Reliable Change Detection for Long-Term LiDAR Mapping in Transient Environments

Seoyeon Jang[1] **Alex Junho Lee**[2] **I Made Aswin Nahrendra**[1] **Hyun Myung**[1]
[1]Korea Advanced Institute of Science and Technology [2]Sookmyung Women's University
{9uantum01, anahrendra, hmyung}@kaist.ac.kr,
alexlee@sookmyung.ac.kr

Abstract: Online change detection is crucial for mobile robots to efficiently navigate through dynamic environments. Detecting changes in transient settings, such as active construction sites or frequently reconfigured indoor spaces, is particularly challenging due to frequent occlusions and spatiotemporal variations. Existing approaches often struggle to detect changes and fail to update the map across different observations. To address these limitations, we propose a dual-head network designed for online change detection and long-term map maintenance. A key difficulty in this task is the collection and alignment of real-world data, as manually registering structural differences over time is both labor-intensive and often impractical. To overcome this, we develop a data augmentation strategy that synthesizes structural changes by importing elements from different scenes, enabling effective model training without the need for extensive ground-truth annotations. Experiments conducted at real-world construction sites and in indoor office environments demonstrate that our approach generalizes well across diverse scenarios, achieving efficient and accurate map updates.

Keywords: Change detection, Data Augmentation, Long-term Map Management

1 Introduction

Online change detection is the task of identifying structural discrepancies during robot operation between a previously constructed map and the current sensor observation. In long-term environmental monitoring or inspection, this capability is crucial for maintaining an accurate understanding of the environment, especially in dynamic settings such as ongoing construction sites or disaster-affected areas. Unlike open environments where main structures generally remain visible, major robot service spaces such as crowded indoor environments involve moving objects, frequent occlusions, and significant structural changes. Online map alignment is particularly important in these contexts, especially when multiple robots are sharing information or when a robot revisits a location after an extended period. In such cases, aligning current observations with previously collected data is necessary to ensure consistent localization and mapping across different times and agents. To properly operate in these applications, spatiotemporal changes must be recognized at scan time, enabling robots to dynamically adjust their behavior in response to the evolving environment.

Changes can be categorized into two types based on their behavior within the period of a single observation: high-dynamic (HD) changes and low-dynamic (LD) changes. HD changes refer to inconsistencies that occur during the observation window, such as walking pedestrians or moving vehicles, whereas LD changes correspond to structural modifications that remain static within a single scan but differ across sessions, as illustrated in Fig. 1. LD changes can be further classified into positive changes (PC), where new structures appear, and negative changes (NC), where previously existing elements disappear.

While extensive research has been conducted on HD change detection [1, 2, 3, 4], LD change detection has received limited attention primarily because it hardly affects local odometry, and moreover, it is difficult to analyze. As relocalization becomes less critical with accurate odometry, prior work has largely focused on handling HD changes. However, even after removing HD changes, unlabeled

LD changes can drive the map unreliable when observations are accumulated over multiple sessions, degrading map quality for localization. To address this problem, we propose a method to detect LD changes and update the prior map to maintain long-term consistency.

LD change detection has been studied using both 2D images [5, 6, 7] and 3D range sensors [8, 9, 10, 11, 12]. Among these, 3D sensors are particularly suitable for capturing structural information and supporting reliable map management in dynamic environments. Classical 3D range sensor-based change detection methods primarily rely on geometric differences between two observations. However, geometry-based methods are generally more vulnerable to occlusion. Moreover, despite potential parallelization, classical methods still suffer from scalability issues as the environment size increases, mainly due to the growing number of points that must be compared.

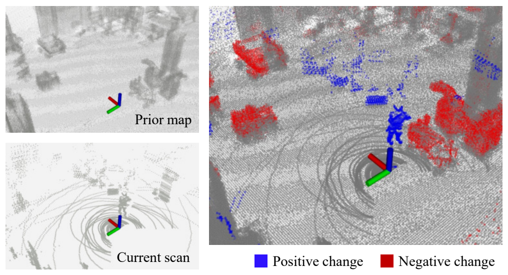

Figure 1: Our method detects low dynamic (LD) changes between the prior map and the current scan. Red: negative changes (NC, disappeared); Blue: positive changes (PC, newly appeared).

Recently, deep learning-based approaches have been actively explored for faster inference than classical methods, using semantic features in change detection [13, 14, 15, 16, 17, 18]. However, they still face fundamental challenges, particularly in long-term settings, including (1) the difficulty of collecting large-scale training datasets that reflect long-term differences, and (2) the challenge of distinguishing between occlusions and true changes.

In this paper, we address these challenges by designing a learning framework that explicitly separates occlusion and disappearance, and by constructing a data generation pipeline tailored to long-term environmental variations. Based on this idea, we propose **Chamelion**, a novel framework for **Cha**nge detection and long-term **M**ap management in transient **E**nvironments, using **Li**DAR and designed for **On**line operation. Our main contributions can be summarized as follows:

First, to address the difficulty of obtaining large-scale long-term change datasets, we propose a composition-based augmentation strategy that synthetically generates pseudo-changes from single-session scans. Second, we design a novel network architecture that employs a 4D CNN backbone [19] and a dual-head structure, consisting of a class head for change classification and a confidence head for occlusion awareness, to enhance generalization and robustness. Finally, through extensive experiments on real-world and synthetic datasets, we show that our method outperforms existing approaches in change detection performance and enables effective online map updates.

2 Related Works

2.1 Change Detection Dataset Generation

Generating 3D change detection datasets is challenging due to the need for multiple observations of the same environment over time and accurate change annotations. Manual data collection and labeling, typically performed in small indoor environments, become impractical for large-scale environments such as construction sites [9, 20, 21]. To address these limitations, simulation-based datasets have been introduced. For example, Park *et al.* [22] proposed the ChangeSim dataset for industrial indoor environments, and Joseph *et al.* [13] presented the LiSTA dataset featuring LiDAR scans in office spaces. However, even simulation-based datasets still require multiple acquisitions of the same environment to simulate temporal differences. Data synthesis techniques have been explored in the 2D domain to overcome this challenge by generating change detection datasets from a single image. Park *et al.* [7] improved 2D change detection by creating synthetic samples through random warping and cut-and-paste operations. Inspired by these techniques, we propose a composition-based augmentation strategy that enables the generation of pseudo-change datasets using only single-session LiDAR scans, eliminating the need for multiple observations over time.

2.2 LiDAR-based Change Detection

Traditional 3D change detection methods primarily rely on geometric differences across different sessions, such as occupancy changes [8], TSDF differences [9, 10] or visibility changes [11, 12]. However, they are sensitive to environmental variations, and their computational cost increases significantly as the number of point clouds grows. Recently, deep learning-based approaches aim to address some of this scalability issues. Supervised learning approaches rely on labeled datasets, but generating large-scale annotated data is time-consuming and labor-intensive[14]. Self-supervised methods have been proposed to reduce label dependency[15], yet they still require multiple observations of the same environment. The other approaches [17, 18] attempts to improve efficiency by converting 3D LiDAR data into 2D range images as network inputs, enabling real-time performance. However, these methods heavily rely on omnidirectional LiDAR and are less generalizable to other types of range sensors or diverse environments. Furthermore, occlusion handling remains an open challenge, requiring additional pre-processing or post-processing steps [13], which increase pipeline complexity.

In this paper, we propose a novel LiDAR-based online change detection framework that explicitly separates true changes from occlusions, generalizes across different types of 3D range sensors, and eliminates the need for extensive pre- or post-processing.

3 Our Approach for Online Change Detection

To address the challenges in long-term LiDAR-based change detection, we propose a novel framework that (1) generates training data from a single session without requiring multiple temporal observations, (2) explicitly models occlusion uncertainty through cross-visibility confidence estimation, and (3) updates the prior map probabilistically based on confidence-aware change detection.

This section details each component of our framework: First, we introduce a composition-based augmentation strategy to generate pseudo-change data from a single session (Section 3.1). Next, we describe our 4D sparse convolutional backbone and dual-head structure, which jointly predict changes and estimate cross-visibility confidence (Sections 3.2 and 3.3). Finally, we present a confidence-aware, probabilistic update scheme for online map maintenance (Section 3.4).

3.1 Composition-based Data Augmentation for Single-Session Pseudo-Label Generation

Training change detection requires multi-session data, which is costly and impractical to collect. To address this, we propose a single-session augmentation method that synthetically generates change pseudo-labels using only a single traversal of the environment, as illustrated in Fig. 2. We start by constructing a static map from single-session LiDAR scans.

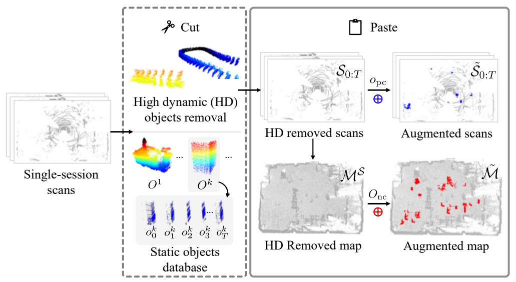

Figure 2: Our proposed composition-based augmentation for the pseudo-label generation.

Let $\mathcal{S}_i^{\text{local}}$ denote the ith LiDAR scan at timestamp t_i in the sensor coordinate frame, and $\mathcal{S}_i$ denote its transformation into the global coordinate frame with its estimated pose G_i, such that $\mathcal{S}_i = G_i * \mathcal{S}_i^{\text{local}}$, where $*$ denotes the transformation operation applied to all points of $\mathcal{S}_i^{\text{local}}$. The global map $\mathcal{M}^{\mathcal{S}}$ is then constructed by taking the union of all transformed scans up to time t_T:

$$\mathcal{M}^{\mathcal{S}} = \bigcup_{i=0}^{T} \mathcal{S}_i. \tag{1}$$

To ensure that the map reflects only the static elements of the environment, we apply moving object segmentation (MOS) [23] to remove high-dynamic (HD) objects from each scan. After filtering out HD objects, we save the point cloud of the j-th snapshot of the k-th static object, denoted as o_j^k ($j = 0, \ldots, T$), and construct a set of static object snapshots O^k. Finally, the collection of all

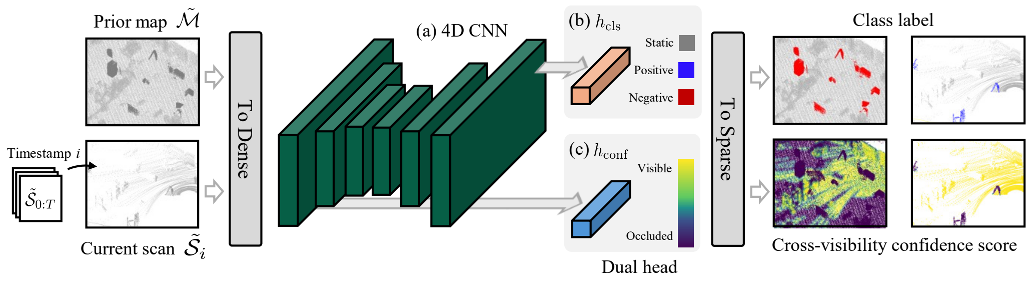

Figure 3: **Our Dual-head architecture.** (a) 4D-CNN feature extractor; (b) change classification; (c) cross-visibility confidence estimation. Map updates use class predictions only in high cross-visibility regions.

static object snapshots is stored as a database $\mathcal{DB}$, defined as:

$$O^k = \{o_j^k \mid j = 0, \dots, T\}, \tag{2}$$

$$\mathcal{DB} = \{O^k \mid k = 1, \dots, N_T\}, \tag{3}$$

where N_T is the number of static objects tracked up to time t_T.

Using this database, we generate training samples with change pseudo-labels by randomly sampling static objects and inserting them into scans and maps. Since the map is a dense accumulation of multiple scans, while each individual scan remains sparse, the insertion strategy differs between the two. Specifically, we insert only single-frame object instances (o_j^k) into scans, and all accumulated snapshots of a tracked object (O^k) into maps. Pseudo-labels are then assigned based on object placement: objects added to the scan but absent from the map are labeled as positive changes ($o_j^k \rightarrow o_{\text{pc}}$, $O^k \rightarrow O_{\text{pc}}$), whereas objects added to the map but absent from the scan are labeled as negative changes ($o_j^k \rightarrow o_{\text{nc}}$, $O^k \rightarrow O_{\text{nc}}$). By expressing the scans accumulated up to timestamp t_T as $\mathcal{S}_{0:T} = \{\mathcal{S}_i \mid i = 0, \dots, T\}$, the augmented map ($\tilde{\mathcal{M}}$) and scans ($\tilde{\mathcal{S}}_{0:T}$) with randomly placed objects are defined as:

$$\tilde{\mathcal{M}} = \mathcal{M}^{\mathcal{S}} \oplus O_{\text{nc}}, \tag{4}$$

$$\tilde{\mathcal{S}}_{0:T} = \{\mathcal{S}_i \oplus o_{\text{pc}} \mid o_{\text{pc}} \in O_{\text{pc}}, i = 0, \dots, T\}, \tag{5}$$

where the operator $\oplus$ denotes object insertion into a scan or a map. To avoid any overlap between inserted objects, we assume $O_{\text{pc}} \cap O_{\text{nc}} = \varnothing$. As a result, $\tilde{\mathcal{M}}$ and $\tilde{\mathcal{S}}_{0:T}$ serve as synthetic multi-session data, enabling effective training of change detection models.

3.2 4D Sparse Convolutional Backbone for Changed Feature Extraction

To effectively capture both spatial and temporal features in sparse LiDAR data, we employ a 4D sparse convolutional neural network based on the MinkowskiEngine [19]. The map $\tilde{\mathcal{M}} \in \mathbb{R}^{m \times 3}$ and the scan $\tilde{\mathcal{S}}_i \in \mathbb{R}^{n \times 3}$, sampled from $\tilde{\mathcal{S}}_{0:T}$, are labeled with a visibility flag ν: 0 for all map points and 1 for all scan points. Here, m and n are the total number of points from the map and scan, respectively. As a result, the two point sets are extended to tensors of size $m \times 4$ and $n \times 4$, where each point is represented as a 4D vector of (x, y, z) coordinates and a visibility flag ν.

The two sets are then concatenated into a 4D dense tensor: $\mathcal{T} \in \mathbb{R}^{(m+n) \times 4}$. This dense tensor is then discretized through a quantization process, resulting in a 4D sparse tensor $\mathcal{T}' \in \mathbb{R}^{(m'+n') \times 4}$, where m' and n' represent the number of non-empty voxels corresponding to the map and scan points, respectively. The sparse tensor serves as the input to the backbone network. The network outputs a set of feature representations, denoted as $F \in \mathbb{R}^{(m'+n') \times D}$, where D is the feature dimension.

3.3 Dual-head Structure for Change Classification and Cross-visibility Confidence Estimation

As illustrated in Fig. 3, we introduce a dual-head structure designed to jointly handle change classification and cross-visibility confidence estimation. The dual-head structure consists of a class head and a confidence head. The class head predicts whether each point is a change or static, while the confidence head estimates the probability that the point is visible in both the map and scan.

The class head is trained using a cross-entropy loss:

$$\mathcal{L}_{\text{cls}} = -\frac{1}{m+n} \sum_{j=1}^{m+n} \sum_{c=0}^{2} y_{j,c} \log(\hat{y}_{j,c}), \tag{6}$$

where $y_{j,c}$ and $\hat{y}_{j,c}$ denote the ground truth and predicted class probabilities for the j-th point, respectively. Here, the class label $c \in \{0, 1, 2\}$ indicates static (0), a positive change (1), or a negative change (2).

Change classification with only the class head often yields inaccurate results, especially for map-wise detection due to occlusions. As shown in Fig. 4(a), the scan often covers only a partial view of the map. This limited coverage can cause occlusion, so that some regions in the map are not observed in the scan. Consequently, it becomes difficult to determine whether the observed negative change is an actual environmental change or a misclassification caused by occlusion.

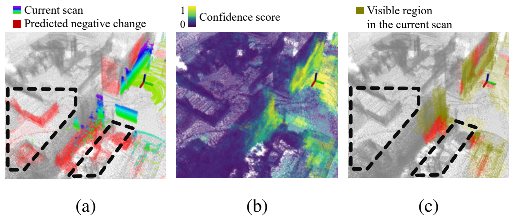

Figure 4: Occlusion handling with cross-visibility confidence. gray: static; red: negative change. (a) Occlusions (black dashed) near walls; (b) visibility from the confidence head; (c) apply class only in visible regions, reducing errors.

To resolve this ambiguity, we introduce a confidence head that estimates cross-visibility confidence for each point. The cross-visibility confidence indicates the probability that a point is visible in both the map and scan. High-confidence points are reliably observed in both, while low-confidence points are likely occluded in one. Although explicit point-to-point distance checks could also estimate visibility, they require costly nearest-neighbor searches. Our confidence head instead provides a lightweight learned approximation that captures both geometric and contextual cues without this overhead.

Let p_j and p_k denote the nearest points in the map and scan, and let d_{jk} be their Euclidean distance. We define the ground truth confidence $c(d_{jk})$ using a exponential decay function:

$$c(d_{jk}) = \begin{cases} 0, & \text{if } d_{jk} \geq \tau_{\text{ocl}}, \\ \exp\left(-\lambda \cdot (d_{jk} - \tau_{\text{vox}})\right), & \text{if } \tau_{\text{vox}} < d_{jk} < \tau_{\text{ocl}}, \\ 1, & \text{if } d_{jk} \leq \tau_{\text{vox}}, \end{cases} \tag{7}$$

where λ represents the exponential decay rate, τ_{vox} is the voxel size, and τ_{ocl} is the occlusion distance threshold.

To improve training stability and convergence speed, we truncate the decay at τ_{ocl} instead of applying it across the full range of d_{jk}. The confidence head is trained using mean squared error (MSE) loss:

$$\mathcal{L}_{\text{conf}} = \frac{1}{|\mathcal{P}|} \sum_{(j,k)\in\mathcal{P}} \left\| c(d_{jk}) - \hat{c}_j \right\|^2, \tag{8}$$

where $\mathcal{P}$ is a set of randomly sampled point pairs (p_j, p_k) from the map and scan, with $|\mathcal{P}|$ denoting the number of sampled pairs. $\hat{c}_j$ is the predicted confidence value from the confidence head.

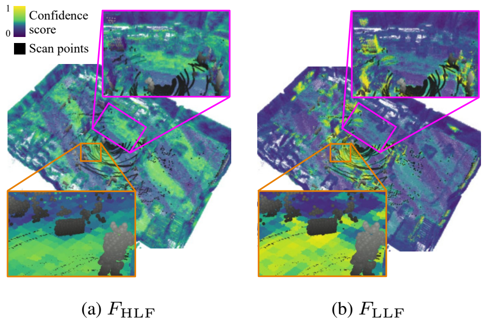

Figure 5: Comparison between confidence head outputs trained with high-level (F_{HLF}) and low-level (F_{LLF}) backbone features.

During training, the class and confidence heads are provided with distinct input features tailored to their respective objectives. Since change classification requires high-level semantic understanding, we feed the final backbone feature map F_{HLF}

as input to the class head. In contrast, cross-visibility estimation relies more on local geometric cues, so we use earlier-stage features F_{LLF} for the confidence head. As shown in Fig. 5, using low-level features significantly improves occlusion detection performance. The total training loss is the weighted sum of the classification loss and the confidence loss, $\mathcal{L} = \mathcal{L}_{\mathrm{cls}} + \alpha\mathcal{L}_{\mathrm{conf}}$. The confidence loss weight α balances the contributions of the main task of change classification and the auxiliary supervision of confidence scores.

3.4 Probabilistic Change Integration and Prior Map Update

In an online map update scenario, our method yields N change detection results on the same prior map for N current observations. To build a consistent map representation, we propose a recursive Bayesian filtering framework to integrate these results and incrementally update the map over time. A key aspect of our approach is that map updates are conditioned on cross-visibility confidence: regions with low cross-visibility are more prone to occlusion-induced errors and therefore are excluded from the update process.

Let $l(\hat{y}_j \mid \mathcal{M}_{0:t})$ denote the log-odds of the predicted class $\hat{y}_j$ at a map point after integrating observations up to time t. The update process is defined as follows:

$$l(\hat{y}_j \mid \mathcal{M}_{0:t}) = \begin{cases} l(\hat{y}_j \mid \mathcal{M}_{0:t-1}) + l(\hat{y}_j \mid \mathcal{M}_t) - l(\hat{y}_j), & \text{if } \hat{c}_j > \tau_{\mathrm{conf}}, \\ l(\hat{y}_j), & \text{otherwise} \end{cases} \tag{9}$$

where $l(\hat{y}_j \mid \mathcal{M}_t)$ is the log-odds derived from the observation at time t, $l(\hat{y}_j)$ is the prior class probability, $\hat{c}_j$ is the predicted confidence score, and τ_{conf} is the confidence threshold for accepting an observation.

4 Experiments

4.1 Experimental Setup

The purpose of our experiments was to evaluate our method's performance in change detection and its effectiveness in updating the prior map.

Datasets. We used two datasets for quantitative evaluation. The first is our custom dataset, which consists of a prior map built in August 2024 and 1,000 scan frames collected in October 2024 from three different environments, including construction sites and a laboratory. As shown in Fig. 6, it contains various low dynamic (LD) changes. The ground truth change labels were manually annotated. To assess the generalization capability of our approach, we also used the LiSTA dataset [13], a changing indoor office dataset, also annotated with ground truth change labels.

Comparison methods. We compared our method with both classical and learning-based baselines: occupancy-based [8], MapMOS [24], and SPS [16]. Especially, we re-trained MapMOS on our pseudo-labeled data, as the original model was trained solely on the SemanticKITTI-MOS dataset.

Evaluation metric. We evaluate the performance of each method in both scan-wise and map-wise change detection. Scan-wise positive change detection is evaluated using the intersection over union (IoU), defined as:

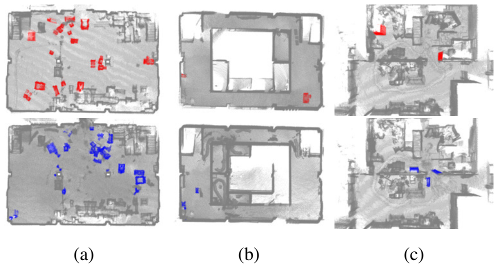

Figure 6: Description of the custom change detection dataset. (a) `Const-1F`, (b) `Const-2F`, (c) `Lab`.

$$\mathrm{IoU}_{\mathrm{PC}} = \frac{\mathrm{TC}}{\mathrm{TC} + \mathrm{FC} + \mathrm{FS}}, \tag{10}$$

where TC, FC, and FS denote the true changes, false changes, and false statics, respectively. For map-wise evaluation of negative change detection, we use the preservation rate (PR) and rejection rate (RR), following the metrics modified from [2]:

Table 1: Change detection performance comparison on the custom dataset in terms of scan-wise IoU and map-wise PR, RR, and F_1 scores. Best results in **bold**, second best in a gray background.

Seq.	Method	Scan-wise	Map-wise		
		IoU ↑	PR ↑	RR ↑	F_1 ↑
`Const-1F`	Occupancy [8]	0.297	0.877	0.715	0.787
	MapMOS [24]	0.610	**0.976**	0.201	0.334
	SPS [16]	0.433	0.173	**0.999**	0.294
	Chamelion (ours)	**0.660**	0.948	0.904	**0.925**
`Lab`	Occupancy [8]	0.259	0.910	0.775	0.837
	MapMOS [24]	0.745	**0.992**	0.130	0.230
	SPS [16]	0.333	0.094	**0.998**	0.171
	Chamelion (ours)	**0.770**	0.962	0.785	**0.865**

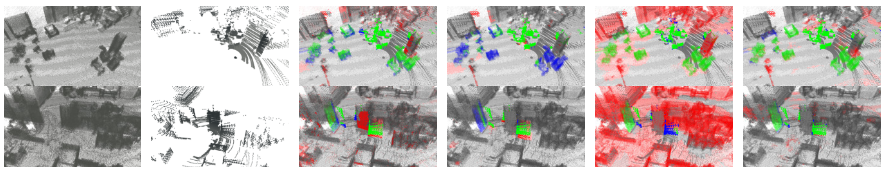
(a) Prior map (b) Current scan (c) Occupancy [8] (d) MapMOS [24] (e) SPS [16] (f) Ours

Figure 7: Qualitative comparison on our dataset. Green = true changes, red = false changes, blue = false statics. The fewer red and blue points, the better the result.

- $\mathrm{PR} = \frac{\text{\# of preserved static points}}{\text{\# of total static points on the prior map}}$,
- $\mathrm{RR} = 1 - \frac{\text{\# of remaining negative changes}}{\text{\# of total negative changes on the prior map}}$,
- $\mathrm{F}_1 = 2\mathrm{PR} \cdot \mathrm{RR}/(\mathrm{PR} + \mathrm{RR})$.

We adopt different metrics for evaluating the map and scans, since the scans consist of multiple frames, whereas the map is a single, accumulated dataset. Applying identical metrics would be inappropriate due to this structural difference.

Implementation details. For training, we augment a custom dataset, collected from a different environment than the evaluation set, using the method in Section 3.1, with a voxel size of 0.1m used for input quantization. The confidence loss weight α is set to 0.01. For confidence score calculation (see (7)), we set $\tau_{\mathrm{ocl}} = 3.0\,\mathrm{m}$ and $\lambda = 10$. Training is performed for up to 50 epochs with a batch size of 2 using the Adam optimizer [25], and the best validation model is selected for evaluation. During inference, regions in scans are classified as *changes* if they are predicted as positive changes and have confidence scores below the threshold $\tau_{\mathrm{conf}} = 0.8$. For the map, the probability of negative change is recursively updated only in regions with confidence scores above $\tau_{\mathrm{conf}} = 0.5$.

4.2 Evaluation of Change Detection Performance in Real-World Construction Sites

In scan-wise evaluation, occupancy-based method recorded the lowest IoU scores across all three sequences, as shown in Table 1. As illustrated in Fig. 7(c), occupancy [8] methods include a significant number of false change points. This result demonstrates that geometric threshold-based change detection methods are vulnerable to data occlusion and noise. In contrast, deep learning-based approaches exhibit robust performance in scan-wise change detection, as they do not rely on geometric discrepancy. However, since SPS is purely based on a regression network, its change detection boundaries are vague and highly sensitive to threshold selection. It is evidenced by the presence of false change points near ground-level regions, as shown in the first row of Fig. 7(e). In contrast, our method and MapMOS, which use classification-based change detection, outperform SPS in the scan-wise IoU.

In the map-wise evaluation, both SPS and MapMOS underperformed compared with ooccupancy-based method. SPS computes stability score only in voxels with scan points, ignoring others during training, and leads to inaccurate predictions. MapMOS also classifies changes only where scan points exist, thus often fails in unobserved regions. In contrast, our method leverages both map and scan points with cross-visibility, and refines the map using a Bayes filter conditioned on visibility confidence, thus enabling more accurate change detection by avoiding updates in low-confidence regions.

Table 2: Change detection performance comparison on the LiSTA dataset in terms of scan-wise IoU and map-wise PR, RR, and F_1 scores. Best results in **bold**, second best in a gray background.

Seq.	Method	Scan-wise	Map-wise		
		IoU ↑	PR ↑	RR ↑	F_1 ↑
`Simu-1`	Occupancy [8]	0.240	0.962	**0.912**	**0.936**
	MapMOS [24]	**0.711**	**0.998**	0.140	0.246
	SPS [16]	0.274	0.284	0.899	0.432
	Chamelion (ours)	0.709	0.964	0.900	0.931
`Simu-3`	Occupancy [8]	0.156	0.954	0.780	**0.858**
	MapMOS [24]	0.518	**0.998**	0.027	0.053
	SPS [16]	0.283	0.254	**0.957**	0.402
	Chamelion (ours)	**0.577**	0.919	0.710	0.801

4.3 Evaluation of Efficient Map Update

To evaluate the efficiency of our map update, we analyze the relationship between map similarity and reused points ratio. Map similarity is a metric that represents how similar the updated prior map ($\mathcal{M}'$) is to the current map ($\mathcal{M}_S$). Here, $\mathcal{M}'$ is obtained by integrating both negative changes (NC) and positive changes (PC) into the prior map ($\mathcal{M}$) as $\mathcal{M}' = \mathcal{M} \ominus \text{NC} \oplus \text{PC}$, where the operator $\oplus$ denotes object insertion into the map and $\ominus$ denotes object removal.

We define similarity as the ratio of points in $\mathcal{M}_S$ whose chamfer distance to $\mathcal{M}'$ is less than the voxel size, to the total number of points in $\mathcal{M}_S$. Both maps are voxelized with the same resolution for fair comparison.

The reused points ratio is defined as the proportion of predicted static points that are present in both $\mathcal{M}$ and $\mathcal{M}_S$, divided by the number of ground-truth static points in the same sets. It indicates how effectively static points from the prior map and current scans can be reused. A high similarity but low reused ratio means unnecessary updates, while the opposite indicates under-updating. As shown in Fig. 8, our method avoids redundant updates of the static points that are already present in the map, reflects only the detected changes, and maintains consistency with up-to-date observations.

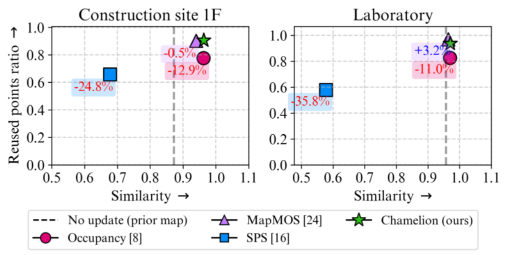

Figure 8: Map similarity and reused points ratio (red: lower than ours; blue: higher). Black dashed lines: similarity between $\mathcal{M}$ and $\mathcal{M}^S$.

4.4 Generalization Performance and Runtime

As shown in Table 3, we evaluated the generalization performance of our method with the LiSTA dataset. Both our method and MapMOS maintained robust scan-wise performance, while our method achieved higher preservation but lower rejection rates in map-wise evaluation due to the dataset's sparse and discontinuous observations. This effect is less critical in real-world settings with frequent observations. To demonstrate our method's online operability, we measured the average model runtime on our custom dataset. Our network processed $25 \times 25\,\text{m}$ maps in 0.0686 s on the NVIDIA RTX 3060, demonstrating that the proposed method is suitable for online operation.

5 Conclusion

In this paper, we introduced Chamelion, a novel approach for online change detection and long-term 3D map management. Our dual-head network structure is designed to robustly detect changes, even with occlusion, and it can be trained using pseudo change datasets generated by composition-based augmentation. Moreover, our method exhibits robust performance across various 3D range sensors and environmental settings, demonstrating its generalization capability. Despite these encouraging results, there is further space for improvements. Currently, we assume that low-dynamic changes either fully disappear or appear as new objects, limiting the ability to detect changes involving object movement or replacement. In future work, we aim to extend our approach to handle object movement or replacement more effectively.

Acknowledgments

This work was supported by the Technology Commercialization Support Program, through the Korea Innovation Foundation funded by the Ministry of Science and ICT, and by Korea Evaluation Institute of Industrial Technology (KEIT) grant funded by the Korea government (MOTIE) (No.20023455, Development of Cooperate Mapping, Environment Recognition and Autonomous Driving Technology for Multi Mobile Robots Operating in Large-scale Indoor Workspace).

References

[1] G. Kim and A. Kim. Remove, then revert: Static point cloud map construction using multiresolution range images. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 10758–10765, 2020.

[2] H. Lim, S. Hwang, and H. Myung. ERASOR: Egocentric ratio of pseudo occupancy-based dynamic object removal for static 3D point cloud map building. *IEEE Robot. Automat. Lett.*, 6 (2):2272–2279, 2021.

[3] 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 Robot. Automat. Lett.*, 7 (3):7503–7510, 2022.

[4] 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 Robot. Automat. Lett.*, 6:6529–6536, 2021.

[5] K. Sakurada, M. Shibuya, and W. Weimin. Weakly supervised silhouette-based semantic scene change detection. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 6861–6867, 2020.

[6] S. Chen, K. Yang, and R. Stiefelhagen. DR-TANet: Dynamic receptive temporal attention network for street scene change detection. In *Proc. IEEE Intell. Veh. Symp.*, pages 502–509, 2021.

[7] J. Park, U. Kim, S. Lee, and J. Kim. Dual task learning by leveraging both dense correspondence and mis-correspondence for robust change detection with imperfect matches. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 13749–13759, 2022.

[8] A. Walcott-Bryant, M. Kaess, H. Johannsson, and J. J. Leonard. Dynamic pose graph SLAM: Long-term mapping in low dynamic environments. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 1871–1878, 2012.

[9] M. Fehr, F. Furrer, D. Ivan, J. Sturm, I. Gilitschenski, R. Siegwart, and C. Cadena. TSDF-based change detection for consistent long-term dense reconstruction and dynamic object discovery. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 5237–5244, 2017.

[10] L. Schmid, J. Delmerico, J. L. Schönberger, J. Nieto, M. Pollefeys, R. Siegwart, and C. Cadena. Panoptic Multi-TSDFs: A flexible representation for online multi-resolution volumetric mapping and long-term dynamic scene consistency. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 8018–8024, 2022.

[11] J. P. Underwood, D. Gillsjö, T. Bailey, and V. Vlaskine. Explicit 3D change detection using ray-tracing in spherical coordinates. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 4735–4741, 2013.

[12] G. Kim and A. Kim. LT-mapper: A modular framework for LiDAR-based lifelong mapping. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 7995–8002, 2022.

[13] R. Joseph, Z. Lintong, and F. Maurice. LiSTA: Geometric object-based change detection in cluttered environments. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 7510–7517, 2024.

[14] T. Ku, S. Galanakis, B. Boom, R. C. Veltkamp, D. Bangera, S. Gangisetty, N. Stagakis, G. Arvanitis, and K. Moustakas. SHREC 2021: 3D point cloud change detection for street scenes. *Computers and Graphics*, 99:192–200, 2021.

[15] I. Hroob, S. Molina, R. Polvara, G. Cielniak, and M. Hanheide. LTS-NET: End-to-end unsupervised learning of long-term 3D stable objects. In *Proc. Intelligent Autonomous Systems 18*, pages 17–29, 2024.

[16] I. Hroob, B. Mersch, C. Stachniss, and M. Hanheide. Generalizable stable points segmentation for 3D LiDAR scan-to-map long-term localization. *IEEE Robot. Automat. Lett.*, 9(4):3546–3553, 2024.

[17] A. Krawciw, J. Sehn, and T. D. Barfoot. Change of scenery: Unsupervised LiDAR change detection for mobile robots. *arXiv preprint arXiv:2309.10924*, 2023.

[18] A. Krawciw, S. Lilge, and T. D. Barfoot. LaserSAM: Zero-shot change detection using visual segmentation of spinning LiDAR. *arXiv preprint arXiv:2402.10321*, 2024.

[19] C. Choy, J. Gwak, and S. Savarese. 4D Spatio-temporal convnets: Minkowski convolutional neural networks. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 3075–3084, 2019.

[20] E. Langer, T. Patten, and M. Vincze. Robust and efficient object change detection by combining global semantic information and local geometric verification. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 8453–8460, 2020.

[21] J. Wald, A. Avetisyan, N. Navab, F. Tombari, and M. Niessner. RIO: 3d object instance re-localization in changing indoor environments. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 7657–7666, 2019.

[22] J.-M. Park, J.-H. Jang, S.-M. Yoo, S.-K. Lee, U.-H. Kim, and J.-H. Kim. ChangeSim: Towards end-to-end online scene change detection in industrial indoor environments. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pages 8578–8585, 2021.

[23] S. Jang, M. Oh, B. Yu, I. Nahrendra, S. Lee, H. Lim, and H. Myung. TOSS: Real-time tracking and moving object segmentation for static scene mapping. In *Proc. Int. Conf. Robot Intell. Tech. Appl.*, pages 7255–7262, 2023.

[24] B. Mersch, T. Guadagnino, X. Chen, I. Vizzo, J. Behley, and C. Stachniss. Building volumetric beliefs for dynamic environments exploiting map-based moving object segmentation. *IEEE Robot. Automat. Lett.*, 8(8):5180–5187, 2023.

[25] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

A Supplementary Material

To further analyze the effect of dataset-specific fine-tuning, we additionally conducted experiments on the LiSTA dataset [13]. We considered two fine-tuning strategies:

- Pseudo-label fine-tuning: We generated pseudo-labels on the first session of the Simu-1 sequence using our composition-based data augmentation introduced in Section 3.1.
- GT-based fine-tuning: We directly used the ground-truth labels of the Simu-1 sequence for training, and evaluated on the Simu-2 and Simu-3 sequences.

The results are summarized in Table 3. Interestingly, the performance of fine-tuning on LiSTA was comparable to that of fine-tuning on our custom dataset, suggesting that dataset-specific fine-tuning does not necessarily yield additional gains. This implies that the benefit of environment-specific fine-tuning can be limited, while also demonstrating the robustness of our pseudo-label generation scheme. Notably, GT-based fine-tuning even resulted in worse performance in both scan-wise and map-wise evaluations. We attribute this to the greater diversity provided by our composition-based data augmentation. Unlike ground-truth labels that reflect fixed, predefined locations, our approach introduces changes at flexible and varied positions. This results in richer and more scalable training signals compared to the constrained ground-truth annotations.

Table 3: Evaluation of dataset-specific fine-tuning on LiSTA in terms of scan-wise IoU and map-wise PR, RR, and F_1. Best results are in **bold**.

Seq.	Training data	Scan-wise	Map-wise		
		IoU ↑	PR ↑	RR ↑	F_1 ↑
`Simu-2`	No fine-tuning	0.706	0.845	**0.754**	**0.797**
	Simu-1 (Pseudo-label)	**0.726**	0.647	0.722	0.682
	Simu-1 (Ground-truth)	0.444	**0.975**	0.178	0.302
`Simu-3`	No fine-tuning	0.530	0.901	**0.540**	**0.675**
	Simu-1 (Pseudo-label)	**0.531**	0.773	0.387	0.516
	Simu-1 (Ground-truth)	0.387	**0.986**	0.113	0.203