Article

The International Journal of Robotics Research
2020, Vol. 39(2-3) 339–355

DOI: 10.1177/0278364919863090
journals.sagepub.com/home/ijr
SAGE

SegMap: Segment-based mapping and localization using data-driven descriptors

Renaud Dubé[1,2*], Andrei Cramariuc[1*], Daniel Dugas[1], Hannes Sommer[1,2], Marcin Dymczyk[1,2], Juan Nieto[1], Roland Siegwart[1] and Cesar Cadena[1]

Abstract

Precisely estimating a robot's pose in a prior, global map is a fundamental capability for mobile robotics, e.g., autonomous driving or exploration in disaster zones. This task, however, remains challenging in unstructured, dynamic environments, where local features are not discriminative enough and global scene descriptors only provide coarse information. We therefore present SegMap: a map representation solution for localization and mapping based on the extraction of segments in 3D point clouds. Working at the level of segments offers increased invariance to view-point and local structural changes, and facilitates real-time processing of large-scale 3D data. SegMap exploits a single compact data-driven descriptor for performing multiple tasks: global localization, 3D dense map reconstruction, and semantic information extraction. The performance of SegMap is evaluated in multiple urban driving and search and rescue experiments. We show that the learned SegMap descriptor has superior segment retrieval capabilities, compared with state-of-the-art handcrafted descriptors. As a consequence, we achieve a higher localization accuracy and a 6% increase in recall over state-of-the-art handcrafted descriptors. These segment-based localizations allow us to reduce the open-loop odometry drift by up to 50%. SegMap is open-source available along with easy to run demonstrations.

Keywords

Global localization, place recognition, simultaneous localization and mapping (SLAM), LiDAR, 3D point clouds, segmentation, 3D reconstruction, convolutional neural network (CNN), auto-encoder

1. Introduction

Mapping and localization are fundamental competencies for mobile robotics and have been well-studied topics over the last couple of decades (Cadena et al., 2016). Being able to map an environment and later localize within it unlocks a multitude of applications, that include autonomous driving, rescue robotics, service robotics, warehouse automation, or automated goods delivery, to name a few. Robotic technologies undoubtedly have the potential to disrupt those applications within the next few years. In order to allow for the successful deployment of autonomous robotic systems in such real-world environments, several challenges need to be overcome: mapping, localization, and navigation in difficult conditions, for example crowded urban spaces, tight indoor areas or harsh natural environments. Reliable, prior-free global localization lies at the core of this challenge. Knowing the precise pose of a robot is necessary to guarantee reliable, robust, and most importantly safe operation of mobile platforms and also allows for multi-agent collaborations.

The problem of mapping and global localization has been well covered by the research community. On the one hand, a large body of algorithms use cameras and visual cues to perform place recognition. Relying purely on appearance has, however, significant limitations. In spite of tremendous progress within this field, state-of-the-art algorithms still struggle with changing seasons, weather, or even day–night variations (Lowry et al., 2016). On the other hand, several approaches address the variability of

[1]Autonomous Systems Lab (ASL), ETH Zurich, Switzerland
[2]Sevensense Robotics AG, Zurich, Switzerland
*Equal contribution

Corresponding authors:
Renaud Dubé, Sevensense Robotics AG, Weinbergstrasse 35, 8092 Zurich, Switzerland.
Email: renaud.dube@sevensense.ch

Andrei Cramariuc, ETH Zurich, Leonhardstrasse 21, 8092 Zurich, Switzerland.
Email: andrei.cramariuc@mavt.ethz.ch

appearance by relying instead on the 3D structure extracted from LiDAR data, which is expected to be more consistent across the aforementioned changes. Current LiDAR-based simultaneous localization and mapping (SLAM) systems, however, mostly use the 3D structure for local odometry estimation and map tracking, but fail to perform global localization without any prior on the pose of the robot (Hess et al., 2016).

There exist several approaches that propose to use 3D point clouds for global place recognition. Some of them make use of various local features (Rusu et al., 2009; Salti et al., 2014), which permit correspondences to be established between a query scan and a map and subsequently estimate a 6-degree-of-freedom (DoF) pose. The performance of those systems is limited, as local features are often not discriminative enough and not repeatable given the changes in the environment. Consequently, matching them is not always reliable and also incurs a large computational cost given the number of processed features. Another group of approaches relies on global descriptors of 3D LiDAR scans (Yin et al., 2018) that permit a correspondence to be found in the map. Global descriptors, however, are view-point dependent, especially when designed for only rotational-invariance and not as translation-invariant. Furthermore, a global scan descriptor is more prone to failures under dynamic scenes, e.g., parked cars, which can be important for reliable global localization in crowded, urban scenarios.

We therefore present *SegMap*:[1] a unified approach for *map representation* in the localization and mapping problem for 3D LiDAR point clouds. *SegMap* is formed on the basis of partitioning point clouds into sets of descriptive segments (Dubé et al., 2017a), as illustrated in Figure 2. The segment-based localization combines the advantages of global scan descriptors and local features: it offers reliable matching of segments and delivers accurate 6-DoF global localizations in real-time. The 3D segments are obtained using efficient region-growing techniques that are able to repeatedly form similar partitions of the point clouds (Dubé et al., 2018b). This partitioning provides the means for compact, yet discriminative features to efficiently represent the environment. During localization global data associations are identified by segment descriptor retrieval, leveraging the repeatable and descriptive nature of segment-based features. This helps satisfy strict computational, memory, and bandwidth constraints, and therefore makes the approach appropriate for real-time use in both multi-robot and long-term applications.

Previous work on segment-based localization considered hand-crafted features and provided only a sparse representation (Dubé et al., 2017a; Tinchev et al., 2018). These features lack the ability to generalize to different environments and offer very limited insights into the underlying 3D structure. In this work, we overcome these shortcomings by introducing a novel data-driven segment descriptor which offers high retrieval performance, even under variations in viewpoint, and that generalizes well to unseen environments. Moreover, as segments typically represent meaningful and distinct elements that make up the environment, a scene can be effectively summarized by only a handful of descriptors. The resulting reconstructions, as depicted in Figure 1, can be built at no extra cost in descriptor computation or bandwidth usage. They can be used by robots for navigating around obstacles and visualized to improve situational awareness of remote operators. Moreover, we show that semantic labeling can be executed through classification in the descriptor space. This information can, for example, lead to increased robustness to changes in the environment by rejecting inherently dynamic classes.

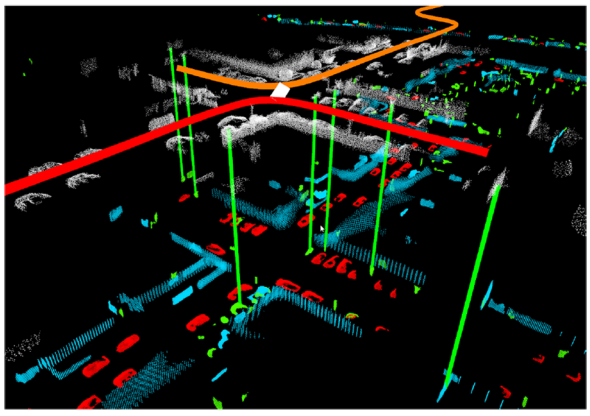

Fig. 1. An illustration of the *SegMap* approach. The red and orange paths represent the trajectories of two robots driving simultaneously in opposite directions through an intersection. In white, we show the local segments extracted from the robots' vicinity and characterized using our compact data-driven descriptor. Correspondences are then made with the target segments, resulting in a successful localization depicted with green vertical lines. A reconstruction of the target segments is illustrated below, where colors represent semantic information (cars in red, buildings in light blue, and others in green), all possible by leveraging the same compact representation. We take advantage of the semantic information by performing localization only against static objects, improving robustness against dynamic changes. Both the reconstruction and semantic classification are computed by leveraging the same descriptors used for global prior-free localization.

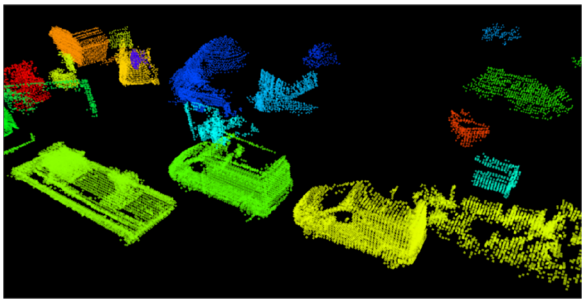

Fig. 2. Exemplary segments extracted from 3D LiDAR data collected in a rural environment. These segments were extracted with an incremental Euclidean distance-based region-growing algorithm and represent, among others, vehicles, vegetation, and parts of buildings (Dubé et al., 2018b).

To the best of the authors' knowledge, this is the first work on robot localization that is able to leverage the extracted features for reconstructing environments in three dimensions and for retrieving semantic information. This reconstruction is, in our opinion, a very interesting capability for real-world, large-scale applications with limited memory and communication bandwidth. To summarize, this paper presents the following contributions.

- A data-driven 3D segment descriptor that improves localization performance.
- A novel technique for reconstructing the environment based on the same compact features used for localization.
- An extensive evaluation of the *SegMap* approach using real-world, multi-robot automotive and disaster scenario datasets.

In relation to the *Robotics: Science and System* conference paper (Dubé et al., 2018a), we make the following additional contributions.

- A comparison of the accuracy of our localization output with the results of recently published techniques based on data-driven global 3D scan descriptors (Yin et al., 2018).
- An evaluation of trajectory estimates by combining our place recognition approach with a state-of-the-art 3D LiDAR-based SLAM technique (Zhang and Singh, 2014).
- A triplet loss descriptor training technique and its comparison with the previously introduced classification-based approach.
- A particularly lightweight variant of our *SegMap* descriptor that can be deployed on platforms with limited computational resources.

The remainder of the paper is structured as follows. Section 2 provides an overview of the related work in the fields of localization and learning-based descriptors for 3D point clouds. The *SegMap* approach and our novel descriptor that enables reconstruction of the environment are detailed in Sections 3 and 4. The method is evaluated in Section 5, and finally Sections 6 and 7 conclude with a short discussion and ideas on future work.

2. Related work

This section first introduces state-of-the-art approaches to localization in 3D point clouds. Data-driven techniques using 3D data that are relevant to the present work are then presented.

2.1. Localization in 3D point clouds

Detecting loop-closures from 3D data has been tackled with different approaches. We have identified three main trends: (i) approaches based on local features, (ii) approaches based on global descriptors, and (iii) approaches based on planes or objects.

A significant number of works propose to extract local features from keypoints and perform matching on the basis of these features. Bosse and Zlot (2013) extracted keypoints directly from the point clouds and described them with a 3D *Gestalt* descriptor. Keypoints then vote for their nearest neighbors in a *vote matrix*, which is eventually thresholded for recognizing places. A similar approach has been used by Gawel et al. (2016). Apart from such Gestalt descriptors, a number of alternative local feature descriptors exist, which can be used in similar frameworks. This includes features such as fast point feature histogram (FPFH) (Rusu et al., 2009) and SHOT Salti et al. (2014). Alternatively, Zhuang et al. (2013) transformed the local scans into bearing-angle images and extract speeded up robust features (SURFs) from these images. A strategy based on 3D spatial information was employed to order the scenes before matching the descriptors. A similar technique by Steder et al. (2010) first transforms the local scans into a range image. Local features are extracted and compared with those stored in a database, employing the Euclidean distance for matching keypoints. This work was extended by Steder et al. (2011) by using normal-aligned radial features (NARF) descriptors and a bag-of-words approach for matching.

Using global descriptors of the local point cloud for place recognition was also proposed in Röhling et al. (2015), Granström et al. (2011), Magnusson et al. (2009), and Cop et al. (2018). Röhling et al. (2015) proposed describing each local point cloud with a 1D histogram of point heights, assuming that the sensor keeps a constant height above the ground. The histograms were then compared using the *Wasserstein* metric for recognizing places. Granström et al. (2011) described point clouds with rotation invariant features such as volume, nominal range, and range histogram. Distances are computed for feature vectors and cross-correlation for histogram features, and an AdaBoost classifier is trained to match places. Finally, iterative closest point (ICP) is used for computing the relative pose between point clouds. In another approach, Magnusson et al. (2009) split the cloud into overlapping grids and compute shape properties (spherical, linear, and several type of planar) of each cell and combine them into a matrix of surface shape histograms. Similar to other works, these descriptors are compared for recognizing places. Recently, Cop et al. (2018) proposed to leverage LiDAR intensity information with a global point cloud descriptor. A two-stage approach is adopted such that, after retrieving places based on global descriptors retrieval, a local keypoint-based geometric verification step estimates localization transformations. The authors demonstrated that using intensity information can reduce the computational timings. However, the complete localization pipeline operates at a frequency one order of magnitude lower than most LiDAR sensor frequencies.

Whereas local keypoint features often lack descriptive power, global descriptors can struggle with variations in view-point. Therefore, other works have also proposed to

use 3D shapes or objects for the place recognition task. Fernández-Moral et al. (2013), for example, proposed to perform place recognition by detecting planes in 3D environments. The planes are accumulated in a graph and an interpretation tree is used to match sub-graphs. A final geometric consistency test is conducted over the planes in the matched sub-graphs. The work was extended in Fernández-Moral et al. (2016) to use the covariance of the plane parameters instead of the number of points in planes for matching. This strategy was only applied to small, indoor environments and assumed a plane model that is no longer valid in unstructured environments. A somewhat analogous, seminal work on object-based loop-closure detection in indoor environments using RGB-D cameras is presented by Finman et al. (2015). Although presenting interesting ideas, their work can only handle a small number of well segmented objects in small-scale environments. Similarly, Bowman et al. (2017) proposed a novel SLAM solution in which semantic information and local geometric features are jointly incorporated into a probabilistic framework. Such semantic-based approaches have significant potential, for example robustness to stark changes in point of view, but require the presence of human-known objects in the scene.

We therefore aim for an approach that does not rely on assumptions about the environment being composed of simplistic geometric primitives such as planes, or a rich library of objects. This allows for a more general, scalable solution.

2.2. Learning with 3D point clouds

In recent years, convolutional neural networks (CNNs) have become the state-of-the-art method for generating learning-based descriptors, owing to their ability to find complex patterns in data (Krizhevsky et al., 2012). For 3D point clouds, methods based on CNNs achieve impressive performance in applications such as object detection (Engelcke et al., 2017; Fang et al., 2015; Li et al., 2016; Maturana and Scherer, 2015; Qi et al., 2017; Riegler et al., 2017; Wohlhart and Lepetit, 2015; Wu et al., 2015), semantic segmentation (Graham et al., 2018; Li et al., 2016; Qi et al., 2017; Riegler et al., 2017; Tchapmi et al., 2017; Wu et al., 2018), and 3D object generation (Wu et al., 2016), and LiDAR-based local motion estimation (Dewan et al., 2018; Velas et al., 2018).

Recently, a handful of works proposing the use of CNNs for localization in 3D point clouds have been published. First, Zeng et al. (2017) proposed extracting data-driven 3D keypoint descriptors (3DMatch) that are robust to changes in view-point. Although impressive retrieval performance is demonstrated using an RGB-D sensor in indoor environments, it is not clear whether this method is applicable in real-time in large-scale outdoor environments. A different approach based on 3D CNNs was proposed by Ye et al. (2017) for performing localization in semi-dense maps generated with visual data. Recently, Yin et al. (2017) introduced a semi-handcrafted global descriptor for performing place recognition and rely on an ICP step for estimating the 6-DoF localization transformations. This method will be used as a baseline solution in Section 5.8 when evaluating the precision of our localization transformations. Elbaz et al. (2017) proposed describing local subsets of points using a deep neural network autoencoder. The authors stated, however, that the implementation has not been optimized for real-time operation and no timings have been provided. In contrast, our work presents a data-driven segment-based localization method that can operate in real-time and that enables map reconstruction and semantic extraction capabilities.

To achieve this reconstruction capability, the architecture of our descriptor was inspired by autoencoders in which an encoder network compresses the input to a small dimensional representation, and a decoder network attempts to decompress the representation back into the original input. The compressed representation can be used as a descriptor for performing 3D object classification (Brock et al., 2016). Brock et al. (2016) also present successful results using variational autoencoders for reconstructing voxelized 3D data. Different configurations of encoding and decoding networks have also been proposed for achieving localization and for reconstructing and completing 3D shapes and environments (Dai et al., 2017; Elbaz et al., 2017; Guizilini and Ramos, 2017; Ricao Canelhas et al., 2017; Schönberger et al., 2018; Varley et al., 2017).

While autoencoders present the interesting opportunity of simultaneously accomplishing both compression and feature extraction tasks, optimal performance at both is not guaranteed. As will be shown in Section 5.4, these two tasks can have conflicting goals when robustness to changes in point of view is desired. In this work, we combine the advantages of the encoding–decoding architecture of autoencoders with a technique proposed by Parkhi et al. (2015). The authors address the face recognition problem by first training a CNN to classify people in a training set and afterwards use the penultimate layer as a descriptor for new faces. Other alternative training techniques include, for example, the use of contrastive loss (Bromley et al., 1994) or triplet loss (Weinberger et al., 2006), the latter being evaluated in Section 5.4. We use the resulting segment descriptors in the context of SLAM to achieve better performance, as well as significantly compressed maps that can easily be stored, shared, and reconstructed.

3. The *SegMap* approach

This section presents our *SegMap* approach to localization and mapping in 3D point clouds. It is composed of five core modules: segment extraction, description, localization, map reconstruction, and semantics extraction. These modules are detailed in this section and together allow single- and multi-robot systems to create a powerful unified representation that can conveniently be transferred.

3.1. Segmentation

The stream of point clouds generated by a 3D sensor is first accumulated in a dynamic voxel grid.[2] Point cloud segments are then extracted in a section of radius R around the robot. In this work, we consider two types of incremental segmentation algorithms (Dubé et al., 2018b). The first starts by removing points corresponding to the ground plane, which acts as a separator for clustering together the remaining points based on their Euclidean distances. The second algorithm computes local normals and curvatures for each point and uses these to extract flat or planar-like surfaces. Both methods are used to incrementally grow segments by using only newly active voxels as seeds that are either added to existing segments, form new segments, or merge existing segments together.[3] This results in a handful of local segments, which are individually associated with a set of past observations i.e., $S_i = \{s_1, s_2, \ldots, s_n\}$. Each observation $s_j \in S_i$ is a 3D point cloud representing a snapshot of the segment as points are added to it. Note that s_n represents the latest observation of a segment and is considered *complete* when no further measurements are collected, e.g., when the robot has moved away.

3.2. Description

Compact features are then extracted from these 3D segment point clouds using the data-driven descriptor presented in Section 4. A global segment map is created online by accumulating the segment centroids and corresponding descriptors. In order for the global map to most accurately represent the latest state of the world, we only keep the descriptor associated with the last and most complete observation.

3.3. Localization

In the next step, candidate correspondences are identified between global and local segments using k-nearest neighbors (k-NN) in feature space. The approximate k nearest descriptors are retrieved through an efficient query in a kd-tree. Localization is finally performed by verifying the largest subset of candidate correspondences for geometrical consistency on the basis of the segment centroids. Specifically, the centroids of the corresponding local and global segments must have the same geometric configuration up to a small jitter in their position, to compensate for slight variations in segmentation. In the experiments presented in Section 5.9, this is achieved using an incremental recognition strategy which uses caching of correspondences for faster geometric verifications (Dubé et al., 2018b).

When a large enough geometrically consistent set of correspondence is identified, a 6-DoF transformation between the local and global maps is estimated. This transformation is fed to an incremental pose-graph SLAM solver which in turn estimates, in real-time, the trajectories of all robots (Dubé et al., 2017b).

3.4. Reconstruction

Thanks to our autoencoder-like descriptor extractor architecture, the compressed representation can at any time be used to reconstruct an approximate map as illustrated in Figure 12. As the *SegMap* descriptor can conveniently be transmitted over wireless networks with limited bandwidth, any agent in the network can reconstruct and leverage this 3D information. More details on these reconstruction capabilities are given in Section 4.3.

3.5. Semantics

The *SegMap* descriptor also contains semantically relevant information without the training process having enforced this property on the descriptor. This can, for example, be used to discern between static and dynamic objects in the environment to improve the robustness of the localization task. In this work, we present an experiment where the network is able to distinguish between three different semantic classes: *vehicles, buildings*, and *others* (see Section 4.4).

4. The *SegMap* descriptor

In this section, we present our main contribution: a data-driven descriptor for 3D segment point clouds that allows for localization, map reconstruction, and semantic extraction. The descriptor extractor's architecture and the processing steps for inputting the point clouds to the network are introduced. We then describe our technique for training this descriptor to accomplish tasks of both segment retrieval and map reconstruction. We finally show how the descriptor can further be used to extract semantic information from the point cloud.

4.1. Descriptor extractor architecture

The architecture of the descriptor extractor is presented in Figure 3. Its input is a 3D binary voxel grid of fixed dimension $32 \times 32 \times 16$ that was determined empirically to offer a good balance between descriptiveness and the size of the network. The description part of the CNN is composed of three 3D convolutional layers with max pool layers placed in between and two fully connected layers. Unless otherwise specified, rectified linear unit (ReLU) activation functions are used for all layers. The original scale of the input segment is passed as an additional parameter to the first fully connected layer to increase robustness to voxelization at different aspect ratios. The descriptor is obtained by taking the activations of the extractor's last fully connected layer. This architecture was selected by grid search over various configurations and parameters.

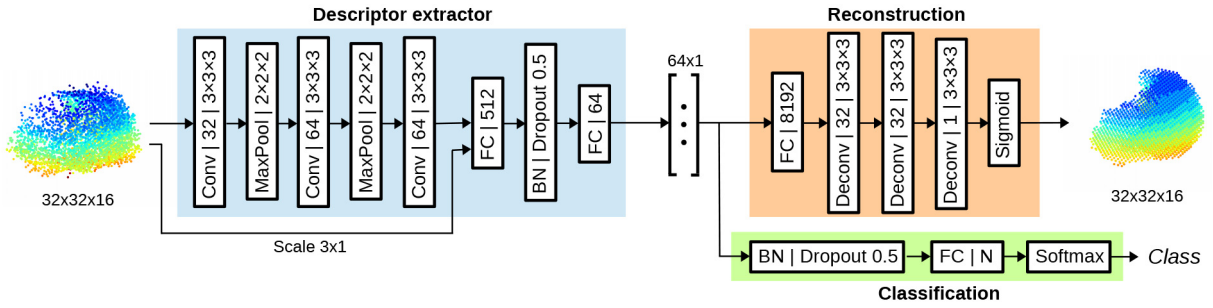

Fig. 3. The descriptor extractor is composed of three convolutional and two fully connected layers. The 3D segments are compressed to a representation of dimension 64×1, which can be used for localization, map reconstruction, and semantic extraction. Right of the descriptor we illustrate the classification and reconstruction layers that are used for training. In the diagram the convolutional (Conv), deconvolutional (Deconv), fully connected (FC), and batch normalization (BN) layers are abbreviated, respectively. As parameters the Conv and Deconv layers have the number of filters and their sizes, FC layers have the number of nodes, max pool layers have the size of the pooling operation, and dropout layers have the ratio of values to drop. Unless otherwise specified, ReLU activation functions are used for all layers.

4.2. Segment alignment and scaling

A pre-processing stage is required in order to input the 3D segment point clouds for description. First, an alignment step is applied such that segments extracted from the same objects are similarly presented to the descriptor network. This is performed by applying a 2D principal components analysis (PCA) of all points located within a segment. The segment is then rotated so that the x-axis of its frame of reference, from the robot's perspective, aligns with the eigenvector corresponding to the largest eigenvalue. We choose to solve the ambiguity in direction by rotating the segment so that the lower half section along the y-axis of its frame of reference contains the highest number of points. From the multiple alignment strategies we evaluated, the presented strategy worked best.

The network's input voxel grid is applied to the segment so that its center corresponds to the centroid of the aligned segment. By default the voxels have minimum side lengths of 0.1 m. These can individually be increased to exactly fit segments having one or more larger dimension than the grid. Whereas maintaining the aspect ratio while scaling can potentially offer better retrieval performance, this individual scaling with a minimum side length better avoids large errors caused by aliasing. We also found that this scaling method offers the best reconstruction performance, with only a minimal impact on the retrieval performance when the original scale of the segments is passed as a parameter to the network.

4.3. Training the SegMap descriptor

In order to achieve both a high retrieval performance and reconstruction capabilities, we propose a customized learning technique. The two desired objectives are imposed on the network by the *softmax cross entropy loss* L_c for retrieval and the reconstruction loss L_r. We propose to simultaneously apply both losses to the descriptor and to this end define a combined loss function L which merges the contributions of both objectives:

$$L = L_c + \alpha L_r \tag{1}$$

where the parameter α weighs the relative importance of the two losses. The value $\alpha = 200$ was empirically found to not significantly affect the performance of the combined network, as opposed to training separately with either of the losses. Weights are initialized based on Xavier's initialization method (Glorot and Bengio, 2010) and trained using the adaptive moment estimation (ADAM) optimizer (Kingma and Ba, 2015) with a learning rate of 10^{-4}. In comparison to stochastic gradient descent (SGD), ADAM maintains separate learning rates for each network parameter, which facilitates training the network with two separate objectives simultaneously. Regularization is achieved using dropout (Srivastava et al., 2014) and batch normalization (Ioffe and Szegedy, 2015).

4.3.1. Classification loss L_c. For training the descriptor to achieve better retrieval performance, we use a learning technique similar to the *N-ways classification problem* proposed by Parkhi et al. (2015). Specifically, we organize the training data into N classes where each class contains all observations of a segment or of multiple segments that belong to the same object or environment part. Note that these classes are solely used for training the descriptor and are not related to the semantics presented in Section 4.4. As seen in Figure 3, we then append a classification layer to the descriptor and teach the network to associate a score to each of the N predictors for each segment sample. These scores are compared with the true class labels using *softmax cross entropy loss*:

$$L_c = -\sum_{i=1}^{N} y_i \log \frac{e^{l_i}}{\sum_{k=1}^{N} e^{l_k}} \tag{2}$$

where y is the one hot encoded vector of the true class labels and l is the layer output.

Given a large number of classes and a small descriptor dimensionality, the network is forced to learn descriptors that better generalize and prevent overfitting to specific segment samples. Note that when deploying the system in a new environment the classification layer is removed, as its output is no longer relevant. The activations of the previous fully connected layer are then used as a descriptor for segment retrieval through k-NN.

4.3.2. Reconstruction loss L_r. As depicted in Figure 3, map reconstruction is achieved by appending a decoder network and training it simultaneously with the descriptor extractor and classification layer. This decoder is composed of one fully connected and three deconvolutional layers with a final sigmoid output. Note that no weights are shared between the descriptor and the decoder networks. Furthermore, only the descriptor extraction needs to be run in real-time on the robotic platforms, whereas the decoding part can be executed any time a reconstruction is desired.

As proposed by Brock et al. (2016), we use a specialized form of the *binary cross entropy loss*, which we denote by L_r:

$$L_r = -\sum_{x,y,z} (\gamma t_{xyz} \log(o_{xyz}) + (1-\gamma)(1-t_{xyz}) \log(1-o_{xyz})) \quad (3)$$

where t and o represent the target segment and the network's output, respectively, and γ is a hyperparameter that weighs the relative importance of false positives and false negatives. This parameter addresses the fact that only a minority of voxels are activated in the voxel grid. In our experiments, the voxel grids used for training were on average only 3% occupied and we found $\gamma = 0.9$ to yield good results.

4.4. Knowledge transfer for semantic extraction

As can be observed from Figure 1, segments extracted by the *SegMap* approach for localization and map reconstruction often represent objects or parts of objects. It is therefore possible to assign semantic labels to these segments and use this information to improve the performance of the localization process. As depicted in Figure 4, we transfer the knowledge embedded in our compact descriptor by training a semantic extraction network on top of it. This last network is trained with labeled data using the *softmax cross entropy loss* and by freezing the weights of the descriptor network.

In this work, we choose to train this network to distinguish between three different semantic classes: *vehicles, buildings*, and *others*. Section 5.9 shows that this information can be used to increase the robustness of the localization algorithm to changes in the environment and to yield smaller map sizes. This is achieved by rejecting segments associated with potentially dynamic objects, such as vehicles, from the list of segment candidates. This video demonstrates the performance of the SegMap approach with a multi-robot SLAM scenario in urban driving environment.

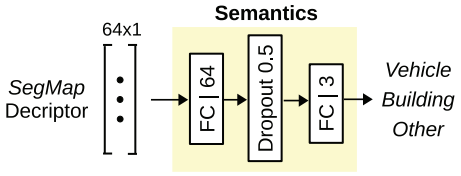

Fig. 4. A simple fully connected network that can be appended to the *SegMap* descriptor (depicted in Figure 3) in order to extract semantic information. In our experiments, we train this network to distinguish between vehicles, buildings, and other objects.

4.5. SegMini

Finally, we propose a lightweight version of the *SegMap* descriptor that is specifically tailored for resource-limited platforms. *SegMini* has the same architecture as *SegMap* (see Figure 3), with the exception that the number of filter in the convolutional layers and the size of the dense layers is halved. Without compromising much on the descriptor retrieval performance this model leads to a computational speedup of a factor of two for GPU and a factor of six for CPU (Section 5.3).

5. Experiments

This section presents the experimental validation of our approach. We first present a procedure for generating training data and detail the performance of the *SegMap* descriptor for localization, reconstruction, and semantics extraction. We finally evaluate the complete *SegMap* solution in multiple real-world experiments.

5.1. Experiment setup and implementation

All experiments were performed on a system equipped with an Intel i7-6700K processor, and an NVIDIA GeForce GTX 980 Ti GPU. The CNN models were developed and executed in real-time using the TensorFlow library. The *libnabo* library is used for descriptor retrieval with fast k-NN search in low-dimensional space (Elseberg et al., 2012). The incremental optimization back-end is based on the iSAM2 implementation from Kaess et al. (2012).

5.2. Training data

The *SegMap* descriptor is trained using real-world data from the KITTI odometry dataset (Geiger et al., 2012). Sequences 05 and 06 are used for generating training and testing data, whereas sequence 00 is solely used for validation of the descriptor performance. In addition, end-to-end

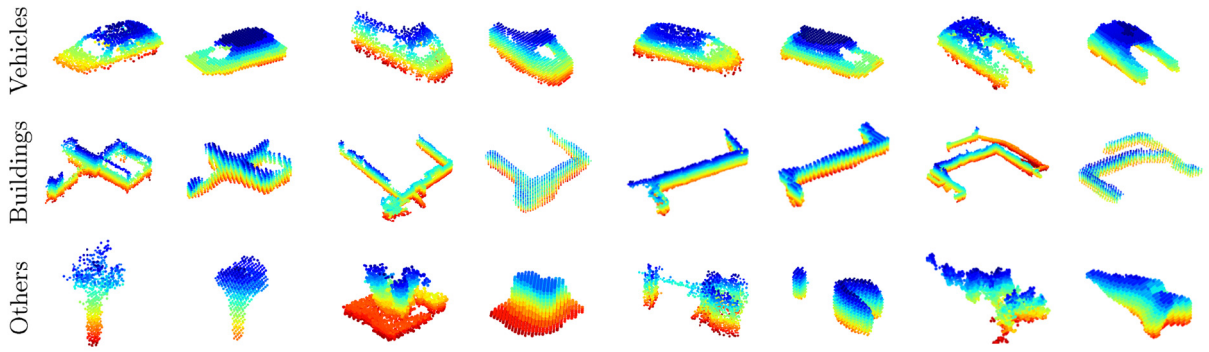

Fig. 5. An illustration of the *SegMap* reconstruction capabilities. The segments are extracted from sequence 00 of the KITTI dataset and represent, from top to bottom, respectively, vehicles, buildings, and other objects. For each segment pair, the reconstruction is shown to the right of the original. The network manages to accurately reconstruct the segments despite the high compression to only 64 values. Note that the voxelization effect is more visible on buildings as larger segments necessitate larger voxels to keep the input dimension fixed.

experiments are performed using sequences 00 and 08, as they feature long tracks with multiple overlapping areas in the trajectories. For each sequence, segments are extracted using an incremental Euclidean distance-based region growing technique (Dubé et al., 2018b). This algorithm extracts point clouds representing parts of objects or buildings which are separated after removing the ground plane (see Figure 5). The training data is filtered by removing segments with too few observations, or training classes (as described in Section 4.3) with too few samples. In this manner, 3,300, 1,750, 810, and 2,400 segments are generated from sequences 00, 05, 06 and 08, respectively, with an average of 12 observations per segment over the whole dataset.

5.2.1. Data augmentation. To further increase robustness by reducing sensitivity to rotation and view-point changes in the descriptor extraction process, the dataset is augmented through various transformations at the beginning of each training epoch. Each segment is rotated at different angles to the alignment described in Section 4.2 to simulate different view-points. In order to simulate the effect of occlusion for each segment we remove all points that fall on one side of a randomly generated slicing plane that does not remove more than 50% of the points. Finally, random noise is simulated by randomly removing up to 10% of the points in the segment. Note that these two data augmentation steps are performed prior to voxelization.

5.2.2. Ground-truth generation. In the following step, we use GPS readings in order to identify ground-truth correspondences between segments extracted in areas where the vehicle performed multiple visits. Only segment pairs with a maximum distance between their centroids of 3.0 m are considered. We compute the 3D convex hull of each segment observation s_1 and s_2 and create a correspondence when the following condition, inspired from the Jaccard index, holds:

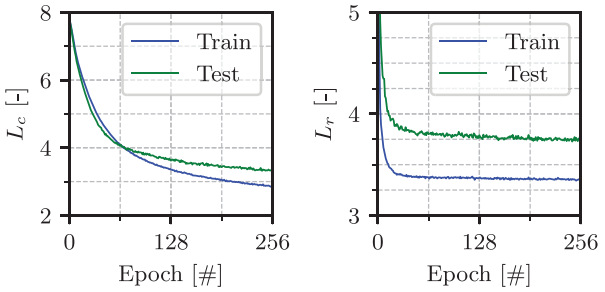

Fig. 6. The classification loss L_c (left) and the reconstruction loss L_r (right) components of the total loss L, when training the descriptor extractor along with the reconstruction and classification networks. The depicted reconstruction loss has already been scaled by α.

$$\frac{\text{Volume}(\text{Conv}(s_1) \cap \text{Conv}(s_2))}{\text{Volume}(\text{Conv}(s_1) \cup \text{Conv}(s_2))} \geqslant p \qquad (4)$$

In our experiments we found $p = 0.3$ to generate a sufficient number of correspondences while preventing false labeling. The procedure is performed on sequences 00, 05, and 06, generating 150, 260, and 320 ground-truth correspondences, respectively. We use two-thirds of the correspondences for augmenting the training data and one-third for creating validation samples. Finally, the ground-truth correspondences extracted from sequence 00 are used in Section 5.4 for evaluating the retrieval performance.

5.3. Training the models

The descriptor extractor and the decoding part of the reconstruction network are trained using all segments extracted from drive 05 and 06. Training lasts 3–4 h on the GPU and produces the classification and scaled reconstruction losses depicted in Figure 6. The total loss of the model is the sum

of the two losses as described in Section 4.3. We note that for classification the validation loss follows the training loss before converging towards a corresponding accuracy of 41% and 43%, respectively. In other words, 41% of the validation samples were correctly assigned to one of the $N = 2,500$ classes. This accuracy is expected given the large quantity of classes and the challenging task of discerning between multiple training samples with similar semantic meaning, but few distinctive features, e.g., flat walls. Note that we achieve a very similar classification loss L_c, when training with and without the L_r component of the combines loss L. On a GPU the *SegMap* descriptor takes on average 0.8 ms to compute, while the *SegMini* descriptor takes 0.3 ms. On the CPU the performance gain is more significant, as it takes 245 ms for a *SegMap* descriptor as opposed to only 41 ms for *SegMini*, which is a factor of six improvement in efficiency.

5.4. Descriptor retrieval performance

We evaluate the retrieval performance of the *SegMap* descriptor against state-of-the-art methods as well as other networks trained with different secondary goals. First, our descriptor is compared with eigenvalue-based point cloud features (Weinmann et al., 2014). We also evaluate the effect of training only for the classification task (Classification) or of training only for the reconstruction one (Autoencoder). In addition, we compare classification-based learning with a triplet loss solution (Schroff et al., 2015), where during training, we enforce segments from the same sequence to have a minimal Euclidean distance. We use a per batch hard mining strategy and the best performing variant of triplet loss as proposed by Hermans et al. (2017). We finally evaluate the *SegMini* model introduced in Section 4.5.

The retrieval performance of the aforementioned descriptors is depicted in Figure 7. The receiver operating characteristic (ROC) curves are obtained by generating 45,000,000 labeled pairs of segment descriptors from sequence 00 of the KITTI odometry dataset (Geiger et al., 2012). Using ground-truth correspondences, a positive sample is created for each possible segment observation pair. For each positive sample, a thousand negative samples are generated by randomly sampling segment pairs whose centroids are further than 20 m apart. The positive to negative sample ratio is representative of our localization problem given that a map created from KITTI sequence 00 contains around 1,000 segments. The ROC curves are finally obtained by varying the threshold applied on the L^2 norm between the two segment descriptors. We note that training with triplet loss offers the best ROC performance on these datasets, as it imposes the most consistent separation margin across all segments.

The ROC is not the best evaluation metric for this retrieval task, because it evaluates the quality of classification for a single threshold across all segments. As introduced in Section 3, correspondences are made between segments

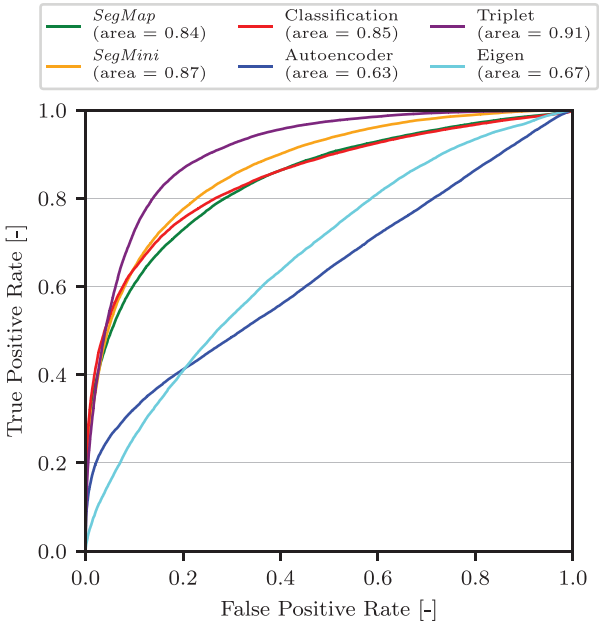

Fig. 7. ROC curves for the descriptors considered in this work. This evaluation is performed using ground-truth correspondences extracted from sequence 00 of the KITTI odometry dataset (Geiger et al., 2012). Note that the ROC is not an optimal measure of the quality of the retrieval performance, because it only considers a single threshold for all segment pairs and does not look at the relative ordering of matches on a per query basis.

from the local and global maps by using k-NN retrieval in feature space. The varying parameter is the number of neighbors that is retrieved and not a threshold on the feature distances, which only matter in a relative fashion on a per query basis. In order to avoid false localizations, the aim is to reduce the number k of neighbors that need to be considered. Therefore, as a segment grows with time, it is critical that its descriptor converges as quickly as possible towards the descriptor of the corresponding segment in the target map, which in our case is extracted from the last and most *complete* observation (see Section 3). This behavior is evaluated in Figure 8a, which relates the number of neighbors that need to be considered to find the correct association, as a function of segment completeness. We note that the *SegMap* descriptor offers competitive retrieval performance at every stage of the growing process. In practice, this is important because it allows challenging loops to be closed such as that presented in Figure 1.

Interestingly, the autoencoder has the worst performance at the early growing stages whereas good performance is observed at later stages. This is in accordance with the capacity of autoencoders to precisely describe the geometry of a segment, without explicitly aiming at gaining a robust representation in the presence of occlusions or changes in view-point. Although the triplet loss training method offers the best ROC performance, Figure 8a suggests that training with the secondary goal of classification yields

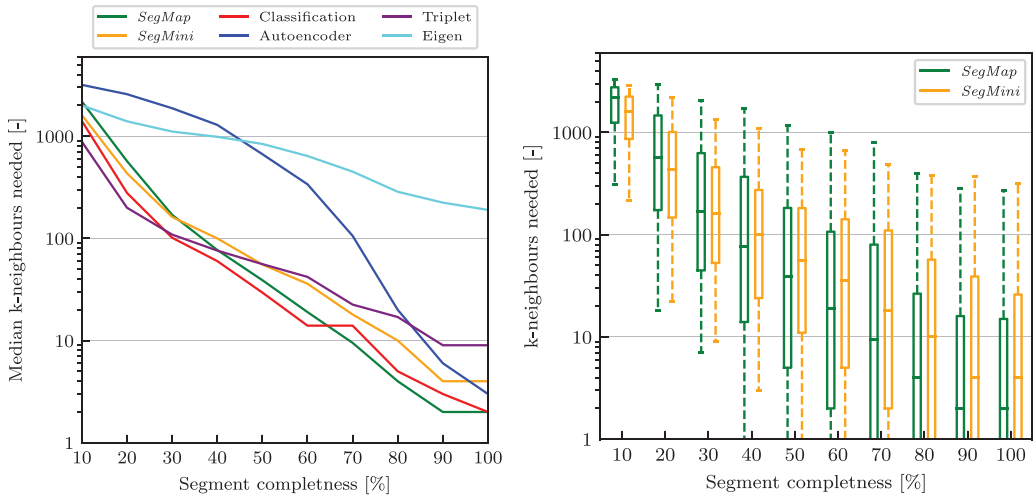

Fig. 8. How *quickly* descriptors extracted from incrementally grown segments contain relevant information that can be used for localization. The *x*-axis represents the completeness of a segment until all its measurements have been accumulated (here termed *complete*, see Section 3). In (a), the log-scaled *y*-axis represents the median of how many neighbors in the target map need to be considered in order to retrieve the correct target segment (the lower the better). Similarly (b) presents the same results in more detail for the proposed models. The *SegMap* descriptor offers over the majority of the growing process one order of magnitude better retrieval performance than the hand-crafted baseline descriptor.

considerably better results at the later stages of growing. The poor performance of the triplet loss method especially for very similar segments could be caused by the hard mining amplifying the noise in the dataset. After a certain point the ordering of matches becomes irrelevant, because the goal is to minimize the number of retrieved neighbors and retrieving too many is computationally unfeasible for later stages of the process. Therefore, although the purely classification-based model performs slightly better for very early observations of a segment, this gain in performance does not matter. The proposed *SegMap* descriptor achieves the best performance for very complete segments, where matches are most likely to happen, and maintains a comparable performance across very partial observations. A more detailed plot for the retrieval performance of the *SegMap* and *SegMini* is presented in Figure 8b, where also the variance in the retrieval accuracy is shown.

5.5. Reconstruction performance

In addition to offering high retrieval performance, the *SegMap* descriptor allows us to reconstruct 3D maps using the decoding CNN described in Section 4.3. Some examples of the resulting reconstructions are illustrated in Figure 5, for various objects captured during sequence 00 of the KITTI odometry dataset. Experiments performed at a larger scale are presented in Figure 14, where buildings of a powerplant and a foundry are reconstructed by fusing data from multiple sensors.

as most segments only sparsely model real-world surfaces, they occupy on average only 3% of the voxel grid. To obtain a visually relevant comparison metric, we calculate for both the original segment and its reconstruction the ratio of points having a corresponding point in the other segment, within a distance of one voxel. The tolerance of one voxel means that the shape of the original segment must be preserved while not focusing on reconstructing each individual point. Results calculated for different descriptor sizes are presented in Table 1, in comparison with the purely reconstruction focused baseline. The *SegMap* descriptor with a size of 64 has on average 91% correspondences between the points in the original and reconstructed segments, and is only slightly outperformed by the autoencoder baseline. Contrastingly, the significantly higher retrieval performance of the *SegMap* descriptor makes it a clear all-rounder choice for achieving both localization and map reconstruction.

Table 1. Average ratio of corresponding points within one voxel distance between original and reconstructed segments. Statistics for *SegMap* and the autoencoder baseline using different descriptor sizes.

	Descriptor size			
	16	32	64	128
Autoencoder	0.87	0.91	0.93	0.94
SegMap	0.86	0.89	0.91	0.92

Overall, the reconstructions are well recognizable despite the high compression ratio. In Figure 12, we note that the quantization error resulting from the voxelization step mostly affects larger segments that have been downscaled to fit into the voxel grid. To mitigate this problem, one can

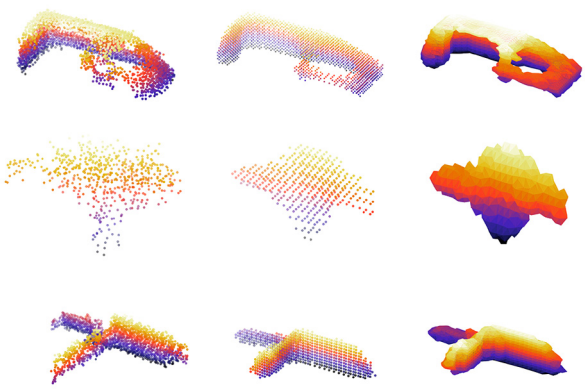

Fig. 9. A visual comparison between (left) the original point cloud, (middle) the reconstruction point cloud, and (right) the reconstruction mesh, for three segments.

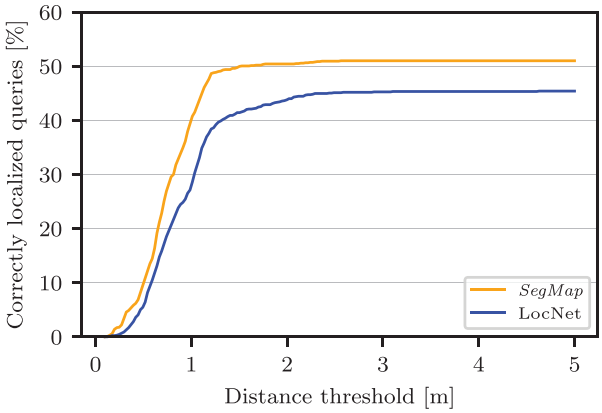

Fig. 10. Cumulative distribution of position errors on KITTI 00 odometry sequence that compares *SegMap* with state-of-the-art data-driven LocNet approach presented in Yin et al. (2018, 2017). Our proposed method retrieves a full 6-DoF pose whereas LocNet uses global scan descriptors to obtain the nearest pose of the target map. *SegMap* retrieves poses for a larger number of scans and the returned estimates are more accurate. The results saturate at about 52% as not all query positions overlap with the target map, with only 65% of them being within a radius of 50 m from the map.

adopt a natural approach to representing this information in 3D space, which is to calculate the isosurface for a given probability threshold. This can be computed using the "marching cubes" algorithm, as presented by Lorensen and Cline (1987). The result is a triangle-mesh surface, which can be used for intuitive visualization, as illustrated in Figures 9 and 12.

5.6. Semantic extraction performance

For training the semantic extractor network (Figure 4), we manually labeled the last observation of all 1,750 segments extracted from KITTI sequence 05. The labels are then propagated to each observation of a segment for a total of 20,000 labeled segment observations. We use 70% of the samples for training the network and 30% for validation. Given the low complexity of the semantic extraction network and the small amount of labeled samples, training takes only a few minutes. We achieve an accuracy of 89% and 85% on the training and validation data respectively. Note that our goal is not to improve over other semantic extraction methods (Li et al., 2016; Qi et al., 2017), but rather to illustrate that our compressed representation can additionally be used for discarding dynamic elements of the environment and for reducing the map size (Section 5.9.1).

5.7. 6-DoF pose retrieval performance

In this section, we demonstrate how the advantageous properties of *SegMap*, particularly the descriptor retrieval performance, translate to state-of-the-art global localization results. We therefore compare our approach with a global localization method, LocNet (Yin et al., 2018, 2017). It uses rotation-invariant, data-driven descriptors that yield reliable matching of 3D LiDAR scans. LocNet retrieves a nearest-neighbor database scan and returns its pose, its output is thus limited to the poses already present in the target map. Therefore, it works reliably in environments with well-defined trajectories (e.g., roads), but fails to return a precise location within large traversable areas such as squares or hallways. In contrast, *SegMap* uses segment correspondences to estimate an accurate 6-DoF pose that includes orientation, which cannot be retrieved directly using the rotation-invariant LocNet descriptors.

Figure 10 presents the evaluation of both methods on the KITTI 00 odometry sequence (4,541 scans). We use the first 3,000 LiDAR scans and their ground-truth poses to create a map, against which we then localize using the last 1,350 scans. *SegMap* demonstrates a superior performance both by successfully localizing about 6% more scans and by returning more accurate localized poses. Note that from the query positions only 65% of them were taken within a distance of 50 m of the target map, therefore limiting the maximum possible saturation. We believe that robust matching of segments, a principle of our method, helps to establish reliable correspondences with the target map, particularly for queries further away from the mapped areas. This state-of-the-art localization performance is further complemented by a compact map representation, with reconstruction and semantic labeling capabilities.

5.8. A complete mapping and localization system

So far, we have only evaluated *SegMap* as a stand-alone global localization system, demonstrating the performance of segment descriptors and the 6-DoF pose retrieval. Such global localization systems, however, are commonly used in conjunction with odometry and mapping algorithms. To prove the qualities of *SegMap* in such a scenario, we have combined it with a state-of-the-art LiDAR odometry and

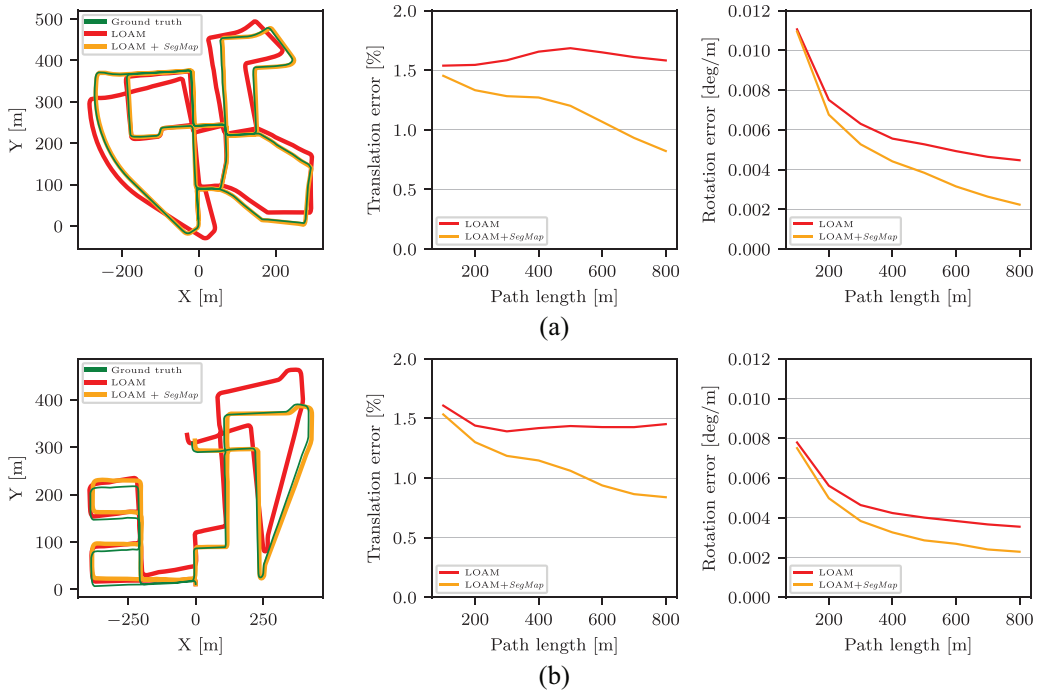

Fig. 11. The trajectories for KITTI odometry sequences (a) 00 and (b) 08 for LOAM and the combination of LOAM and *SegMap* . In addition, we show translation and rotation errors for the two approaches, using the standard KITTI evaluation method Geiger et al. (2012).

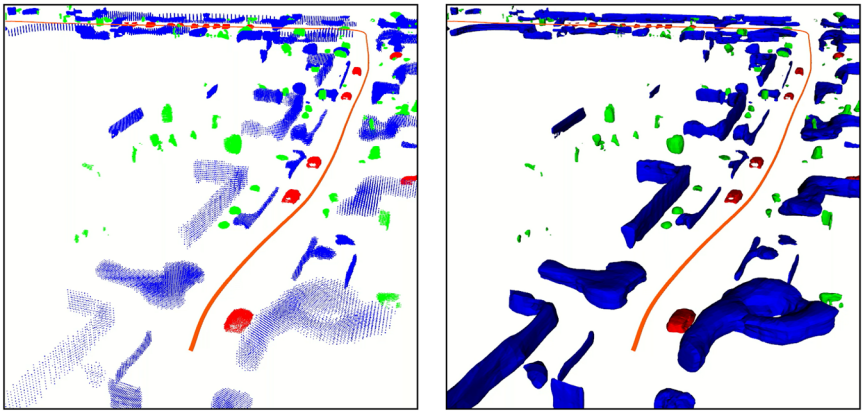

Fig. 12. Visualization of segment reconstructions, as point clouds (left) and as surface meshes (right), generated from sequence 00 of the KITTI dataset. The quantization of point cloud reconstructions is most notable in the large wall segments (blue) visible in the background. Equivalent surface mesh representations do not suffer from this issue.

mapping system, LOAM (Zhang and Singh, 2014). Our implementation is based on a publicly available version of LOAM and achieves similar odometry performance results on KITTI, as the ones reported by other works, such as Velas et al. (2018). We use a loosely coupled approach, where LOAM is used to undistort the scans and provide an odometry estimate between frames, in real-time. The scans from LOAM are used to build a local map from which segments are extracted and attached to a pose-graph, together with the odometry measurements. Loop closures can then be added in real-time as constraints in the graph, to correct the drifting odometry. This results in a real-time LiDAR-only end-to-end pipeline that produces segment-based maps of the environment, with loop-closures.

In all experiments, we use a local map with a radius of 50 m around the robot. When performing segment retrieval we consider 64 neighbors and require a minimum of 7 correspondences, which are altogether geometrically consistent, to output a localization. These parameters were chosen empirically using the information presented in Figures 7 and 8 as a reference.

Our evaluations on KITTI sequences 00 and 08 (Figure 11) demonstrate that global localization results from *SegMap* help correct for the drift of the odometry

estimates. The trajectories outputted by the system combining *SegMap* and LOAM, follow more precisely the ground-truth poses provided by the benchmark, compared with the open-loop solution. We also show how global localizations reduce both translational and rotational errors. In particular, over longer paths *SegMap* is able to reduce the drift in the trajectory estimate by up to a factor of two, considering both translation and rotation errors. For shorter paths, the drift only improves marginally or remains the same, as local errors are more dependent on the quality of the odometry estimate. We believe that our evaluation showcases not only the performance of *SegMap*, but also the general benefits stemming from global localization algorithms.

5.9. Multi-robot experiments

We evaluate the *SegMap* approach on three large-scale multi-robot experiments: one in an urban-driving environment and two in search and rescue scenarios. In both indoor and outdoor scenarios we use the same model that was trained on the KITTI sequences 05 and 06 as described in Section 5.3.

The experiments are run on a single machine, with a multi-thread approach to simulating a centralized system. One thread per robot accumulates the 3D measurements, extracting segments, and performing the descriptor extraction. The descriptors are transmitted to a separate thread that localizes the robots through descriptor retrieval and geometric verification, and runs the pose-graph optimization. In all experiments, sufficient global associations need to be made, in real-time, for co-registration of the trajectories and merging of the maps. Moreover, in a centralized setup it might be crucial to limit the transmitted data over a wireless network with potentially limited bandwidth.

5.9.1. Multi-robot SLAM in an urban scenario. In order to simulate a multi-robot setup, we split sequence 00 of the KITTI odometry dataset into five sequences, which are simultaneously played back on a single computer for a duration of 114 seconds. In this experiment, the semantic information extracted from the *SegMap* descriptors is used to reject segments classified as *vehicles* from the retrieval process.

With this setup, 113 global associations were discovered, allowing all the robot trajectories to be linked and a common representation to be created. We note that performing ICP between the associated point clouds would refine the localization transformation by, on average, only 0.13 ± 0.06 m, which is of the order of our voxelization resolution. However, this would require the original point cloud data to be kept in memory and transmitted to the central computer. Future work could consider refining the transformations by performing ICP on the reconstructions.

Localization and map reconstruction was performed at an average frequency of 10.5 Hz and segment description was responsible for 30% of the total runtime with an average duration of 28.4 ms per local cloud. A section of the target map that has been reconstructed from the descriptors is depicted in Figure 1.

Table 2. Statistics resulting from the three experiments.

Statistic	KITTI	Powerplant	Foundry
Duration (s)	114	850	1,086
Number of robots	5	3	2
Number of segmented local cloud	557	758	672
Average number of segments per cloud	42.9	37.0	45.4
Bandwidth for transmitting local clouds (kB/s)	4,814.7	1,269.2	738.1
Bandwidth for transmitting segments (kB/s)	2,626.6	219.4	172.2
Bandwidth for transmitting descriptors (kB/s)	60.4	9.5	8.1
Final map size with the *SegMap* descriptor (kB)	386.2	181.3	121.2
Number of successful localizations	113	27	85

Table 2 presents the results of this experiment. The required bandwidth is estimated by considering that each point is defined by three 32-bit floats and that 288 additional bits are required to link each descriptor to the trajectories. We only consider the *useful data* and ignore any transfer overhead. The final map of the KITTI sequence 00 contains 1,341 segments out of which 284 were classified as vehicles. A map composed of all the raw segment point clouds would be 16.8 MB whereas using our descriptor it is reduced to only 386.2 kB. This compression ratio of $43.5\times$ can be increased to $55.2\times$ if one decides to remove vehicles from the map. This shows that our approach can be used for mapping much larger environments.

5.9.2. Multi-robot SLAM in disaster environments. For the two following experiments, we use data collected by unmanned ground vehicles (UGVs) equipped with multiple motor encoders, an Xsens MTI-G inertial measurement unit (IMU), and a rotating 2D SICK LMS-151 LiDAR. First, three UGVs were deployed at the decommissioned Gustav Knepper powerplant: a large two-floor utility building measuring 100 m long by 25 m wide. The second mission took place at the Phoenix-West foundry in a semi-open building made of steel. A section measuring 100 m by 40 m was mapped using two UGVs. The buildings are shown in Figure 13.

Fig. 13. Buildings of the Gustav Knepper powerplant (left) and the Phoenix-West foundry (right).

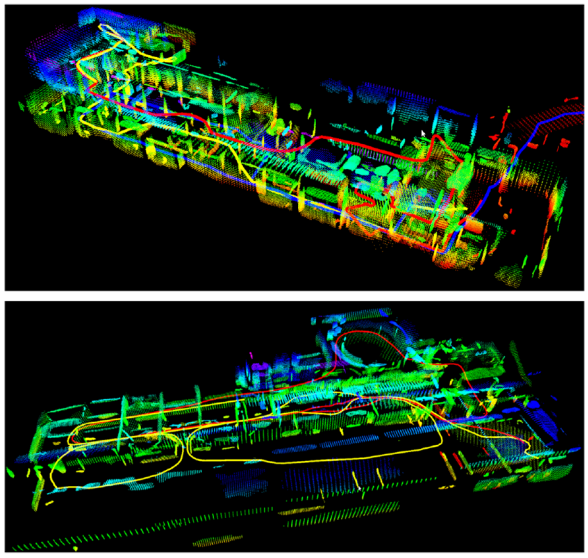

Fig. 14. Reconstruction of the buildings of the Gustav Knepper powerplant (top) and the Phoenix-West foundry (bottom). The point clouds are colored by height and the estimated robot trajectories are depicted with colored lines.

For these two experiments, we used an incremental smoothness-based region growing algorithm that extracts plane-like segments (Dubé et al., 2018b). The resulting *SegMap* reconstructions are shown in Figure 14 and detailed statistics are presented in Table 2. Although these planar segments have a very different nature than those used for training the descriptor extractor, multiple localizations have been made in real-time so that consistent maps could be reconstructed in both experiments. Note that these search and rescue experiments were performed with sensors without full 360° field of view. Nevertheless, *SegMap* allowed robots to localize in areas visited in opposite directions.

6. Discussion and future work

While our proposed method works well in the demonstrated experiments it is limited by the ability to only observe the geometry of the surrounding structure. This can be problematic in some man-made environments, which are repetitive and can lead to perceptual aliasing, influencing both the descriptor and the geometric consistency verification. This could be addressed by detecting such aliasing instances and dealing with them explicitly, through, for example, an increase in the constraints of the geometric verification. On the other hand, featureless environments, such as flat fields or straight corridors, are equally challenging. As even LiDAR-based odometry methods struggle to maintain an accurate estimate of pose, these environments do not allow for reliable segment extraction. In these cases the map will drift until a more distinct section is reached that can be loop-closed, thus allowing for partial correction of the previously built pose-graph. In different environments the two segmentation algorithms will have varying performances, with the Euclidean distance based one working better in outdoor scenarios, while the curvature-based one is more suited to indoor scenarios. A future approach would be to run the two segmentation strategies in parallel, thus allowing them to compensate for each other's shortcomings and enabling robots to navigate in multiple types of environments during the same mission.

In order to address some of the aforementioned drawbacks, in the future work we would like to extend the *SegMap* approach to different sensor modalities and different point cloud segmentation algorithms. For example, integrating information from camera images, such as color, into the descriptor learning could mitigate the lack of descriptiveness of features extracted from segments with little distinct geometric structure. In addition, color and semantic information from camera images could not only be used to improve the descriptor, but also to enhance the robustness of the underlying segmentation process. Considering the real-time constraints of the system, to note with respect to future work are the additional computational expenses introduced by processing and combining more data modalities.

Furthermore, whereas the present work performs segment description in a discrete manner, it would be interesting to investigate incremental updates of learning-based descriptors that could make the description process more

efficient, such as the voting scheme proposed by Engelcke et al. (2017). Instead of using a feed-forward network, one could also consider a structure that leverages temporal information in the form of recurrence in order to better describe segments based on their evolution in time. Moreover, it could be of interest to learn the usefulness of segments as a precursory step to localization, based on their distinctiveness and semantic attributes.

7. Conclusion

In this article, we have presented *SegMap*: a segment-based approach for *map representation* in localization and mapping with 3D sensors. In essence, the robots' surroundings are decomposed into a set of segments, and each segment is represented by a distinctive, low-dimensional learning-based descriptor. Data associations are identified by segment descriptor retrieval and matching, made possible by the repeatable and descriptive nature of segment-based features.

We have shown that the descriptive power of *SegMap* outperforms hand-crafted features as well as the evaluated data-driven baseline solutions. Our experiments have indicated that *SegMap* offers competitive localization performance, in comparison to the state-of-the-art LocNet method. In addition, we have combined our localization approach with LOAM, a LiDAR-based local motion estimator, and have demonstrated that the output of *SegMap* helps correct the drift of the open-loop odometry estimate. Finally, we have introduced *SegMini*: a lightweight version of our *SegMap* descriptor which can more easily be deployed on platforms with limited computational power.

In addition to enabling global localization, the *SegMap* descriptor allows us to reconstruct a map of the environment and to extract semantic information. The ability to reconstruct the environment while achieving a high compression rate is one of the main features of *SegMap*. This allows us to perform both SLAM and 3D reconstruction with LiDARs at large scale and with low communication bandwidth between the robots and a central computer. These capabilities have been demonstrated through multiple experiments with real-world data in urban driving and search and rescue scenarios. The reconstructed maps could allow performing navigation tasks such as, for instance, multi-robot global path planning or increasing situational awareness.

Acknowledgements

The authors would like to thank Abel Gawel, Mark Pfeiffer, Mattia Gollub, Helen Oleynikova, Philipp Krüsi, Igor Gilitschenski, and Elena Stumm for their valuable collaboration and support.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the European Union's Seventh Framework Program for research, technological development and demonstration under the "Long-Term Human–Robot Teaming for Robots Assisted Disaster Response" (TRADR) project (project number FP7-ICT-609763), from the EU H2020 research project (grant agreement number 688652), the Swiss State Secretariat for Education, Research and Innovation (SERI; grant number 15.0284), and by the Swiss National Science Foundation through the National Center of Competence in Research Robotics (NCCR).

Notes

1. *SegMap* is open-source available along with easy to run demonstrations at www.github.com/ethz-asl/segmap. A video demonstration is available at https://youtu.be/CMk4w4eRobg (Extension 1).
2. In our experiments, we consider two techniques for estimating the local motion by registering successive LiDAR scans: one which uses ICP and one based on LOAM (LiDAR Odometry and Mapping in Real-time) (Zhang and Singh, 2014).
3. For more information on these segmentation algorithms, the reader is encouraged to consult our prior work (Dubé et al., 2018b).

References

Bosse M and Zlot R (2013) Place recognition using keypoint voting in large 3D LiDAR datasets. In: *IEEE International Conference on Robotics and Automation (ICRA)*.

Bowman SL, Atanasov N, Daniilidis K and Pappas GJ (2017) Probabilistic data association for semantic SLAM. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1722–1729.

Brock A, Lim T, Ritchie J and Weston N (2016) Generative and discriminative voxel modeling with convolutional neural networks. In: *Workshop on 3D Deep Learning, NIPS*.

Bromley J, Guyon I, LeCun Y, Säckinger E and Shah R (1994) Signature verification using a "Siamese" time delay neural network. In: *Advances in Neural Information Processing Systems*, pp. 737–744.

Cadena C, Carlone L, Carrillo H, et al. (2016) Past, present, and future of simultaneous localization and mapping: Towards the robust-perception age. *IEEE Transactions on Robotics* 32(6): 1309–1332.

Cop K, Borges P and Dubé R (2018) DELIGHT: An efficient descriptor for global localisation using LiDAR intensities. In: *IEEE International Conference on Robotics and Automation (ICRA)*.

Dai A, Ruizhongtai Qi C and Niessner M (2017) Shape completion using 3D-encoder–predictor CNNs and shape synthesis. In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Dewan A, Caselitz T and Burgard W (2018) Learning a local feature descriptor for 3D LiDAR scans. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain.

Dubé R, Cramariuc A, Dugas D, Nieto J, Siegwart R and Cadena C (2018a) SegMap: 3D segment mapping using data-driven descriptors. In: *Robotics: Science and Systems (RSS)*.

Dubé R, Dugas D, Stumm E, Nieto J, Siegwart R and Cadena C (2017a) SegMatch: Segment-based place recognition in 3D point clouds. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5266–5272.

Dubé R, Gawel A, Sommer H, Nieto J, Siegwart R and Cadena C (2017b) An online multi-robot SLAM system for 3D LiDARs. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 1004–1011.

Dubé R, Gollub MG, Sommer H, et al. (2018b) Incremental segment-based localization in 3D point clouds. *IEEE Robotics and Automation Letters* 3(3): 1832–1839.

Elbaz G, Avraham T and Fischer A (2017) 3D point cloud registration for localization using a deep neural network auto-encoder. In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 2472–2481.

Elseberg J, Magnenat S, Siegwart R and Nüchter A (2012) Comparison of nearest-neighbor-search strategies and implementations for efficient shape registration. *Journal of Software Engineering for Robotics* 3(1): 2–12.

Engelcke M, Rao D, Wang DZ, Tong CH and Posner I (2017) Vote3deep: Fast object detection in 3D point clouds using efficient convolutional neural networks. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1355–1361.

Fang Y, Xie J, Dai G, et al. (2015) 3D deep shape descriptor. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2319–2328.

Fernández-Moral E, Mayol-Cuevas W, Arevalo V and Gonzalez-Jimenez J (2013) Fast place recognition with plane-based maps. In: *IEEE International Conference on Robotics and Automation (ICRA)*.

Fernández-Moral E, Rives P, Arévalo V and González-Jiménez J (2016) Scene structure registration for localization and mapping. *Robotics and Autonomous Systems* 75: 649–660.

Finman R, Paull L and Leonard JJ (2015) Toward object-based place recognition in dense RGB-d maps. In: *ICRA Workshop on Visual Place Recognition in Changing Environments*.

Gawel A, Cieslewski T, Dubé R, Bosse M, Siegwart R and Nieto J (2016) Structure-based vision-laser matching. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 182–188.

Geiger A, Lenz P and Urtasun R (2012) Are we ready for autonomous driving? The KITTI vision benchmark suite. In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Glorot X and Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. *Aistats* 9: 249–256.

Graham B, Engelcke M and van der Maaten L (2018) 3d semantic segmentation with submanifold sparse convolutional networks. In: *IEEE Conference on Computer Vision and Pattern Recognition*. pp. 18–22.

Granström K, Schön TB, Nieto JI and Ramos FT (2011) Learning to close loops from range data. *The International Journal of Robotics Research* 30(14): 1728–1754.

Guizilini V and Ramos F (2017) Learning to reconstruct 3D structures for occupancy mapping. In: *Robotics: Science and Systems (RSS)*.

Hermans A, Beyer L and Leibe B (2017) In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*.

Hess W, Kohler D, Rapp H and Andor D (2016) Real-time loop closure in 2D LiDAR SLAM. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1271–1278.

Ioffe S and Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: *International Conference on Machine Learning*, pp. 448–456.

Kaess M, Johannsson H, Roberts R, Ila V, Leonard JJ and Dellaert F (2012) iSAM2: Incremental smoothing and mapping using the Bayes tree. *The International Journal of Robotics Research* 31(2): 216–235.

Kingma DP and Ba JL (2015) ADAM: A method for stochastic optimization. In: *International Conference on Learning Representations*, pp. 1–13.

Krizhevsky A, Sutskever I and Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*, pp. 1097–1105.

Li B, Zhang T and Xia T (2016) Vehicle detection from 3D lidar using fully convolutional network. In: *Robotics: Science and Systems (RSS)*.

Lorensen WE and Cline HE (1987) Marching cubes: A high resolution 3D surface construction algorithm. In: *ACM Siggraph Computer Graphics*, Vol. 21. New York: ACM Press, pp. 163–169.

Lowry S, Sunderhauf N, Newman P, et al. (2016) Visual place recognition: A survey. *IEEE Transactions on Robotics* 32(1): 1–19.

Magnusson M, Andreasson H, Nüchter A and Lilienthal AJ (2009) Automatic appearance-based loop detection from three-dimensional laser data using the normal distributions transform. *Journal of Field Robotics* 26(11–12): 892–914.

Maturana D and Scherer S (2015) VoxNet: A 3D convolutional neural network for real-time object recognition. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Parkhi OM, Vedaldi A, Zisserman A, et al. (2015) Deep face recognition. In: *Proceedings of the British Machine Vision Conference (BMVC)*, Vol. 1, p. 6.

Qi CR, Su H, Mo K and Guibas LJ (2017) Pointnet: Deep learning on point sets for 3D classification and segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Ricao Canelhas D, Schaffernicht E, Stoyanov T, Lilienthal AJ and Davison AJ (2017) Compressed voxel-based mapping using unsupervised learning. *Robotics* 6(3): 15.

Riegler G, Osman Ulusoy A and Geiger A (2017) OctNet: Learning deep 3D representations at high resolutions. In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Röhling T, Mack J and Schulz D (2015) A fast histogram-based similarity measure for detecting loop closures in 3-D LiDAR data. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Rusu RB, Blodow N and Beetz M (2009) Fast point feature histograms (FPFH) for 3D registration. In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3212–3217.

Salti S, Tombari F and Di Stefano L (2014) SHOT: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding* 125: 251–264.

Schönberger JL, Pollefeys M, Geiger A and Sattler T (2018) Semantic visual localization. In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Schroff F, Kalenichenko D and Philbin J (2015) Facenet: A unified embedding for face recognition and clustering. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 815–823.

Srivastava N, Hinton GE, Krizhevsky A, Sutskever I and Salakhutdinov R (2014) Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1): 1929–1958.

Steder B, Grisetti G and Burgard W (2010) Robust place recognition for 3D range data based on point features. In: *IEEE International Conference on Robotics and Automation (ICRA)*.

Steder B, Ruhnke M, Grzonka S and Burgard W (2011) Place recognition in 3D scans using a combination of bag of words and point feature based relative pose estimation. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Tchapmi LP, Choy CB, Armeni I, Gwak J and Savarese S (2017) SEGCloud: Semantic segmentation of 3D point clouds. In: *International Conference on 3D Vision (3DV)*.

Tinchev G, Nobili S and Fallon M (2018) Seeing the wood for the trees: Reliable localization in urban and natural environments. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

Varley J, DeChant C, Richardson A, Ruales J and Allen P (2017) Shape completion enabled robotic grasping. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2442–2447.

Velas M, Spanel M, Hradis M and Herout A (2018) CNN for IMU assisted odometry estimation using Velodyne LiDAR. In: *2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE, pp. 71–77.

Weinberger KQ, Blitzer J and Saul LK (2006) Distance metric learning for large margin nearest neighbor classification. In: *Advances in Neural Information Processing Systems*, pp. 1473–1480.

Weinmann M, Jutzi B and Mallet C (2014) Semantic 3D scene interpretation: A framework combining optimal neighborhood size selection with relevant features. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2(3): 181.

Wohlhart P and Lepetit V (2015) Learning descriptors for object recognition and 3D pose estimation. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3109–3118.

Wu B, Wan A, Yue X and Keutzer K (2018) Squeezeseg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1887–1893.

Wu J, Zhang C, Xue T, Freeman B and Tenenbaum J (2016) Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In: *Advances in Neural Information Processing Systems*, pp. 82–90.

Wu Z, Song S, Khosla A, et al. (2015) 3D shapenets: A deep representation for volumetric shapes. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1912–1920.

Ye Y, Cieslewski T, Loquercio A and Scaramuzza D (2017) Place recognition in semi-dense maps: Geometric and learning-based approaches. In: *Proceedings of the British Machine Vision Conference (BMVC)*.

Yin H, Tang L, Ding X, Wang Y and Xiong R (2018) Locnet: Global localization in 3d point clouds for mobile vehicles. In: *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 728–733.

Yin H, Wang Y, Tang L, Ding X and Xiong R (2017) Locnet: Global localization in 3D point clouds for mobile robots. *arXiv preprint arXiv:1712.02165*.

Zeng A, Song S, Nießner M, Fisher M, Xiao J and Funkhouser T (2017) 3DMatch: Learning local geometric descriptors from rgb-d reconstructions. In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Zhang J and Singh S (2014) LOAM: LiDAR odometry and mapping in real-time. In: *Robotics: Science and Systems (RSS)*.

Zhuang Y, Jiang N, Hu H and Yan F (2013) 3-D-laser-based scene measurement and place recognition for mobile robots in dynamic indoor environments. *IEEE Transactions on Instrumentation and Measurement* 62(2): 438–450.

Appendix. Index to multimedia extensions

Archives of IJRR multimedia extensions published prior to 2014 can be found at http://www.ijrr.org, after 2014 all videos are available on the IJRR YouTube channel at http://www.youtube.com/user/ijrrmultimedia

Table of Multimedia Extensions

Extension	Media type	Description
1	Video	https://www.youtube.com/watch?v=CMk4w4eRobg&t=2s