Building Volumetric Beliefs for Dynamic Environments Exploiting Map-Based Moving Object Segmentation

Benedikt Mersch, Tiziano Guadagnino, Xieyuanli Chen, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss

***Abstract*—Mobile robots that navigate in unknown environments need to be constantly aware of the dynamic objects in their surroundings for mapping, localization, and planning. It is key to reason about moving objects in the current observation and at the same time to also update the internal model of the static world to ensure safety. In this paper, we address the problem of jointly estimating moving objects in the current 3D LiDAR scan and a local map of the environment. We use sparse 4D convolutions to extract spatio-temporal features from scan and local map and segment all 3D points into moving and non-moving ones. Additionally, we propose to fuse these predictions in a probabilistic representation of the dynamic environment using a Bayes filter. This volumetric belief models, which parts of the environment can be occupied by moving objects. Our experiments show that our approach outperforms existing moving object segmentation baselines and even generalizes to different types of LiDAR sensors. We demonstrate that our volumetric belief fusion can increase the precision and recall of moving object segmentation and even retrieve previously missed moving objects in an online mapping scenario.**

***Index Terms*—Mapping; Computer Vision for Transportation; Intelligent Transportation Systems**

I. Introduction

SEGMENTING moving and non-moving objects is key for mobile robots operating in dynamic environments. It is an important step for online applications like mapping [33], [38], localization [14], [31], planning [21], or occupancy prediction [15]. To solve such tasks, a robot needs to reason about which parts of the environment are moving and which are not in an online fashion. For successful navigation and planning, this knowledge should not only be limited to what the robot currently perceives but rather be integrated into a representation of the environment.

In this paper, we investigate the problem of segmenting moving objects in both current and past 3D LiDAR scans. Additionally, we maintain a 3D model of the environment representing our belief about which part of the space can contain moving objects as depicted in Fig. 1. We update the belief online by fusing our predictions in a probabilistic manner to increase precision and recall of moving object segmentation (MOS). Also, for simultaneous localization and mapping (SLAM), it is of central interest to estimate, which parts of the environment are dynamic. The knowledge of moving objects can be directly integrated into the optimization as an object motion estimation as shown by Henein et al. [14], in this example done for rigid body motion. Pfreundschuh et al. [31] and Chen et al. [7], [9] demonstrate the effectiveness of moving object segmentation for data associations. An alternative strategy for localization and long-term planning is to build a map and clean it from traces of dynamic objects in a post-processing step, [1], [2], [18], [22]. Thus, the addressed estimation problem has multiple relevant applications in robotics.

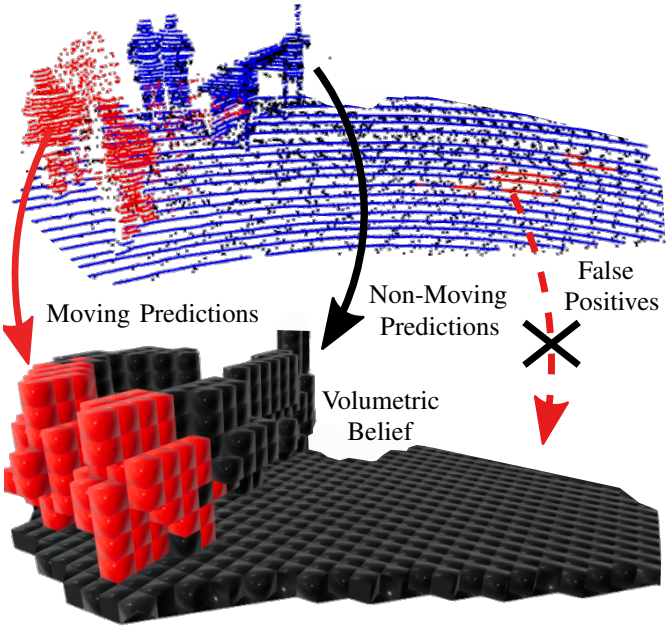

Fig. 1: Our approach identifies moving objects *(red)* in the current scan *(blue)* and the local map *(black)* of the environment. We maintain a volumetric belief map representing the dynamic environment and fuse new predictions in a probabilistic fashion. This allows us to reject false positive predictions that contradict our volumetric belief.

If building a static map is required online, one way is to segment each incoming scan into moving and non-moving and then integrate only the static points into the map [7]. In this setup, each segmentation is done independently of

Manuscript received: March 10, 2023; Revised: June 06, 2023; Accepted: June 28, 2023. This paper was recommended for publication by Editor Javier Civera upon evaluation of the Associate Editor and Reviewers' comments.

This work has partially been funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017008 (Harmony) and by the European Union's Horizon Europe research and innovation programme under grant agreement No 101070405 (DigiForest).

All authors are with the University of Bonn, Germany. Cyrill Stachniss is additionally with the Department of Engineering Science at the University of Oxford, UK, and with the Lamarr Institute for Machine Learning and Artificial Intelligence, Germany.

Digital Object Identifier (DOI): see top of this page.

previous predictions. The downside of this approach is that it is not straightforward to recover a missed moving object that was added to the map. Recently, 4DMOS [25] improved the segmentation robustness by re-estimating moving objects in a scan after receiving more observations and fusing them in a binary Bayes filter. However, the MOS robustness can only be improved as long as the corresponding scan is within the limited buffer of past scans that 4DMOS and related approaches consider for prediction. Also, the idea of using a local buffer assumes that the movement of an object can be identified from consecutive measurements. This usually holds for most rotating LiDAR scanners that scan the surroundings with a regular scanning pattern at high frequency, but not for scanners with a limited field of view or irregular sampling patterns [23].

The main contributions of this paper are two-fold. First, we propose an approach to predict moving objects in a local map constructed using all past LiDARs measurements recorded in this area without limiting the time horizon. Second, we build and maintain a volumetric belief map and fuse new predictions in a voxel-wise binary Bayes filter to previous estimates online, which increases robustness and corrects previously wrong predictions. In sum, we make four key claims: Our approach is able to (i) accurately segment an incoming LiDAR scan into moving and non-moving objects based on a local map of past observations, (ii) generalize well to new environments and sensor setups while achieving state-of-the-art performance, (iii) increase the precision and recall of moving object segmentation by fusing multiple predictions into a volumetric belief, (iv) recover from wrong predictions for online mapping through a volumetric belief. These claims are backed up by the paper and our experimental evaluation. Our code, pre-trained models, and labels for evaluation are available at https://github.com/PRBonn/MapMOS.

II. Related Work

Online LiDAR MOS is usually achieved by comparing the current scan against the past, with the goal of segmenting the corresponding point cloud into moving and non-moving parts [8], [25]. Yoon et al. [42] identify moving objects based on the residual between two scans, free space filtering, and region growing post-processing. One drawback of such an approach is that objects can be temporarily occluded, which makes it hard to identify motion only considering two scans.

Subsequent works extend the temporal horizon of past information that is used for prediction. Processing more points at full resolution is often computationally demanding. Therefore, most methods project the data into a lower-dimensional representation [7], [19], [36]. Chen et al. [7] use past residual images together with a semantic segmentation network to segment a range-image representation of the scan into moving and non-moving. Work by Kim et al. [19] extends this idea by additionally predicting movable and non-movable objects from semantics. Sun et al. [36] propose a point refinement module to reduce the effect of imprecise boundaries, sometimes referred to as "label bleeding" for range image-based segmentation [26].

The problem of label bleeding is also addressed in 4DMOS [25] by predicting moving objects in the voxelized 4D space without prior projection. It assumes that the motion of an object is visible within a limited time horizon of consecutive past scans, which are aggregated to a sparse 4D point cloud. By shifting this temporal window, the prediction of previous scans can be refined by fusing them in a point-wise binary Bayes filter. To reduce the effort of labeling, Kreutz et al. [20] proposed a feature encoding and clustering approach based on a 4D occupancy time series.

Similar to 4DMOS, we extract spatio-temporal features using 4D convolution instead of projecting the data. In contrast to the aforementioned methods [7], [19], [25], [36], our proposed approach predicts moving objects using the current scan and a voxelized local point cloud of all past scans without limiting the time horizon.

Static Map Building – A standard approach to obtain a static model of the environment is to only integrate static points based on a scan-wise MOS [7]. Other researchers focused on geometric approaches to obtain a static representation of the environment. For example, occupancy maps divide the space into occupied, free, and unobserved areas [37]. The static belief of voxels is updated by ray-tracing and recursive Bayesian estimation using an inverse sensor model [37]. The final map can be used to decide if a new measurement belongs to a dynamic object or not [40]. Stachniss and Burgard [35] propose an approach for 2D grid-based localization in non-static environments by clustering possible configurations of the changing environment which improves localization. To cover the full spectrum of temporal changes in the environment, Biber and Duckett [5] update a map based on different time scales.

In contrast, Nuss et al. [29] propose to use random finite sets to explicitly model the dynamic state of each grid cell, which has been further used for tasks like occupancy prediction [15]. To deal with 3D LiDAR data, Wurm et al. [41] and Hornung et al. [16] introduced OctoMap which extends occupancy grid mapping to the 3D space by using an octree data structure. Ray-tracing on volumetric occupancy grids has also been researched to remove dynamic objects from a set of LiDAR scans [12], [34]. Similarly, Pagad et al. [30] use an octree to build an occupancy grid map by first detecting ground and object points and using ray-tracing to update voxel occupancy. Arora et al. [1], [2] exploit OctoMap for static map cleaning and leverage ground-segmentation and a voting scheme to deal with unknown points.

Visibility-based methods [18], [22], [32] alleviate the computational cost of ray-tracing by checking the consistency of a query point with respect to a pre-built map. For example, Lim et al. [22] identify temporarily occluded regions in an accumulated point cloud map based on height discrepancy between query and map. Instead of removing dynamic points, Huang et al. [17] explicitly target the reconstruction of moving objects for 3D scene analysis. To deal with the sparse measurements from moving objects, the authors register multiple point clouds and estimate offset vectors of previously classified moving points.

In our work, we aim at closing the gap between scan-wise

online MOS and an offline volumetric representation of the dynamic environment. We propose an approach that segments the current scan as well as previously received measurements into moving and non-moving points and fuses these predictions in a 3D volumetric representation. In contrast to most of the aforementioned approaches, we maintain this belief online and use it to robustify the current prediction and to retrieve previously missed moving objects for online mapping.

III. Our Approach

We propose to segment moving objects based on the discrepancy between the current LiDAR frame and a local map consisting of the previously measured scans in that area. Given the current LiDAR frame at time t, we first register it to our current local map as explained in Sec. III-A. Next, we jointly predict moving objects in the aligned scan and the local map, see Sec. III-B. After that, we fuse these predictions into a probabilistic volumetric belief to maintain a representation of the dynamic environment, see Sec. III-C. We can query the volumetric belief for a set of points as explained in Sec. III-D to obtain the current belief if these points belong to moving objects or not.

A. Scan Registration with KISS-ICP

Our approach does not require ground truth poses, it only relies on sequential 3D LiDAR data. When a new measurement is available, we register the scan using KISS-ICP [39], which is a robust odometry pipeline that generalizes well to varying motion profiles and sensor platforms without the need for changing parameters.

Our local map used within this paper is a sparse voxel grid as the one of KISS-ICP. We maintain the original coordinates of the points in the voxels to avoid discretization errors. In contrast to the original KISS-ICP implementation, we additionally store for each point the timestamp of the scan it stems from to maintain temporal information in the local map. Our method directly uses this temporal information to predict moving objects for both the registered scan and the local map.

B. Map-based Moving Object Segmentation

We start by explaining how to jointly predict moving objects after registering a new LiDAR frame. We exploit two different mechanisms to segment moving objects. First, we consider the spatial discrepancy between the current scan and the local map. This information indicates if an object may have moved with respect to all previous measurements in that area. Second, we identify the motion of objects based on the evolution of the timestamps given by the feature attached to each point. This allows us to segment both the current scan and the local map into moving and non-moving parts.

In contrast to previous works [7], [19], [25], [36], our method is not restricted to a fixed set of past scans. This is advantageous in cases a moving object is not fully visible within a short time horizon due to occlusion, limited field of view, or an irregular shooting pattern of the LiDAR. In practice, this makes a substantial difference.

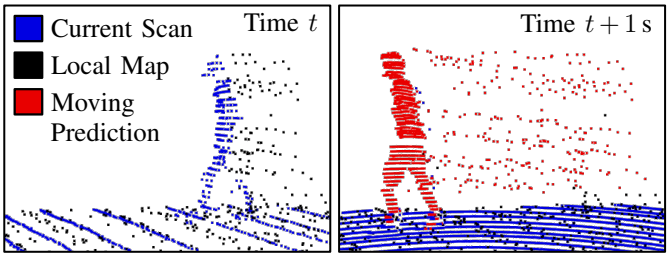

Fig. 2: Current Scan and local map for two different times with our moving predictions in red. Although our method initially failed to fully identify the moving pedestrian in the beginning *(left)*, we successfully predict it at a later point in time and our method backtraces the corresponding points in the local map *(right)*.

Additionally, instead of predicting moving objects in the current scan or a limited buffer of scans, we segment both the current scan and the local map. Segmenting the local map enables us to identify traces of moving objects that were not segmented in previous scan predictions. This backtracing of dynamic objects allows us to correct initial false negative predictions as shown in Fig. 2.

Our local map is the voxel grid structure of KISS-ICP, but we store for every point its 4D coordinate (position plus time). To maintain the ordering of scan and local map during the convolutions, we organize them in a 4D tensor. We use the timestamps as features for the points and normalize them based on the minimum and maximum values since we are only interested in their relative difference. This avoids the model overfitting to the sequence lengths and, therefore, the maximum timestamps it has seen during training.

At time t, we voxelize the 4D point cloud $\mathcal{P}_t$ of scan and local map and represent it as a sparse 4D tensor using the MinkowskiEngine [10]. Sparse tensors are a more memory-efficient representation for 4D tensor data and allow to directly apply sparse convolutions. We jointly extract spatial and temporal features with sparse 4D convolutions. Our network architecture is a 4D MinkUNet [10] with $1.8\,\mathrm{Mio}$ parameters. This network first downsamples the points and features in an encoder to extract high-level information and then upsamples both to the original resolution in a decoder. Residual blocks and skip connections help to maintain detailed information about the points and their corresponding features. The last layer predicts the logits $\mathcal{S}_t$ of both current scan and local map points being moving. Fig. 3 depicts an overview of our approach.

C. Volumetric Belief Update

In this section, we present our approach to fuse per-point MOS predictions into a probabilistic volumetric belief. Fusing multiple independent predictions over time can filter out prediction errors from the neural network and has been previously explored on the point-level [25]. Instead of fusing per-points predictions, our goal is to model, which parts of the environment have a higher probability of containing a dynamic object. Notice that in this case, we do not want to just identify current dynamics, but rather determine which portion of the map is traversed by moving objects. We define this property as *dynamic occupancy*.

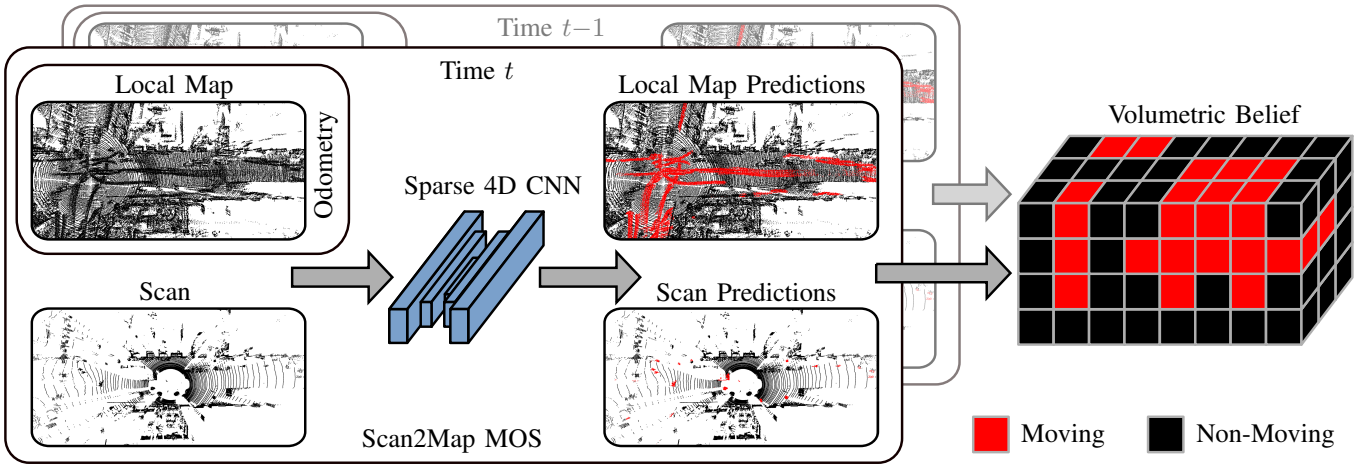

Fig. 3: Overview our proposed Scan2Map moving object segmentation approach and volumetric belief fusion. At time t, we predict moving objects in the current scan and local map using sparse 4D convolutions. Next, we update our volumetric belief about which parts of the environment can contain moving objects based on the previous volumetric belief at time $t-1$ and our new predictions.

We assume that the binary state $m_i \in \{0, 1\}$ of dynamic occupancy for a voxel v_i does not change over time. Intuitively, this means that if a point falls into a voxel previously occupied by dynamics, we assume that this point also belongs to a moving object. On the other hand, if a voxel was occupied by static points, we do not expect to observe a moving object in this volume. Note that this state definition is different from occupancy grid mapping, where the world is assumed to be static and a fixed occupancy probability of a cell is estimated.

At time t, we predict N logits $\mathcal{S}_t = \{s_{t,1}, s_{t,2}, \dots s_{t,N}\}$ with $s_{t,j} \in \mathbb{R}$ for N points $\mathcal{P}_t = \{\boldsymbol{p}_{t,1}, \boldsymbol{p}_{t,2}, \dots \boldsymbol{p}_{t,N}\}$ with $\boldsymbol{p}_{t,j} \in \mathbb{R}^4$ as described in Sec. III-B. It is possible to fuse the logits for the current scan but also for the local map points. We provide an experiment in Sec. IV-D to showcase the results for different fusion strategies. Note that our volumetric belief is not restricted to our logits, but that predictions from different sources could be integrated. Our goal is to estimate the joint probability distribution of the volumetric belief map state for all voxels $\mathcal{M} = \{m_i\}$ reading

$$p\left(\mathcal{M} \mid \mathcal{P}_{1:t}, \mathcal{S}_{1:t}\right) = \prod_i p\left(m_i \mid \mathcal{P}_{1:t}, \mathcal{S}_{1:t}\right), \tag{1}$$

with $\mathcal{P}_{1:t}$ and $\mathcal{S}_{1:t}$ being the sets of previously measured points and predicted logits up to time t, respectively.

After applying Bayes' rule to the right-hand side per-voxel probability distribution, we can derive the recursive binary Bayes filter equations according to Thrun et al. [37]. We use the log-odds notation $l(x) = \log \frac{p(x)}{1-p(x)}$ resulting in

$$\begin{aligned} l\left(m_i \mid \mathcal{P}_{1:t}, \mathcal{S}_{1:t}\right) = l\left(m_i \mid \mathcal{P}_{1:t-1}, \mathcal{S}_{1:t-1}\right) \\ + l\left(m_i \mid \mathcal{P}_t, \mathcal{S}_t\right) - l(m_i), \end{aligned} \tag{2}$$

for updating a single voxel cell belief $l\left(m_i \mid \mathcal{P}_{1:t}, \mathcal{S}_{1:t}\right)$. Here, $l\left(m_i \mid \mathcal{P}_{1:t-1}, \mathcal{S}_{1:t-1}\right)$ is the recursive term currently stored in the voxel, which aggregates the previous predictions, $l\left(m_i \mid \mathcal{P}_t, \mathcal{S}_t\right)$ is the update term for the voxel which integrates the predictions at the current time t, and $l(m_i)$ are the log-odds of the prior probability p_0. We do not assume to have prior knowledge about the dynamic occupancy of a voxel v_i and therefore set it to $p_0 = 0.5$.

The remaining step is to get a per-voxel update $l\left(m_i \mid \mathcal{P}_t, \mathcal{S}_t\right)$ from the points $\mathcal{P}_t$ and logits $\mathcal{S}_t$. The prediction $s_{t,j} \in \mathcal{S}_t$ at time t for a single point $\boldsymbol{p}_{t,j}$ with index j indicates if it belongs to a moving object or not. Since multiple points with different logits can end up in the same voxel, we need to aggregate their information and take the arithmetic mean of logits inside a voxel i resulting in

$$l\left(m_i \mid \mathcal{P}_t, \mathcal{S}_t\right) = \frac{\sum_{j \in \mathcal{V}_{t,i}} s_{t,j}}{|\mathcal{V}_{t,i}|}, \tag{3}$$

where $\mathcal{V}_{t,i} = \{j \mid \boldsymbol{p}_{t,j} \in v_i\}$ is the set of points falling into the voxel v_i at time t and $|\mathcal{V}_{t,i}|$ is the cardinality of the set. Taking the arithmetic mean of per-point log-odds corresponds to the geometric mean of the individual likelihoods of a point being moving. Likelihood aggregation using the geometric mean has been previously used in Monte-Carlo localization sensor model designs [43].

We implement our volumetric belief as a hash table, which is a more memory-efficient representation compared to dense 3D arrays, [28], [39]. Each 3D voxel v_i stores the log-odds belief $l\left(m_i \mid \mathcal{P}_{1:t}, \mathcal{S}_{1:t}\right)$ about its dynamic occupancy state $m_i \in \{0, 1\}$ after integrating predictions up to time t.

D. Volumetric Belief Query

For a given set of points, we can query our volumetric belief by indexing the corresponding voxels v_i and converting the log-odds beliefs $l\left(m_i \mid \mathcal{P}_{1:t}, \mathcal{S}_{1:t}\right)$ to a posterior probability p using $p(x) = \frac{e^{l(x)}}{1+e^{l(x)}}$. We assume that a point is moving if the probability is larger than 0.5. Note that the voxel size of our volumetric belief needs to be appropriate since the underlying assumption is that all points inside a voxel share the same dynamic occupancy state. This assumption is violated if the voxel size is too large.

E. Online Mapping

For online mapping, we are interested in accurately removing moving points. We experienced discretization effects at the boundaries of moving objects, for example, false negative predictions on the wheels of vehicles close to the ground. To achieve sub-voxel accuracy and a high recall for identifying moving objects, we combine the filtered voxel-wise volumetric belief with the point-wise scan prediction for online mapping. We demonstrate this in an experiment in Sec. IV-E.

F. Implementation Details

We set the voxel size used for downsampling the scans in our odometry system to $0.5\,\mathrm{m}$. We train our 4D CNN by supervising the prediction for scan and local map points using the cross-entropy loss for 100 epochs and save the model performing best on the validation set. Since some sequences of the training set do not contain a lot of moving objects, we skip a batch if the ratio between moving and static points is less than $0.1\,\%$. Next, we crop a rectangular patch of the scenes and augment the batch by rotating, flipping, and scaling. Lastly, we randomly drop points with a dropout rate sampled from the interval $[0, 0.5]$ to vary the density of the point clouds. One epoch takes less than 25 min on an NVIDIA RTX A5000.

For choosing a voxel size of our volumetric belief map, one needs to trade off between computational efficiency and accuracy due to the discretization. For our experiments, we set the fixed voxel size to $0.25\,\mathrm{m}$. Additionally, we clip the volumetric belief map at $150\,\mathrm{m}$.

IV. Experimental Evaluation

The main focus of this work is an approach to identify moving objects in the current LiDAR frame and a local map of aggregated past scans and to fuse these predictions into a probabilistic volumetric belief map. We present our experiments to show the capabilities of our method. The results of our experiments also support our key claims, which are: Our approach (i) accurately segments an incoming LiDAR scan into moving and non-moving objects based on a local map of past observations, (ii) generalizes well to new environments and sensor setups while achieving state-of-the-art performance, (iii) increases the precision and recall of MOS by fusing multiple predictions into a volumetric belief, (iv) recovers from wrong predictions for online mapping through a volumetric belief.

A. Datasets, Metrics, and Baselines

For the following experiments, we train all models on the moving labels of the SemanticKITTI [3], [4] training sequences 00-07 and 09-10 and use sequence 08 for validation. We do not use the KITTI pose information since our approach registers the scans using KISS-ICP [39] as described in Sec. III-A.

Besides the commonly used SemanticKITTI MOS benchmark [7] based on the SemanticKITTI labels, we also evaluate and compare our approach on a labeled sequence from the KITTI Tracking [13] dataset recorded with the same sensor setup in a street with a lot of moving pedestrians. We additionally report results on a subset of the Apollo Columbia Park MapData [24] with labels provided by Chen et al. [8]. This data is recorded with the same sensor, but in a different city environment.

Method	Test 11-21	Validation 08
LMNet [7]	58.3	66.4
MotionSeg3D, v1 [36]	62.5	68.1
MotionSeg3D, v2 [36]	64.9	71.4
4DMOS, delayed [25]	65.2	77.2
Ours, Scan	65.9	83.8
Ours, Volumetric Belief	**66.0**	**86.1**
RVMOS [19]*	**73.3***	71.2*

TABLE I: Comparison of average moving IoU on the SemanticKITTI validation sequence 08 and the SemanticKITTI MOS benchmark [7]. Best results in bold. The * indicates that the approach additionally *exploits semantic labels*.

To push the generalization capabilities of MOS approaches, we test the models trained on SemanticKITTI with 64 vertical beams at $10\,\mathrm{Hz}$ frequency on the nuScenes [6] dataset, which has 32 vertical beams at $20\,\mathrm{Hz}$. We evaluate the MOS for nuScenes based on the moving labels from the annotated keyframes of the 150 validation sequences.

We assess the performance using the commonly known intersection-over-union (IoU) [11] of the moving points and additionally precision and recall in Sec. IV-D.

We compare our method to the projection-based baselines LMNet [7], MotionSeg3D [36], and RVMOS [19]. For MotionSeg3D, we show the results without (v1) and with the proposed point refinement (v2). 4DMOS [25] applies sparse 4D convolutions, but on a limited buffer of aggregated, registered past scans.

B. Moving Object Segmentation Performance

In the first experiment, we evaluate how well our approach segments a scan into moving and non-moving points by using a local map of past observations. We show the originally reported baseline results on the SemanticKITTI validation set and the SemanticKITTI MOS benchmark. To provide fair comparisons, we only consider approaches, which are trained and validated on the original SemanticKITTI split. This eliminates the positive bias of using additional training data [8], [36].

We evaluate both the predictions of the current scan (referred to as "Scan") and the volumetric belief with a delay of 10 scans (referred to as "Volumetric Belief"). The choice of 10 scans is an initial estimate that trades off the ability to correct previous wrong estimates and the required waiting time. Besides fusing all scan predictions, we decide to only integrate the local map points that we predict to be moving. This has two reasons: First, we are mainly interested in the moving objects in the local map that we have missed in previous scan predictions. Second, integrating all local map points reduces the runtime of the system. The delay of 10 scans helps to get a more informed belief about the voxels with additional local map predictions before querying their state.

One can see in Tab. I that our volumetric belief helps to improve the results on the validation sequence, whereas the effect is smaller on the test set. We further investigate the effect of the volumetric belief in Sec. IV-D. Our approach outperforms 4DMOS, showing that not limiting past information is beneficial for MOS. In general, we rank second best on the hidden test set and are only outperformed by RVMOS, which requires additional semantic labels for training, while all other approaches just use the moving object labels. Our approach using the volumetric belief achieves the highest result on the validation set with $86.1\,\%$ IoU for the moving points.

Our MOS model runs at 12 Hz for the SemanticKITTI MOS benchmark using an NVIDIA RTX A5000. We implemented the volumetric belief update and querying in C++ and it runs at 44 Hz on a Intel(R) Xeon(R) W-1290P CPU @ 3.70 GHz processor with multi-threading.

C. Generalization Capabilities

The next experiment analyzes how well our approach generalizes to new environments and sensor setups. Since MOS is often a supervised task and labeling is expensive, generalization is an important property. We provide an experiment in Tab. II that realizes different levels of domain shift and compare how well the approaches generalize.

All baselines require external pose information, which we acquire using KISS-ICP for a fair comparison. Note that in the case of 4DMOS, we also report the result of segmenting the most recent scan to compare the online performance before refining with the originally proposed receding horizon strategy and binary Bayes filter. Unfortunately, the code for RVMOS is not publicly available so we cannot run it on these additional datasets.

One can see that the projection-based approaches LMNet and MotionSeg3D perform worse on the highly crowded KITTI Tracking sequence 19. Their performance drops even further on the Apollo dataset. We believe this is because they implicitly overfit to the calibration of the LiDAR sensor, such as mounting location and intensity measurements.

In contrast, 4DMOS and our approach only use the temporal information of the scans and therefore generalize well to a new sensor calibration. We again obtain the best result using our volumetric belief with a delay of 10 scans (referred to as "Volumetric Belief") and outperform 4DMOS.

For the nuScenes dataset, we cannot evaluate the pre-trained models for the projection-based approaches in a fair comparison, because the range image dimensions change due to the different vertical resolutions of the sensors. Both 4DMOS and our approach are still able to segment moving objects, but the average moving IoU is lower. Here, the strategy of 4DMOS shows the best results. When comparing the current scan predictions only, we again achieve a better result in terms of moving IoU.

D. Volumetric Belief

Next, we carry out experiments that show how our proposed volumetric belief can improve moving IoU, recall, and precision. We compare the prediction of our model for the current scan (referred to as "Scan") to our volumetric belief after fusing only the scan prediction (referred to as "Volumetric Belief, Scan Only").

Method	KITTI [13] Tracking 19	Apollo [24]	nuScenes [6] Validation
LMNet [7]	45.3	13.7	n/a
MotionSeg3D, v1 [36]	54,6	6.5	n/a
MotionSeg3D, v2 [36]	54,8	8.8	n/a
4DMOS, delayed [25]	75.5	70.9	**44.8**
4DMOS, online	71.1	68.7	34.6
Ours, Scan	77.0	79.2	36.8
Ours, Volumetric Belief	**78.4**	**81.7**	40.3

TABLE II: Generalization capabilities of different methods on datasets outside of the training distribution. We report the average moving IoU. Best results in bold.

One can see from Tab. III that the probabilistic fusion using a binary Bayes filter consistently increases the precision of our scan prediction by rejecting false positives in previously predicted regions. At the same time, the recall drops due to the discretization error between ground points and the boundary of moving objects. Next, we additionally fuse the local map points that we predict to be moving (referred to as "Volumetric Belief, No Delay"). The results indicate that additionally fusing the local map predictions increases the recall compared to the volumetric belief that only integrates scan predictions.

Our last setup (referred to as "Volumetric Belief") first integrates 10 scan and moving local map predictions into our volumetric belief before querying it for evaluation as explained in Sec. IV-B. This setup again achieves the best result in terms of IoU on most of the sequences since we can now use the local map predictions to identify traces of moving objects and update the volumetric belief accordingly, even if the previous scan-based prediction was static. Solely in the case of Apollo, the setup using the volumetric belief only fusing scan predictions is slightly better in terms of moving IoU. Since the recall of moving objects in the scan predictions is already very high for Apollo, we believe that the negative impact of discretization errors from additionally fusing moving local map points is more dominant in the final IoU than the improvement from correcting false negatives.

E. Online Mapping

Finally, we analyze how we can use our approach and the corresponding volumetric belief for online mapping. We use the VDBFusion [38] library that provides a TSDF-based reconstruction pipeline using the VDB data structure to build a final 3D model [27]. We show the results in Fig. 4 for the CYT_02 sequence [23] (top row) and for the KITTI Tracking sequence 19 (bottom row). The CYT_02 data was obtained with a Livox MID40 scanner, which has a smaller field of view and an irregular sampling pattern compared to rotating 3D LiDARs. This makes it harder to identify moving objects from a limited sequence of frames.

The first column shows the reconstructed surfaces from integrating all scans, including moving points. One can see the traces of moving objects in the map which are undesirable for planning.

	SemanticKITTI [4], [7] Validation 08			KITTI [13] Tracking 19			Apollo [24]			nuScenes [6] Validation		
Method	IoU	R	P	IoU	R	P	IoU	R	P	IoU	R	P
Scan	83.8	87.5	95.3	77.0	**84.6**	89.6	79.2	**93.0**	84.5	36.8	43.4	70.0
Volumetric Belief, Scan Only	84.0	86.4	**96.8**	76.7	80.3	**94.4**	**82.1**	92.3	**88.6**	36.6	40.8	**81.1**
Volumetric Belief, No Delay	83.9	86.7	96.3	76.9	81.8	92.8	81.3	92.4	87.7	36.9	41.7	79.4
Volumetric Belief	**86.1**	**88.7**	**96.8**	**78.4**	83.4	92.9	81.7	92.9	87.7	**40.3**	**45.7**	77.9

TABLE III: Ablation study on average moving IoU, recall (R), and precision (P) in % for our scan-based prediction and different volumetric belief fusion strategies.

The middle column shows the reconstruction after integrating the static predictions from 4DMOS using the receding horizon strategy. Although 4DMOS removes most of the dynamic traces, some moving objects remain in the map, as indicated by the solid markers in Fig. 4. When used with the Livox scanner, 4DMOS removes a lot of static points due to the irregular sampling pattern and the limited number of past scans as encircled by the dashed marker in Fig. 4. We show our final map in the right column.

Based on the high recall achieved in Sec. IV-D, we query the volumetric belief after fusing 10 scan and local map predictions. To additionally handle the discretization error close to the ground as explained in Sec. III-E, we only integrate points for which both the map belief and the corresponding scan prediction are static. By doing so, we can achieve sub-voxel accuracy and even remove moving points that are close to the ground.

V. Conclusion

In this paper, we presented a novel approach to segment moving objects in the current scan and local map. We use a sparse 4D CNN to jointly extract spatio-temporal features based on the discrepancy between scan and map as well as the relative timestamps between points. Additionally, we suggest fusing our predictions into a probabilistic volumetric belief. This allows us to successfully segment moving objects and even recover from false positive predictions. We evaluated our approach on different datasets with different sensor setups and demonstrated its effectiveness and generalization capabilities. Finally, we carried out experiments to evaluate the impact of our volumetric belief and show that it improves the precision and recall of our MOS and can be effectively used to construct a static representation of the environment online.

Acknowledgments

We thank Hyungtae Lim for fruitful discussions and Jiadai Sun for providing baseline results.

References

[1] M. Arora, L. Wiesmann, X. Chen, and C. Stachniss. Mapping the Static Parts of Dynamic Scenes from 3D LiDAR Point Clouds Exploiting Ground Segmentation. In *Proc. of the Europ. Conf. on Mobile Robotics (ECMR)*, 2021.

[2] M. Arora, L. Wiesmann, X. Chen, and C. Stachniss. Static Map Generation from 3D LiDAR Point Clouds Exploiting Ground Segmentation. *Journal on Robotics and Autonomous Systems (RAS)*, 159:104287, 2023.

[3] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, J. Gall, and C. Stachniss. Towards 3D LiDAR-based Semantic Scene Understanding of 3D Point Cloud Sequences: The SemanticKITTI Dataset. *Intl. Journal of Robotics Research (IJRR)*, 40(8–9):959–967, 2021.

[4] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019.

[5] P. Biber and T. Duckett. Dynamic Maps for Long-Term Operation of Mobile Service Robots. In *Proc. of Robotics: Science and Systems (RSS)*, 2005.

[6] H. Caesar, V. Bankiti, A. Lang, S. Vora, V. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuScenes: A Multimodal Dataset for Autonomous Driving. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[7] X. Chen, S. Li, B. Mersch, L. Wiesmann, J. Gall, J. Behley, and C. Stachniss. Moving Object Segmentation in 3D LiDAR Data: A Learning-based Approach Exploiting Sequential Data. *IEEE Robotics and Automation Letters (RA-L)*, 6(4):6529–6536, 2021.

[8] X. Chen, B. Mersch, L. Nunes, R. Marcuzzi, I. Vizzo, J. Behley, and C. Stachniss. Automatic Labeling to Generate Training Data for Online LiDAR-Based Moving Object Segmentation. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):6107–6114, 2022.

[9] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss. SuMa++: Efficient LiDAR-based Semantic SLAM. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[10] C. Choy, J. Gwak, and S. Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[11] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *Intl. Journal of Computer Vision (IJCV)*, 88(2):303–338, 2010.

[12] J. Gehrung, M. Hebel, M. Arens, and U. Stilla. An Approach to Extract Moving Objects From MLS Data Using a Volumetric Background Representation. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-1/W1:107–114, 2017.

[13] A. Geiger, P. Lenz, and R. 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)*, 2012.

[14] M. Henein, G. Kennedy, V. Ila, and R. Mahony. Simultaneous Localization and Mapping with Dynamic Rigid Objects. *arXiv preprint*, arXiv:1805.03800, 2018.

[15] S. Hoermann, M. Bach, and K. Dietmayer. Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[16] A. Hornung, K. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Autonomous Robots*, 34(3):189–206, 2013.

[17] S. Huang, Z. Gojcic, J. Huang, A. Wieser, and K. Schindler. Dynamic 3D Scene Analysis by Point Cloud Accumulation. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2022.

[18] G. Kim and A. Kim. Remove, Then Revert: Static Point Cloud Map Construction Using Multiresolution Range Images. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[19] J. Kim, J. Woo, and Sunghoon. RVMOS: Range-View Moving Object Segmentation Leveraged by Semantic and Motion Features. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):8044–8051, 2022.

[20] T. Kreutz, M. Mühlhäuser, and A.S. Guinea. Unsupervised 4D LiDAR Moving Object Segmentation in Stationary Settings with Multivariate Occupancy Time Series. In *Proc. of the IEEE Winter Conf. on Applications of Computer Vision (WACV)*, 2023.

[21] R. Kümmerle, M. Ruhnke, B. Steder, C. Stachniss, and W. Burgard. A Navigation System for Robots Operating in Crowded Urban Environments. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2013.

[22] H. Lim, S. Hwang, and H. Myung. ERASOR: Egocentric Ratio of Pseudo Occupancy-Based Dynamic Object Removal for Static 3D Point

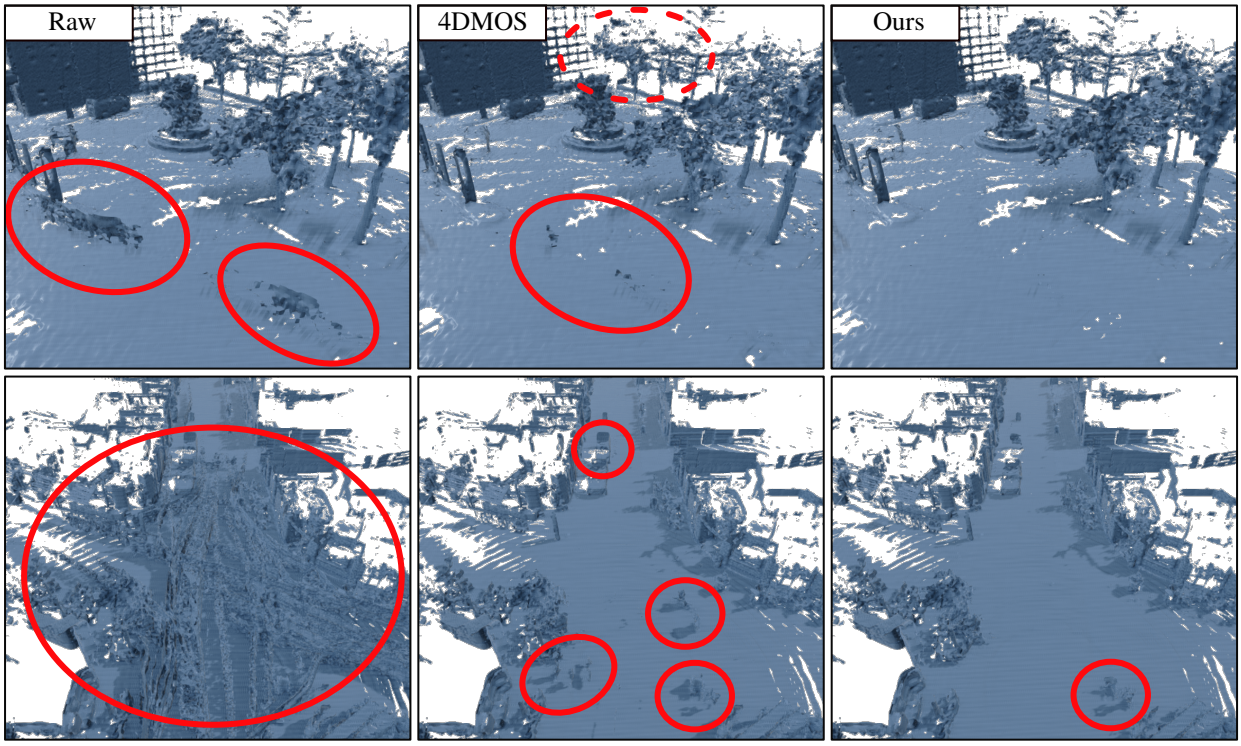

Fig. 4: Reconstructed surfaces obtained with VDBFusion [38] for CYT_02 [23] using a Livox MID40 scanner in the top row and KITTI Tracking [13] sequence 19 in the bottom row. *Left:* Integrating both moving and non-moving points. *Middle:* Integrating static points based on 4DMOS [25] predictions using the receding horizon strategy. *Right:* Integrating static points using our volumetric belief after waiting 10 frames. Solid markers show remaining traces from moving objects in the map and the dashed marker indicates a static area that 4DMOS removed due to false positive predictions.

Cloud Map Building. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):2272–2279, 2021.

[23] J. Lin and F. Zhang. Loam_livox A Robust LiDAR Odemetry and Mapping LOAM Package for Livox LiDAR. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[24] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song. L3-Net: Towards Learning Based LiDAR Localization for Autonomous Driving. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[25] B. Mersch, X. Chen, I. Vizzo, L. Nunes, J. Behley, and C. Stachniss. Receding Moving Object Segmentation in 3D LiDAR Data Using Sparse 4D Convolutions. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7503–7510, 2022.

[26] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss. RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[27] K. Museth, J. Lait, J. Johanson, J. Budsberg, R. Henderson, M. Alden, P. Cucka, D. Hill, and A. Pearce. OpenVDB: An Open-source Data Structure and Toolkit for High-resolution Volumes. In *ACM SIGGRAPH 2013 courses*. 2013.

[28] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3D Reconstruction at Scale using Voxel Hashing. In *Proc. of the SIGGRAPH Asia*, 2013.

[29] D. Nuss, S. Reuter, M. Thom, T. Yuan, G. Krehl, M. Maile, A. Gern, and K. Dietmayer. A Random Finite Set Approach for Dynamic Occupancy Grid Maps with Real-Time Application. *Intl. Journal of Robotics Research (IJRR)*, 37(8):841–866, 2018.

[30] S. Pagad, D. Agarwal, S. Narayanan, K. Rangan, H. Kim, and G. Yalla. Robust Method for Removing Dynamic Objects from Point Clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2020.

[31] P. Pfreundschuh, H.F.C. Hendrikx, V. Reijgwart, R. Dubé, R. Siegwart, and A. Cramariuc. Dynamic Object Aware LiDAR SLAM based on Automatic Generation of Training Data. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2021.

[32] F. Pomerleau, P. Krüsiand, F. Colas, P. Furgale, and R. Siegwart. Long-term 3D Map Maintenance in Dynamic Environments. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2014.

[33] P. Ruchti and W. Burgard. Mapping with Dynamic-Object Probabilities Calculated from Single 3D Range Scans. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[34] J. Schauer and A. Nüchter. The Peopleremover – Removing Dynamic Objects From 3-D Point Cloud Data by Traversing a Voxel Occupancy Grid. *IEEE Robotics and Automation Letters (RA-L)*, 3(3):1679–1686, 2018.

[35] C. Stachniss and W. Burgard. Mobile Robot Mapping and Localization in Non-Static Environments. In *Proc. of the National Conf. on Artificial Intelligence (AAAI)*, 2005.

[36] J. Sun, Y. Dai, X. Zhang, J. Xu, R. Ai, W. Gu, and X. Chen. Efficient Spatial-Temporal Information Fusion for LiDAR-Based 3D Moving Object Segmentation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022.

[37] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[38] I. Vizzo, T. Guadagnino, J. Behley, and C. Stachniss. VDBFusion: Flexible and Efficient TSDF Integration of Range Sensor Data. *Sensors*, 22(3), 2022.

[39] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss. KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way. *IEEE Robotics and Automation Letters (RA-L)*, 8(2):1029–1036, 2023.

[40] L. Wellhausen, R. Dubé, A. Gawel, R. Siegwart, and C. Cadena. Reliable Real-time Change Detection and Mapping for 3D LiDARs. In *Proc. of the IEEE Intl. Sym. on Safety, Security, and Rescue Robotics (SSRR)*, 2017.

[41] K. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: A Probabilistic, Flexible, and Compact 3D Map Representation for Robotic Systems. In *Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation, IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2010.

[42] D. Yoon, T. Tang, and T. Barfoot. Mapless Online Detection of Dynamic Objects in 3D Lidar. In *Proc. of the Conf. on Computer and Robot Vision (CRV)*, 2019.

[43] N. Zimmerman, L. Wiesmann, T. Guadagnino, T. Läbe, J. Behley, and C. Stachniss. Robust Onboard Localization in Changing Environments Exploiting Text Spotting. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022.

CERTIFICATE OF REPRODUCIBILITY

The authors of this publication declare that:

1) The software related to this publication is distributed in the hope that it will be useful, support open research, and simplify the reproducability of the results but it comes without any warranty and without even the implied warranty of merchantability or fitness for a particular purpose.

2) *Benedikt Mersch* primarily developed the implementation related to this paper. This was done on Ubuntu 20.04.

3) *Tiziano Guadagnino* verified that the code can be executed on a machine that follows the software specification given in the Git repository available at:

   https://github.com/PRBonn/MapMOS

4) *Tiziano Guadagnino* verified that the experimental results presented in this publication can be reproduced using the implementation used at submission, which is labeled with a tag in the Git repository and can be retrieved using the command:

   ```
   git checkout mersch2023ral
   ```