating 3D LiDAR sensor installed on a wheeled robot or similar vehicle. Instead of using handcrafted features [15, 31], we propose a deep neural network designed to find loop closure candidates. Our network predicts both, a so-called overlap defined on range images and a relative yaw angle between two 3D LiDAR scans recorded with a typical sensor setup often used on automated cars. The concept of overlap has been used in photogrammetry to estimate image overlaps, see also Sec. III-A and III-B, and we use it on LiDAR range images. It is a useful tool for loop closure detection as illustrated in Fig. 1 and can quantify the quality of matches. The yaw estimate serves as an initial guess for a subsequent application of iterative closest point (ICP) [4] to determine the relative pose between scans to derive loop closures constraints for the pose graph optimization. Instead of ICP, one could also use global scan matching [7, 37, 34] to estimate the relative pose between scans.

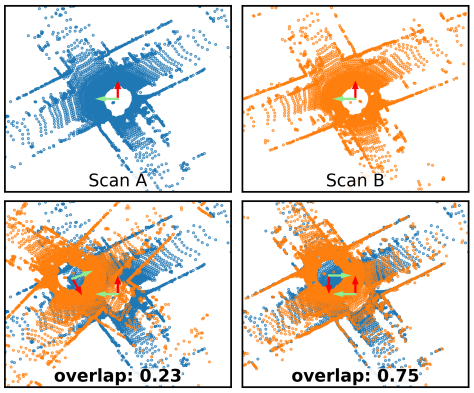


Fig. 1: Overlap of two scans (blue and orange points) at a loop closure location but computed with different relative transformations. The overlap depends on the relative transformation and larger overlap values often correspond to better alignment between the point clouds. Our approach can predict the overlap *without* knowing the relative transformation between the scans.

The main contribution of this paper is a deep neural network that exploits different types of information generated from LiDAR scans to provide overlap and relative yaw angle estimates between pairs of 3D scans. This information includes depth, normals, and intensity or remission values. We additionally exploit a probability distribution over semantic classes that can be computed for each laser beam. Our approach relies on a spherical projection of LiDAR scans, rather than the raw point clouds, which makes the proposed OverlapNet comparably lightweight. We furthermore integrate it into a state-of-the-art SLAM system [3] for loop closure detection and evaluate its performance also with respect to generalization to different environments.

We train the proposed OverlapNet on parts of the KITTI odometry dataset and evaluate it on unseen data.We thoroughly evaluate our approach, provide ablation studies using different modalities, and test the integrated SLAM system in an online manner. Furthermore, we provide results for the Ford campus dataset, which was recorded using a different sensor setup in a different country and a differently structured environment. The experimental results suggest that our method outperforms other state-of-the-art baseline methods and is also able to generalize well to unseen environments.

In sum, our approach is able to (i) predict the overlap and relative yaw angle between pairs of LiDAR scans by

exploiting multiple cues without using relative poses, (ii) combine odometry information with overlap predictions to detect correct loop closure candidates, (iii) improve the overall pose estimation results in a state-of-the-art SLAM system yielding more globally consistent maps, (iv) solve loop closure detection without prior pose information, (v) initialize ICP using the OverlapNet predictions yielding correct scan matching results. The implementation of our approach is available at: https://github.com/PRBonn/OverlapNet

II. Related Work

Loop closure detection using various sensor modalities [16, 28] is a classical topic in robot mapping. We refer to the article by Lowry *et al*. [21] for an overview of approaches using cameras. Here, we mainly concentrate on related work addressing 3D LiDAR-based approaches.

Steder *et al*. [31] propose a place recognition system operating on range images generated from 3D LiDAR data that uses a combination of bag-of-words and a NARF-feature-based [30] relative poses estimation exploiting ideas of FABMAP [11]. Röhling *et al*. [26] present an efficient method for detecting loop closures through the use of similarity measures on histograms extracted from 3D LiDAR scans. The work by He *et al*. [15] presents M2DP, which projects a LiDAR scan into multiple reference planes to generate a descriptor using a density signature of points in each plane. Besides using pure geometric information, there is also work [9, 14] exploiting the remission information, i.e., how well LiDAR beams are reflected by a surface, to create descriptors for localization and loop closure detection with 3D LiDAR data.

Motivated by the success of deep learning in computer vision [20], deep learning-based methods have been proposed recently. Barsan *et al*. [2] propose a deep network-based localization method, which embeds LiDAR sweeps and intensity maps into a joint embedding space and achieves localization by matching between these embeddings. Dubé *et al*. [12] advocate the usage of segments for loop closure detection. Cramariuc *et al*. [10] train a CNN to extract descriptors from segments and use it to retrieve near-by place candidates. Schaupp *et al*. [27] propose a system called OREOS for place recognition, that also estimates the yaw discrepancy between scans. Furthermore, Yin *et al*. [35] develop LocNet, which uses semi-handcrafted feature learning based on a siamese network to solve place recognition. Lu *et al*. [22] proposed L^3-net, which uses 3D convolutions and a recurrent neural network to learn local descriptors for global localization. Uy and Lee [33] proposed PointNetVLAD to generate a global descriptor for 3D point clouds. Kim *et al*. [19] proposed a learning-based descriptor called SCI to solve long-term global localization. Most recently, Sun *et al*. [32] also proposed a learning-based method combined with Monte Carlo localization to achieve a fast global localization.

Contrary to the above-mentioned methods, our method exploits multiple types of information extracted from 3D LiDAR scans, including depth, normal information, intensity/remission and probabilities of semantic classes generated by a semantic segmentation system [23].

Similar to LocNet [35], we also use a siamese network, but we learn features and yield predictions end-to-end. Our network can directly provide estimates for overlap and the relative yaw angle between pairs of LiDAR scans. Different from OREOS [27], our method not only provides loop closures candidates but also an estimate of the matching quality in terms of the overlap.

Recently, Zaganidis *et al*. [36] proposed a Normal Distributions Transform (NDT) histogram-based loop closure detection method, which is also assisted by semantic information. In contrast to ours, their method needs a dense global map and cannot estimate the relative yaw angle.

III. Our Approach

A. The Concept of Overlap

The idea of overlap that we are using here has its origin in the photogrammetry and computer vision community [18]. To successfully match two images and calculate their relative pose, the images must overlap. This can be quantified by defining the overlap percentage as the percentage of pixels in the first image, which can successfully be projected back into the second image without occlusion. Note that this measure is not symmetric: If there is a large scale difference of the image pair, e.g., one image shows a wall and the other shows many buildings around that wall, the overlap percentage for the first to the second image can be large and from the second to the first image low. In this paper, we use the idea of overlap for range images, exploiting the range information explicitly.

For loop closing, a threshold on the overlap percentage can be used to decide whether two LiDAR scans are at the same place and/or a loop closing can be done. For loop closing, this measure maybe even better than the commonly used distance between the recorded positions of a pair of scans, since the positions might be affected by drift and therefore unreliable. The overlap predictions are independent of the relative poses and can be therefore used to find loop closures without knowing the correct relative pose between scans. Fig. 1 shows the overlap of two scans as an example.

B. Definition of the Overlap between Pairs of LiDAR Scans

We use spherical projections of LiDAR scans as input data, which is often used to speed up computations [3, 5, 8]. We project the point cloud $\mathcal{P}$ to a so-called vertex map $\mathcal{V} : \mathbb{R}^2 \mapsto \mathbb{R}^3$, where each pixel is mapped to the nearest 3D point. Each point $\mathbf{p}_i = (x, y, z)$ is converted via the function $\Pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$ to spherical coordinates and finally to image coordinates (u, v) by

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \frac{1}{2}\left[1 - \arctan(y, x)\pi^{-1}\right] w \\ \left[1 - \left(\arcsin(zr^{-1}) + \mathrm{f}_{\mathrm{up}}\right)\mathrm{f}^{-1}\right] h \end{pmatrix}, \quad (1)$$

where $r = ||\mathbf{p}||_2$ is the range, $\mathrm{f} = \mathrm{f}_{\mathrm{up}} + \mathrm{f}_{\mathrm{down}}$ is the vertical field-of-view of the sensor, and w, h are the width and height of the resulting vertex map $\mathcal{V}$.

For a pair of LiDAR scans $\mathcal{P}_1$ and $\mathcal{P}_2$, we generate the corresponding vertex maps $\mathcal{V}_1$, $\mathcal{V}_2$. We denote the sensor-centered coordinate frame at time step t as C_t. Each pixel in coordinate frame C_t is associated with the world frame W by a pose $\mathbf{T}_{WC_t} \in \mathbb{R}^{4\times4}$. Given the poses $\mathbf{T}_{WC_1}$ and $\mathbf{T}_{WC_2}$, we can reproject scan $\mathcal{P}_1$ into the coordinate frame of the other's vertex map $\mathcal{V}_2$ and generate a reprojected vertex map $\mathcal{V}_1'$:

$$\mathcal{V}_1' = \Pi\left(\mathbf{T}_{WC_1}^{-1}\mathbf{T}_{WC_2}\mathcal{P}_1\right). \tag{2}$$

We then calculate the absolute difference of all corresponding pixels in $\mathcal{V}_1'$ and $\mathcal{V}_2$, considering only those pixels that correspond to valid range readings in both range images. The overlap is then calculated as the percentage of all differences in a certain distance ϵ relative to all valid entries, i.e., the overlap of two LiDAR scans $O_{C_1C_2}$ is defined as follows:

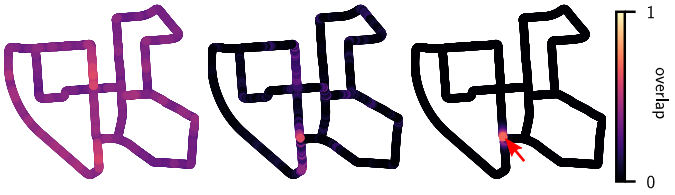
$$O_{C_1C_2} = \frac{\sum_{(u,v)} \mathbb{I}\Big\{ ||\mathcal{V}_1'(u,v) - \mathcal{V}_2(u,v)|| \leq \epsilon \Big\}}{\min\left(\text{valid}(\mathcal{V}_1'), \text{valid}(\mathcal{V}_2)\right)}, \tag{3}$$

where $\mathbb{I}\{a\} = 1$ if a is true and 0 otherwise. valid$(\mathcal{V})$ is the the number of valid pixels in $\mathcal{V}$, since not all pixel might have a valid LiDAR measurement associated after the projection.

We use Eq. (3) only for creating training data, i.e., only positive examples of correct loop closures get a non-zero overlap assigned using the relative poses between scans, as shown in Fig. 2(c). However, when performing loop closure detection for online SLAM, the approximate relative poses from SLAM before loop closure are not accurate enough to calculate usable overlaps by using Eq. (3) because of accumulated drift. We tried directly estimating overlaps using Eq. (3) assuming the relative pose as identity and applying different orientations, e.g., every 30 degrees rotation around the vertical axis, and using the maximum over all these overlaps as an estimate. Fig. 2 shows the estimated overlaps for all scans using a query scan produced by this method and the result of the estimated overlap for all scans using OverlapNet. We leave out the 100 most recent scans because they will not be loop closure candidates. In the case of the exhaustive approach, many wrong loop closure candidates get high overlap values, while our approach performs better since it produces a highly distinctive peak around the correct location. Furthermore, it takes on average 1.2 s to calculate the overlap for one pair of scans using the exhaustive approach, which makes it unusable in real-world scenarios. In contrast, the complete OverlapNet needs on average 17 ms for one pair overlap estimation when using depth and normal information only.

C. Overlap Network Architecture

The overview of the proposed OverlapNet is depicted in Fig. 3. We exploit multiple cues, which can be generated from a single LiDAR scan, including depth, normal, intensity, and semantic class probability information. The depth information is stored in the range map $\mathcal{R}$, which consists of one channel. We use neighborhood information of the vertex map to generate a normal map $\mathcal{N}$, which has three channels encoding the normal coordinates. We directly obtain the intensity information, also called remission, from the sensor and represent the intensity information as a one-channel intensity map $\mathcal{I}$. The point-wise semantic class probabilities are computed using RangeNet++ [23] and we represent them as a semantic map $\mathcal{S}$. RangeNet++ delivers probabilities for 20 different classes. For efficiency sake, we reduce the 20-dimensional RangeNet++ output to a compressed 3-dimensional vector using principal component analysis. The information is combined into an input tensor of size $64 \times 900 \times D$, where $64, 900$ are the height and width of the inputs, and D depends on the types of data used.



(a) Exhaustive evaluation of Eq. (3) (b) OverlapNet estimates (c) Ground truth

Fig. 2: Overlap estimations of one frame to all others. The red arrow points out the position of the query scan. If we directly use Eq. (3) to estimate the overlap between two LiDAR scans without knowing the accurate relative poses, it is hard to decide which pairs of scans are true loop closures, since most evaluations of Eq. (3) show high values. In contrast, our OverlapNet can predict the overlaps between two LiDAR scans well.

Our proposed OverlapNet is a siamese network architecture [6], which consists of two legs sharing weights and two heads that use the same pair of feature volumes generated by the two legs. The trainable layers are listed in Tab. I.

1) Legs: The proposed OverlapNet has two legs, which have the same architecture and share the same weights. Each leg is a fully convolutional network (FCN) consisting of 11 convolutional layers. This architecture is quite lightweight and generates feature volumes of size $1 \times 360 \times 128$. Note that our range images are cyclic projections and that a change in the yaw angle of the vehicle results in a cyclic column shift of the range image. Thus, the single row in the feature volume can represent a relative yaw angle estimate (because a yaw angle rotation results in a pure horizontal shift of the input maps). As the FCN is translation-equivariant, the feature volume will be shifted horizontally. The number of columns of the feature volume defines the resolution of the yaw estimation, which is 1 degree in the case of our leg architecture.

2) Delta Head: The delta head is designed to estimate the overlap between two scans. It consists of a delta layer, three convolutional layers, and one fully connected layer.

The delta layer, shown in Fig. 4, computes all possible absolute differences of all pixels. It takes the output feature volumes $\mathbf{L}^l \in \mathbb{R}^{H\times W\times C}$ from the two legs l as input. These are stacked in a tiled tensor $\mathbf{T}^l \in \mathbb{R}^{HW\times HW\times C}$ as follows:

$$\mathbf{T}^0(iW + j, k, c) = \mathbf{L}^0(i, j, c) \tag{4}$$

$$\mathbf{T}^1(k, iW + j, c) = \mathbf{L}^1(i, j, c), \tag{5}$$

with $k = \{0, \ldots, HW - 1\}$, $i = \{0, \ldots, H - 1\}$ and $j = \{0, \ldots, W - 1\}$.

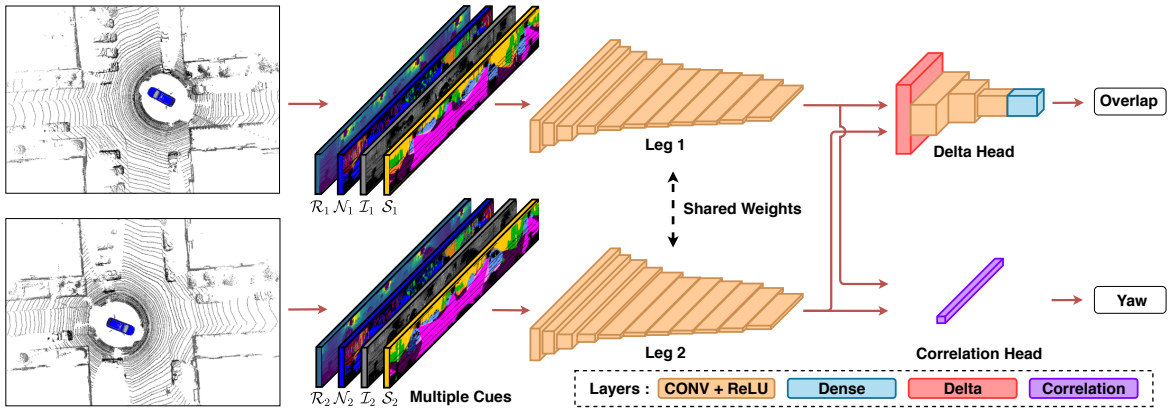


Fig. 3: Pipeline overview of our proposed approach. The left-hand side shows the preprocessing of the input data which exploits multiple cues generated from a single LiDAR scan, including range $\mathcal{R}$, normal $\mathcal{N}$, intensity $\mathcal{I}$, and semantic class probability $\mathcal{S}$ information. The right-hand side shows the proposed OverlapNet which consists of two legs sharing weights and the two heads use the same pair of feature volumes generated by the two legs. The outputs are the overlap and relative yaw angle between two LiDAR scans.

TABLE I: Layers of our network architecture

	Operator	Stride	Filters	Size	Output Shape
Legs	Conv2D	(2, 2)	16	(5, 15)	$30 \times 443 \times 16$
	Conv2D	(2, 1)	32	(3, 15)	$14 \times 429 \times 32$
	Conv2D	(2, 1)	64	(3, 15)	$6 \times 415 \times 64$
	Conv2D	(2, 1)	64	(3, 12)	$2 \times 404 \times 64$
	Conv2D	(2, 1)	128	(2, 9)	$1 \times 396 \times 128$
	Conv2D	(1, 1)	128	(1, 9)	$1 \times 388 \times 128$
	Conv2D	(1, 1)	128	(1, 9)	$1 \times 380 \times 128$
	Conv2D	(1, 1)	128	(1, 9)	$1 \times 372 \times 128$
	Conv2D	(1, 1)	128	(1, 7)	$1 \times 366 \times 128$
	Conv2D	(1, 1)	128	(1, 5)	$1 \times 362 \times 128$
	Conv2D	(1, 1)	128	(1, 3)	$1 \times 360 \times 128$
Delta Head	Conv2D	(1, 15)	64	(1, 15)	$360 \times 24 \times 64$
	Conv2D	(15, 1)	128	(15, 1)	$24 \times 24 \times 128$
	Conv2D	(1, 1)	256	(3, 3)	$22 \times 22 \times 256$
	Dense	-	-	-	1

Note that $\mathbf{T}^1$ is transposed in respect to $\mathbf{T}^0$, as depicted in the middle of Fig. 4. After that, all differences are calculated by element-wise absolute differences between $\mathbf{T}^0$ and $\mathbf{T}^1$.

By using the delta layer, we can obtain a representation of the latent difference information, which can be later exploited by the convolutional and fully-connected layers to estimate the overlap. Different overlaps induce different patterns in the output of the delta layer.

3) Correlation Head: The correlation head [24] is designed to estimate the yaw angle between two scans using the feature volumes of the two legs. To perform the cross-correlation, we first pad horizontally one feature volume by copying the same values (as the range images are cyclic projections around the yaw angle). This doubles the size of the feature volume. We then use the other feature volume as a kernel that is shifted over the first feature volume generating a 1D output of size 360. The argmax of this feature serves as the estimate of the relative yaw angle of the two input scans with a 1 degree resolution.

D. Loss Functions

We train our OverlapNet end-to-end to estimate the overlap and the relative yaw angle between two LiDAR scans at the same time. Typically, to train a neural network one needs a large amount of manually labeled ground truth data. In our case, this is $(I_1,\ I_2,\ Y_O,\ Y_Y)$, where I_1, I_2 are two inputs and Y_O, Y_Y are the ground truth overlaps and the ground truth yaw angles respectively. We are however able to generate the input and the ground truth without any manual effort in a fully automated fashion given a dataset with pose information. From given poses, we can calculate the ground truth overlap and relative yaw angles directly. We denote the legs part network with trainable weights as $f_L(\cdot)$, the delta head as $f_D(\cdot)$ and the correlation head as $f_C(\cdot)$.

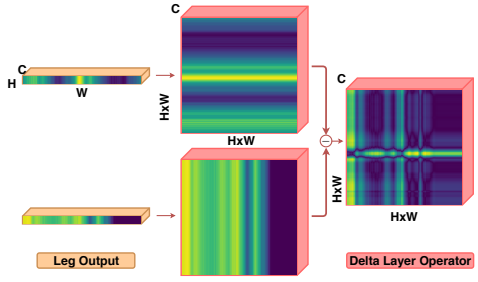


Fig. 4: Delta layer. Computation of pairwise differences is efficiently performed by concatenating the feature volumes and transposition of one concatenated feature volume.

For training, we combine the loss $L_O(\cdot)$ for the overlap and the loss $L_Y(\cdot)$ for the yaw angle using a weight α:

$$L\left(I_1, I_2, Y_O, Y_Y\right) = L_O\left(I_1, I_2, Y_O\right) + \alpha L_Y\left(I_1, I_2, Y_Y\right). \quad (6)$$

We treat the overlap estimation as a regression problem and use a weighted absolute difference of ground truth Y_O and network output $\hat{Y}_O = f_D\left(f_L\left(I_1\right), f_L\left(I_2\right)\right)$ as the loss function. For weighting, we use a scaled sigmoid function:

$$L_O\left(I_1, I_2, Y_O\right) = \text{sigmoid}\left(s\left(\left|\hat{Y}_O - Y_O\right| + a\right) - b\right), \quad (7)$$

with $\text{sigmoid}(v) = (1 + \exp(-v))^{-1}$, a, b are offsets and s being a scaling factor.

For the yaw angle estimation, we use a lightweight representation of the correlation head output, which leads to a

one-dimensional vector of size 360. We take the index of the maximum, the argmax, as the estimate of the relative angle in degrees. As the argmax is not differentiable, we cannot treat this as a simple regression problem. The yaw angle estimation, however, can be regarded as a binary classification problem that decides for every entry of the head output whether it is the correct angle or not. Therefore, we use the binary cross-entropy loss given by

$$L_Y\left(I_1, I_2, Y_Y\right) = \sum\nolimits_{i=\{1,\dots,N\}} \mathrm{H}\left(Y_Y^i, \hat{Y}_Y^i\right), \tag{8}$$

where $\mathrm{H}(p,q) = p\log(q) - (1-p)\log(1-q)$ is the binary cross entropy and N is the size of the output 1D vector. $\hat{Y}_Y = f_C\left(f_L\left(I_1\right), f_L\left(I_2\right)\right)$ is the relative yaw angle estimate. Note that we only train the network to estimate the relative yaw angle of a pair of scans with overlap larger than 30%, since this minimum overlap was needed to result in correct pose estimates of the ICP as explained in Sec. IV-A, but also experimentally validated in Sec. IV-F.

E. SLAM Pipeline

We use the surfel-based mapping system called SuMa [3] as our SLAM pipeline and integrate OverlapNet in SuMa replacing its original heuristic loop closure detection method. We only summarize here the steps of SuMa relevant to our approach and refer for more details to the original paper [3].

SuMa uses the same vertex map $\mathcal{V}_D$ and normal map $\mathcal{N}_D$ as discussed in Sec. III-B. Furthermore, SuMa uses projective ICP with respect to a rendered map view $\mathcal{V}_M$ and $\mathcal{N}_M$ at timestep $t-1$, the pose update $\mathbf{T}_{C_{t-1}C_t}$ and consequently $\mathbf{T}_{WC_t}$ by chaining all pose increments. Therefore, each vertex $\mathbf{u} \in \mathcal{V}_D$ is projectively associated to a reference vertex $\mathbf{v}_\mathbf{u} \in \mathcal{V}_M$. Given this association information, SuMa estimates the transformation between scans by incrementally minimizing the point-to-plane error given by

$$E(\mathcal{V}_D, \mathcal{V}_M, \mathcal{N}_M) = \sum_{\mathbf{u}\in\mathcal{V}_D} \left(\mathbf{n}_u^\top \left(\mathbf{T}_{C_{t-1}C_t}^{(k)}\mathbf{u} - \mathbf{v}_u\right)\right)^2. \tag{9}$$

Each vertex $\mathbf{u} \in \mathcal{V}_D$ is projectively associated to a reference vertex $\mathbf{v}_u \in \mathcal{V}_M$ and its normal $\mathbf{n}_u \in \mathcal{N}_M$ via

$$\mathbf{v}_u = \mathcal{V}_M\left(\Pi\left(\mathbf{T}_{C_{t-1}C_t}^{(k)}\mathbf{u}\right)\right) \tag{10}$$

$$\mathbf{n}_u = \mathcal{N}_M\left(\Pi\left(\mathbf{T}_{C_{t-1}C_t}^{(k)}\mathbf{u}\right)\right). \tag{11}$$

SuMa then minimizes the objective of Eq. (9) using Gauss-Newton and determines increments δ by iteratively solving

$$\delta = \left(\mathbf{J}_\delta^\top \mathbf{W}\mathbf{J}_\delta\right)^{-1}\mathbf{J}_\delta^\top \mathbf{W}\mathbf{r}, \tag{12}$$

where $\mathbf{W} \in \mathbb{R}^{n\times n}$ is a diagonal matrix containing weights w_u, $\mathbf{r} \in \mathbb{R}^n$ is the stacked residual vector, and $\mathbf{J}_\delta \in \mathbb{R}^{n\times 6}$ the Jacobian of $\mathbf{r}$ with respect to the increment δ.

SuMa employs a loop closure detection module, which considers the nearest frame in the built map as the candidate for loop closure given the current pose estimate. Loop closure detection works well for small loops, but the heuristic fails in areas with only a few large loops. Furthermore, drift in the odometry estimate can lead to large displacements, where the heuristic of just taking the nearest frame in the already mapped areas does not yield correct candidates, which will be also shown in our experiments.

F. Covariance Propagation for Geometric Verification

SuMa's loop closure detection uses a fixed search radius. In contrast, we use the covariance of the pose estimate and error propagation to automatically adjust the search radius.

We assume a noisy pose $\mathbf{T}_{C_{t-1}C_t} = \{\bar{\mathbf{T}}_{C_{t-1}C_t}, \Sigma_{C_{t-1}C_t}\}$ with mean $\bar{\mathbf{T}}_{C_{t-1}C_t}$ and covariance $\Sigma_{C_{t-1}C_t}$. We can estimate the covariance matrix by

$$\Sigma_{C_{t-1}C_t} = \frac{1}{K}\frac{E}{N-M}\left(\mathbf{J}_\delta^\top \mathbf{W}\mathbf{J}_\delta\right)^{-1}, \tag{13}$$

where K is the correction factors of the Huber robustized covariance estimation [17], E is the sum of the squared point-to-plane errors (sum of squared residuals) given the pose $\mathbf{T}_{C_{t-1}C_t}$, see Eq. (9), N is the number of correspondences, $M = 6$ is the dimension of the transformation between two 3D poses.

To estimate the propagated uncertainty during the incrementally pose estimation, we can update the mean and covariance as follows:

$$\bar{\mathbf{T}}_{C_{t-1}C_{t+1}} = \bar{\mathbf{T}}_{C_{t-1}C_t}\bar{\mathbf{T}}_{C_tC_{t+1}} \tag{14}$$

$$\Sigma_{C_{t-1}C_{t+1}} \approx \Sigma_{C_{t-1}C_t} + \mathbf{J}_{C_tC_{t+1}}^\top \Sigma_{C_tC_{t+1}}\mathbf{J}_{C_tC_{t+1}}, \tag{15}$$

where $\mathbf{J}_{C_tC_{t+1}}$ is the Jacobian of $\mathbf{T}_{C_tC_{t+1}}$.

Since we need the Mahalanobis distance D_M as a probabilistic distance measure between two poses, we make use of Lie algebra to express $\mathbf{T}$ as a 6D vector $\xi \in \mathfrak{se}(3)$ using $\xi = \log\mathbf{T}$, yielding

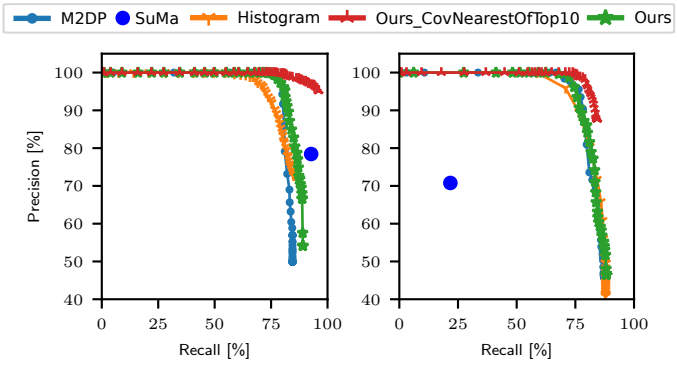
$$D_M\left(\mathbf{T}_{C1}, \mathbf{T}_{C2}\right) = \sqrt{\Delta\xi_{C1C2}^\top \Sigma_{C1C2}^{-1}\Delta\xi_{C1C2}}. \tag{16}$$

Using the scaled distance, we can now restrict the search space depending on the pose uncertainty to save computation time. However, we can use our framework also without any prior information, i.e., perform place recognition.

IV. Experimental Evaluation

The experimental evaluation is designed to support the key claims that our approach is able to: (i) predict the overlap and relative yaw angle between pairs of LiDAR scans by exploiting multiple cues without given poses, (ii) combine odometry information with overlap predictions to detect correct loop closure candidates, (iii) improve the overall pose estimation results in graph-based SLAM yielding more globally consistent maps, (iv) solve loop closure detection without prior pose information, (v) initialize ICP using OverlapNet predictions yielding correct scan matching results.

We train and evaluate our approach on the KITTI odometry benchmark [13], which provides LiDAR scans recorded with a Velodyne HDL-64E of urban areas around Karlsruhe in Germany. We follow the experimental setup of Schaupp *et al.* [27] and use sequence 00 for evaluation. Sequences $03-10$ are used for training and sequence 02 is used for validation.



(a) KITTI sequence 00 (b) Ford campus sequence 00

Fig. 5: Precision-Recall curves of different approaches.

To evaluate the generalization ability of our method, we also test it on the Ford campus dataset [25], which is recorded on the Ford research campus and downtown Dearborn in Michigan using a different version of the Velodyne HDL-64E. In the case of the Ford campus dataset, we test our method on sequence 00 which has several large loops. Note that we never trained our approach on the Ford campus dataset.

For generating overlap ground truth, we only use points within a distance of 75 m to the sensor. For overlap computation, see Eq. (3), we use $\epsilon = 1\,\text{m}$. We use a learning rate of 10^{-3} with a decay of 0.99 every epoch and train at most 100 epochs. For the combined loss, Eq. (6), we set $\alpha = 5$. For the overlap loss, Eq. (7), we use $a = 0.25, b = 12$, and scale factor $s = 24$.

A. Loop Closure Detection

In our first experiments, we investigate the loop closure performance of our approach and compare it to existing methods. Loop closure detection typically assumes that robots revisit places during the mapping while moving with uncertain odometry. Therefore, the prior information about robot poses extracted from the pose graph is available for the loop closure detection. The following criteria are used in these experiments:

- To avoid detecting a loop closure in the most recent scans, we do not search candidates in the latest 100 scans.
- For each query scan, only the best candidate is considered throughout this evaluation.
- Most SLAM systems search for potential closures only within the 3σ area around the current pose estimate. We do the same, either using the Euclidean or the Mahalanobis distance, depending on the approach.
- We use a relatively low threshold of 30 % for the overlap to decide if a candidate is a true positive. We aim to find more loops even in some challenging situations with low overlaps, e.g., when the car drives back to an intersection from the opposite direction (as highlighted in the supplementary video[1]). Furthermore, ICP can find correct poses if the overlap between pairs of scans is around 30 %, as illustrated in the experimental evaluation.

[1]https://youtu.be/YTfliBco6aw

We evaluate OverlapNet on both KITTI sequence 00 and Ford campus sequence 00 using the precision-recall curves shown in Fig. 5. We compare our method, trained with two heads and all cues (labeled as *Ours (AllChannel, TwoHeads)*) with three state-of-the-art approaches, M2DP [15], Histogram [26], and the original SuMa [3]. Since SuMa always uses the nearest frame as the candidate for loop closure detection, we can only get one pair of precision and recall value resulting in a single point. We also show the result of our method using prior information, named *Ours_CovNearestOfTop10*, which uses covariance propagation (Sec. III-F) to define the search space with the Mahalanobis distance and use the nearest in Mahalanobis distance of the top 10 predictions of OverlapNet as the loop closure candidates.

Tab. II shows the comparison between our approach and the state of the art using the F1 score and the area under the curve (AUC) on both KITTI and Ford campus dataset. For the KITTI dataset, our approach uses the model trained with all cues, including depth, normals, intensity, and a probability distribution over semantic classes. For the Ford campus dataset, our approach uses the model trained with geometric information only, namely *Ours (GeoOnly)*, since other cues are not available in this dataset. We can see that our method outperforms the other methods on the KITTI dataset and attains a similar performance on the Ford campus dataset. There are two reasons to explain the worse performance on the Ford campus dataset. First, we never trained our network on the Ford campus dataset or even US roads, and secondly, there is only geometric information available on the Ford campus dataset. However, our method outperforms all baseline methods in both, KITTI and Ford campus dataset, if we integrate prior information.

We also show the performance in comparison to variants of our method in Tab. III. We compare our best model *AllChannel* using two heads and all available cues to a variant which only uses a basic multilayer perceptron as the head named *MLPOnly* which consists of two hidden fully connected layers and a final fully connected layer with two neurons (one for overlap, one for yaw angle). The substantial difference of the AUC and F1 scores shows that such a simple network structure is not sufficient to get a good result. Training the network with only one head (only the delta head for overlap estimation, named *DeltaOnly*), has not a significant influence on the performance. A huge gain can be observed when regarding the nearest frame in Mahalanobis distance of the top 10 candidates in overlap percentage (*CovNearestOfTop10*).

B. Qualitative Results

The second experiment is designed to support the claim that our method is able to improve the overall mapping result. Fig. 6 shows the odometry results on KITTI sequence 02. The color in Fig. 6 shows the 3D translation error (including height). The left figure shows the SuMa and the right figure shows *Ours_CovNearestOfTop10* using the proposed OverlapNet to detect loop closures. We can see that after integrating our method, the overall odometry is much more

TABLE II: Comparison with state of the art.

Dataset	Approach	AUC	F1 score
KITTI	Histogram [26]	0.83	0.83
	M2DP [15]	0.83	0.87
	SuMa [3]	-	0.85
	Ours (AllChannel, TwoHeads)	**0.87**	**0.88**
Ford Campus	Histogram [26]	0.84	0.83
	M2DP [15]	0.84	**0.85**
	SuMa [3]	-	0.33
	Ours (GeoOnly)	**0.85**	0.84

TABLE III: Comparison with our variants.

Dataset	Variant	AUC	F1 score
KITTI	MLPOnly	0.58	0.65
	DeltaOnly	0.85	0.88
	CovNearestOfTop10	**0.96**	**0.96**
	Ours (AllChannel, TwoHeads)	0.87	0.88
Ford Campus	Ours (GeoOnly)	0.85	0.84
	GeoCovNearestOfTop10	**0.85**	**0.88**

accurate since we can provide more loop closure candidates with higher accuracy in terms of overlap. The colors represent the translation error of the estimated poses with respect to the ground truth. Furthermore, after integrating the proposed OverlapNet, the SLAM system can find more loops even in some challenging situations, e.g., when the car drives back to an intersection from the opposite direction, which is highlighted in the supplementary video[1].

C. Loop Closure Detection without Odometry Information

The third experiment is designed to support the claim that our approach is well-suited for solving the more general loop closure detection task without using odometry information.

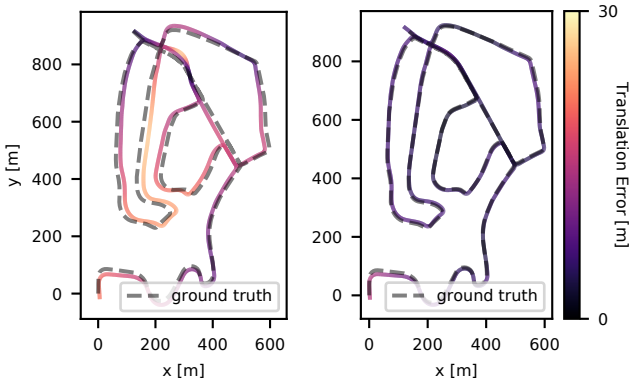
In this case, we assume that we have no prior information about the robot pose. To compare with the state-of-the-art method OREOS [27], we follow their experimental setup and refer to the original paper for more details. The OREOS results are those produced by the authors of OREOS.

The respective loop closure candidates recall results are shown in Fig. 7. Our method outperforms all the baseline methods with a small number of candidates and attains similar performance as baseline methods for higher values of numbers of candidates. However, OREOS and LocNet++ attain a slightly higher recall if more candidates are considered.

D. Yaw Estimation

We aim at supporting our claim that our network provides good relative yaw angle estimates. We use the same experimental setup as described in Sec. IV-C. Tab. IV summarizes the yaw angle errors on KITTI sequence 00.

We can see that our method outperforms the other methods in terms of mean error and standard deviations. In terms of recall, OverlapNet and OREOS always provide a yaw angle estimate, since both approaches are designed to estimate the



(a) SuMa (b) Ours_CovNearestOfTop10

Fig. 6: Qualitative result on KITTI sequence 02.

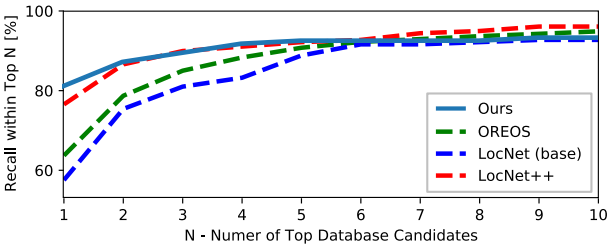


Fig. 7: Loop closure detection performance on KITTI sequence 00.

relative yaw angle for any pairs of scans in contrast to the RANSAC-based method that sometimes fails.

The superior performance can be mainly attributed to the correlation head exploiting the fact that the orientation in LiDAR scans can be well represented by the shift in the range projection. Therefore, it is easier to train the correlation head to accurately predict the relative yaw angles rather than a multilayer perceptron used in OREOS [27]. Furthermore, there is also a strong relationship between overlap and yaw angle, which also improves the results when trained together.

Fig. 8 shows the relationship between real overlap and yaw angle estimation error. As expected, the yaw angle estimate gets better with increasing overlap. Based on these plots, our method not only finds candidates but also measures the quality, i.e., when the overlap of two scans is larger than 90%, our method can accurately estimate the relative yaw angle with an average error of about only 1 degree.

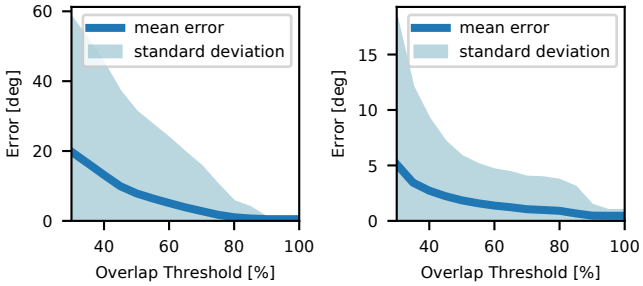
E. Ablation Study on Input Modalities

An ablation study on the usage of different inputs is shown in Tab. V. As can be seen, when employing more input modalities, the proposed method is more robust. We notice that exploiting only depth information with OverlapNet can already perform reasonable in terms of overlap prediction, while it does not perform well in yaw angle estimation. When combining with normal information, the OverlapNet can perform well in both tasks. Another interesting finding is the drastic reduction of yaw angle mean error and standard deviation when using semantic information. One reason could

TABLE IV: Yaw estimation errors without ICP

Approach	Mean[deg]	std[deg]	Recall[%]
FPFH+RANSAC*	13.28	32.19	97
OREOS*	12.67	15.23	100
Ours (AllChannel, TwoHeads)	**1.13**	**3.34**	**100**

*: The results are those produced by the authors of OREOS [27].



(a) KITTI sequence 00 (b) Ford campus sequence 00

Fig. 8: Overlap and yaw estimation relationship.

be that adding semantic information will make the input data more distinguishable when the car drives in symmetrical environments. We also notice that semantic information will increase the computation time, see Sec. IV-G. However, from the ablation study, one could also notice that the proposed method can also achieve good performance by only employing geometric information (depth and normals).

F. Using OverlapNet Predictions as Initial Guesses for ICP

We aim at supporting the claim that our network provides good initializations for ICP with 3D laser scans collected on autonomous cars. Fig. 9 shows the relation between the overlap and ICP registration error with and without using OverlapNet predictions as initial guesses. The error of ICP registration is here depicted by the Euclidean distance between the estimated relative translation and the ground-truth translation. As can be seen, the yaw angle prediction of the OverlapNet increases the chance to get a good result from the ICP even if two frames are relatively far away from each other (with low overlap). Therefore in some challenging cases, e.g. the car drives back into an intersection from a different street, our approach can still find loop closures (see in the supplementary video[1]). The results also show that the overlap estimates measure the quality of the found loop closure: larger overlap values result in better registration results of the involved ICP.

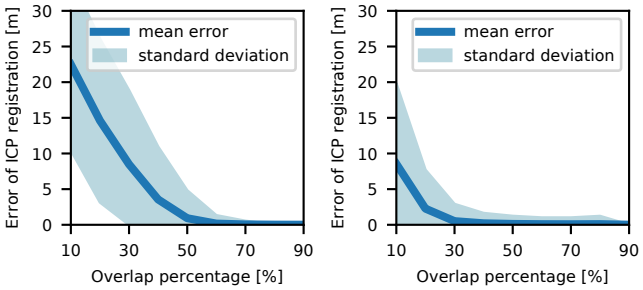
G. Runtime

We tested our method on a system equipped with an Intel i7-8700 with 3.2 GHz and an Nvidia GeForce GTX1080 Ti with 11 GB memory.

For the KITTI sequence 00, we could exploit all input cues including the semantic classes provided by RangeNet++ [23]. We need on average 75 ms per frame for the input data preprocessing, 6 ms per frame for the legs feature extraction, 27 ms per frame for the head matching. The worst case for the head matching takes 630 ms for all candidates in the search space.

TABLE V: Ablation study on usage of input modalities.

Depth	Normals	Intensity	Semantics	overlap AUC	overlap F1	yaw angle[deg] Mean	yaw angle[deg] Std
✓				0.86	0.87	11.67	25.32
✓	✓			0.86	0.85	2.97	14.28
✓	✓	✓		0.87	0.87	2.53	14.56
✓	✓	✓	✓	0.87	0.88	1.13	3.34



(a) With identity as initial guess (b) With yaw angle as initial guess

Fig. 9: ICP using OverlapNet predictions as initial guess. The error of ICP registration here is the Euclidean distance between the estimated translation and the ground-truth translation.

For the Ford campus dataset, we used only geometric information, which could be generated in 10 ms on average per frame, 2 ms for feature extraction and 24 ms for matching with the worst case of 550 ms. In real SLAM operation, we only search loop closure candidates inside a certain search space given by pose uncertainty using the Mahalanobis distance (see Sec. III-F). Therefore, our method can achieve online operation in long-term tasks, since we usually only have to evaluate a small number of candidate poses.

V. Conclusion

In this paper, we presented a novel approach for LiDAR-based loop closure detection. It is based on the overlap between LiDAR scan range images and provides a measure for the quality of the loop closure. Our approach utilizes a siamese network structure to leverage multiple cues and allows us to estimate the overlap and relative yaw angle between scans. The experiments on two different datasets suggest that when combined with odometry information our method outperforms other state-of-the-art methods and that it generalizes well to different environments never seen during training.

Despite these encouraging results, there are several avenues for future research. First, we want to investigate the integration of other input modalities, such as vision and radar information. We furthermore plan to test our approach with other datasets collected in different seasons.

Acknowledgments

This work has been supported in part by the German Research Foundation (DFG) under Germany's Excellence Strategy, EXC-2070 - 390732324 (PhenoRob) and under grant number BE 5996/1-1 as well as by the Chinese Scholarship Committee.

References

[1] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localisation and mapping (SLAM): Part I. *IEEE Robotics and Automation Magazine (RAM)*, 13(2):99–110, 2006. ISSN 1070-9932. doi: 10.1109/MRA.2006.1638022.

[2] Ioan Andrei Barsan, Shenlong Wang, Andrei Pokrovsky, and Raquel Urtasun. Learning to Localize Using a LiDAR Intensity Map. In *Proc. of the Second Conference on Robot Learning (CoRL)*, pages 605–616, 2018.

[3] Jens Behley and Cyrill Stachniss. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In *Proc. of Robotics: Science and Systems (RSS)*, 2018.

[4] Paul J. Besl and Neil D. McKay. A Method for Registration of 3D Shapes. *IEEE Trans. on Pattern Analalysis and Machine Intelligence (TPAMI)*, 14(2): 239–256, 1992.

[5] Igor Bogoslavskyi and Cyrill Stachniss. Fast range image-based segmentation of sparse 3d laser scans for online operation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016.

[6] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature Verification using a "Siamese" Time Delayed Neural Network. *Intl. Journal of Pattern Recognition and Artificial Intelligence*, 07(04):669–688, 1993. doi: 10.1142/S0218001493000339.

[7] Andrea Censi and Stefano Carpin. HSM3D: feature-less global 6DOF scan-matching in the Hough/Radon domain. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 3899–3906. IEEE, 2009.

[8] Xieyuanli Chen, Andres Milioto, Emanuele Palazzolo, Philippe Giguère, Jens Behley, and Cyrill Stachniss. SuMa++: Efficient LiDAR-based Semantic SLAM. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[9] Konrad P. Cop, Paulo V.K. Borges, and Renaud Dubé. Delight: An efficient descriptor for global localisation using lidar intensities. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[10] Andrei Cramariuc, Renaud Dubé, Hannes Sommer, Roland Siegwart, and Igor Gilitschenski. Learning 3D Segment Descriptors for Place Recognition. *arXiv preprint*, 2018.

[11] Mark Cummins and Paul Newman. Highly scalable appearance-only SLAM - FAB-MAP 2.0. In *Proc. of Robotics: Science and Systems (RSS)*, 2009.

[12] Renaud Dubé, Daniel Dugas, Elena Stumm, Juan Nieto, Roland Siegwart, and Cesar Cadena. SegMatch: Segment Based Place Recognition in 3D Point Clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2017.

[13] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012.

[14] Jiadong Guo, Paulo V.K. Borges, Chanoh Park, and Abel Gawel. Local descriptor for robust place recognition using LiDAR intensity. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):1470–1477, 2019.

[15] Li He, Xiaolong Wang, and Hong Zhang. M2DP: A Novel 3D Point Cloud Descriptor and Its Application in Loop Closure Detection. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016.

[16] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-Time Loop Closure in 2D LIDAR SLAM. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2016.

[17] Peter J. Huber. *Robust Statistics*. Wiley, 1981.

[18] Mushtaq Hussain and James Bethel. Project and mission planing. In Chris McGlone, Edward Mikhail, James Bethel, and Roy Mullen, editors, *Manual of Photogrammetry*, chapter 15.1.2.6, pages 1109–1111. American Society for Photogrammetry and Remote Sensing, 2004.

[19] Giseop Kim, Byungjae Park, and Ayoung Kim. 1-day learning, 1-year localization: Long-term LiDAR localization using scan context image. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):1948–1955, 2019.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):8490, May 2017. ISSN 0001-0782. doi: 10.1145/3065386.

[21] Stephanie Lowry, Niko Sünderhauf, Paul Newman, John J Leonard, David Cox, Peter Corke, and Michael J Milford. Visual place recognition: A survey. *IEEE Trans. on Robotics (TRO)*, 32(1):1–19, 2016. ISSN 1552-3098. doi: 10.1109/TRO.2015.2496823.

[22] Weixin Lu, Yao Zhou, Guowei Wan, Shenhua Hou, and Shiyu Song. L3-Net: Towards Learning Based LiDAR Localization for Autonomous Driving. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[23] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[24] Sei Nagashima, Koichi Ito, Takafumi Aoki, Hideaki Ishii, and Koji Kobayashi. A high-accuracy rotation estimation algorithm based on 1D phase-only correlation. In *Proc. of the Intl. Conf. on Image Analysis and Recognition*, pages 210–221, 2007.

[25] Gaurav Pandey, James R. McBride, and Ryan M. Eustice. Ford campus vision and lidar data set. *Intl. Journal of Robotics Research (IJRR)*, 30(13):1543–1552, 2011.

[26] Timo Röhling, Jennifer Mack, and Dirk Schulz. A Fast Histogram-Based Similarity Measure for Detecting Loop Closures in 3-D LIDAR Data. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*,

pages 736–741, 2015.

[27] Lukas Schaupp, Mathias Bürki, Renaud Dubé, Roland Siegwart, and Cesar Cadena. OREOS: Oriented Recognition of 3D Point Clouds in Outdoor Scenarios. *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[28] Cyrill Stachniss, Dirk Hähnel, Wolfram Burgard, and Giorgio Grisetti. On Actively Closing Loops in Grid-based FastSLAM. *Advanced Robotics*, 19(10):1059–1080, 2005.

[29] Cyrill Stachniss, John J. Leonard, and Sebastian Thrun. *Springer Handbook of Robotics, 2nd edition*, chapter Chapt. 46: Simultaneous Localization and Mapping. Springer Verlag, 2016.

[30] Basitan Steder, Radu B. Rusu, Kurt Konolige, and Wolfram Burgard. NARF: 3D range image features for object recognition. In *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[31] Bastian Steder, Michael Ruhnke, Slawomir Grzonka, and Wolfram Burgard. Place Recognition in 3D Scans Using a Combination of Bag of Words and Point Feature Based Relative Pose Estimation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[32] Li Sun, Daniel Adolfsson, Martin Magnusson, Henrik Andreasson, Ingmar Posner, and Tom Duckett. Localising Faster: Efficient and precise lidar-based robot localisation in large-scale environments. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2020.

[33] Mikaela A. Uy and Gimm H. Lee. PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 4470–4479, 2018.

[34] Heng Yang, Jingnan Shi, and Luca Carlone. TEASER: Fast and Certifiable Point Cloud Registration. *arXiv preprint*, 2020.

[35] Huan Yin, Yue Wang, Xiaqing Ding, Li Tang, Shoudong Huang, and Rong Xiong. 3D LiDAR-Based Global Localization Using Siamese Neural Network. *IEEE Trans. on Intelligent Transportation Systems (TITS)*, 2019.

[36] Anestis Zaganidis, Alexandros Zerntev, Tom Duckett, and Grzegorz Cielniak. Semantically Assisted Loop Closure in SLAM Using NDT Histograms. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[37] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pages 766–782, 2016.