POCD: Probabilistic Object-Level Change Detection and Volumetric Mapping in Semi-Static Scenes

Jingxing Qian[*,1], Veronica Chatrath[*,1], Jun Yang[1], James Servos[2], Angela Schoellig[1], Steven L. Waslander[1]

***Abstract*—Maintaining an up-to-date map to reflect recent changes in the scene is very important, particularly in situations involving repeated traversals by a robot operating in an environment over an extended period. Undetected changes may cause a deterioration in map quality, leading to poor localization, inefficient operations, and lost robots. Volumetric methods, such as truncated signed distance functions (TSDFs), have quickly gained traction due to their real-time production of a dense and detailed map, though map updating in scenes that change over time remains a challenge. We propose a framework that introduces a novel probabilistic object state representation to track object pose changes in semi-static scenes. The representation jointly models a stationarity score and a TSDF change measure for each object. A Bayesian update rule that incorporates both geometric and semantic information is derived to achieve consistent online map maintenance. To extensively evaluate our approach alongside the state-of-the-art, we release a novel real-world dataset in a warehouse environment. We also evaluate on the public ToyCar dataset. Our method outperforms state-of-the-art methods on the reconstruction quality of semi-static environments.**

I. Introduction

In recent years autonomous robots have been deployed in challenging environments for extended operations, such as warehouse robotics, self-driving, and indoor monitoring. This highlights the importance of reliable map maintenance, as an up-to-date map allows robots to navigate efficiently and interact with objects in the scene. During long-term task completion, robots may encounter dynamic objects (people, robots), moderately dynamic objects (boxes, pallets), and objects that are usually static but can be modified occasionally (fences, walls). To map these environments, many widely adopted methods make use of dense, volumetric representations such as truncated signed distance functions (TSDFs) [1, 2, 3]. However, such methods assume an independent and identically distributed (i.i.d.) occupancy in each voxel, making them prone to corruption when the scene changes.

Recent works that attempt to handle scene dynamics extend monolithic map representations by using neural networks to segment the scene and identify potentially dynamic objects. However, most of these approaches focus primarily on either static [4, 5, 6] or short-term dynamic situations [1, 7, 8, 9, 10].

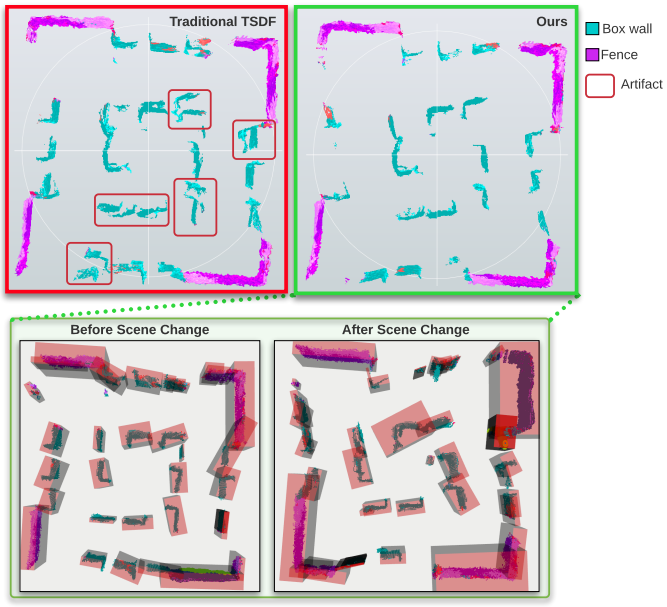

Fig. 1: A comparison of semi-static scene reconstruction between our method, POCD, and that of a traditional TSDF method [5] on the TorWIC_1-5 route. Using the traditional TSDF update method, artifacts, as boxed in red, persist after the box walls have moved between traversals. By comparison, our method produces a clean map that reflects the up-to-date scenario despite changed object poses. The bottom row shows our reconstructed map, along with object bounding boxes, before and after the scene has changed. The reference desired reconstruction is shown in Figure 6.

Promising results on table-top dynamic scenes have been demonstrated, but large, real-world environments wherein objects appear, disappear, or shift between robot traversals are overlooked. Such unobserved changes to objects pose a unique challenge to existing mapping systems, as commonly used change detection methods rely on motion tracking to identify dynamic objects, instead of assessing the consistency of a prior object pose with current sensor measurements. As a result, changes in the scene that occur when the robot is offline may not be detected easily, causing it to lose localization as the map accuracy degrades over time.

In this work, we aim to address the long-term map maintenance challenge in semi-static environments. We define as semi-static those environments that are primarily static but exhibit changed object locations between robot traversals. The primary motivation is that many useful operating environments for robots change slowly and at an object level, and these objects tend to either shift slightly, move significantly, appear or disappear. To address the impact of these changes on the

[*]Equal contribution
[1]The authors are with the University of Toronto Institute for Aerospace Studies and the University of Toronto Robotics Institute.
Emails: {firstname.lastname}@robotics.utias.utoronto.ca
[2]The author is with Clearpath Robotics, Waterloo, Canada.
Email: jservos@clearpath.ai
This work was supported by the Vector Institute for Artificial Intelligence in Toronto and the NSERC Canadian Robotics Network (NCRN).
Dataset details and download: https://github.com/Viky397/TorWICDataset

accuracy of the map, we introduce a novel framework, POCD, to track and correct object-level changes in the environment. The core of POCD is an object-level, probabilistic state representation that jointly models the stationarity (whether or not the object pose has changed) and the magnitude of geometric change for each object in the scene. A Bayesian update rule is derived to propagate each object's state representation by leveraging both geometric and semantic measurements. Additionally, we notice current change detection datasets focus on either table-top or household scenarios [11]. There exists a lack of datasets for long-term mapping in large industrial semi-static environments, such as warehouses and retail stores. With the help of Clearpath Robotics, we present the Toronto Warehouse Incremental Change Dataset (TorWIC), a representative change detection dataset in a warehouse setting.

Our proposed method is evaluated on both the aforementioned dataset and the public ToyCar dataset [12], and is compared against state-of-the-art approaches [1, 5, 7, 13]. We show that our framework produces more consistent maps despite the scene changing significantly, and is robust to sensor noise, partial observations and dynamic objects. We summarize our work with the following contributions:

- We introduce a Bayesian object-level update rule with a stationarity measure and a geometric change magnitude assessment that are modelled via a joint probability distribution to detect object pose changes. The update rule leverages both geometric and semantic measurements for robust change detection in incrementally evolving scenes.
- We introduce a novel method to measure object-level geometric change using the sum of voxel differences between TSDFs.
- We design POCD, a novel, online, object-aware mapping framework that simultaneously tracks and reconstructs changes in indoor semi-static environments which employs our Bayesian object-level update rule and geometric change calculation to propagate detected changes across entire objects, consistently.
- We release a real-world change detection dataset captured in a warehouse setting. The environment contains static objects that change between runs, as seen by both a RGB-D camera and 2D LiDAR, with robot poses provided.

To the best of our knowledge, POCD is the first to achieve online, probabilistic, object-level change detection and map updating for large, indoor, semi-static environments.

II. Related Works

A. Map Representation

The main map representations to model environments are:

- sparse, feature-based methods used in vision-based systems, such as ORB [4, 14] and SURF features [15],
- surface representations like meshes and surfels [16],
- point cloud representations [17], and
- dense, volumetric methods such as occupancy grids [18] and TSDFs[19].

In particular, TSDFs have gained traction in recent years due to their unique advantage of containing rich geometric information, resulting in higher robustness to sensor noise and smoother scene reconstruction [19]. Moreover, dense methods allow for measurement integration to be parallelized, resulting in efficient real-time updates.

However, most volumetric representations only capture the full geometry of the scene, ignoring object-level information. This renders them inefficient when large structures change pose, as all affected voxels must be updated through new observations to maintain an accurate map.

B. Semantic Information and Object-level Reconstruction

In recent years, researchers have expanded on volumetric map representations, incorporating both semantic and object-level information in map updates. Voxblox++ [26] and Kimera [5] extend [2] by incorporating voxel-level semantic information into the dense reconstruction. Recent works take this a step further by introducing objects into the map representation. Some methods use 3D CAD models to identify a set of pre-defined objects in the scene [27], while others construct unseen objects on-the-fly [28, 29]. However, these methods still adopt the static world assumption, which is unrealistic for a robot deployed in the real world for extended operations.

C. Handling of Dynamic Objects

Recent works have attempted to handle dynamic changes in the environment, adopting one of two common strategies. The first is to specifically identify static structure classes and treat all potentially dynamic objects, usually extracted with an image-based semantic segmentation network such as Mask R-CNN [30], as outliers, ignoring them completely in localization and mapping[31, 32, 33, 34]. Though this method has proven effective when a small number of fast-moving objects are present, it can fail when used in large, crowded environments, as only a small number of static background structures will remain after dynamic object pruning [35]. Moreover, many static objects from non-static object classes may be mission-critical (e.g., the robot may need to navigate to, and survey, pallets), and ignoring such objects may lead to inefficient task completion.

The second approach is to track the dynamic objects using multi-object tracking (MOT) methods [7, 10, 36, 37]. MaskFusion [7] decides whether an object is static based on motion inconsistency, and tracks dynamic objects by minimizing both geometric and photometric projective error. TSDF++ [38] actively updates the poses of detected objects by registering their 3D models in the global frame using iterative closest point (ICP) between consecutive frames. MID-Fusion [10] also uses motion inconsistency to detect dynamic objects and estimates their poses via joint dense ICP and RGB tracking. However, such methods only handle dynamics that can be detected immediately in consecutive frames, rendering them ineffective under semi-static infrastructure changes, which occur over a longer time horizon. As well, when the robot is surrounded by moving objects, this method may aggressively update

TABLE I: A comparison of indoor, changing scene datasets.

Dataset	Real-world	Environment	Full Depth	Camera Poses	Additional Sensors	Semantic Masks	Incremental Changes	Dynamic Objects	Size
InteriorNet [20]	✗	Household	✓	✓	IMU	✓	✓	✗	Million frames
Langer et al. [21]	✓	Household	✓	✓	✗	✓	✓	✗	31 trajectories, 5 scenes
Ambrus et al. [22]	✓	Office	✓	✓	✗	✓	✓	✗	88 trajectories, 8 scenes
Co-Fusion [12]	2: ✗ 3: ✓	Table-top/ Household	✓	✓	✗	✓	✗	✓	5 trajectories, 5 scenes
3RScan [11]	✓	Household	✓	✓	✗	✓	✓	✗	1482 trajectories, 478 scenes
TUM RGB-D Dynamic Objects[23]	✓	Office	✓	✓	IMU	✗	✗	✓	9 trajectories, 2 scenes
OpenLORIS [24]	✓	Household/ office/mall	✓	✓	Odom/IMU/ Fisheye/LiDAR	✗	✓	✓	22 trajectories, 5 scenes
Fehr *et al.* [1]	✓	Household	✓	✓	✗	✗	✓	✓	23 trajectories, 3 scenes
ChangeSim [25]	✗	Warehouse	✗	✓	✗	✓	✓	✗	80 trajectories $\sim$ 130k frames
POCD (ours)	✓	Warehouse	✓	✓	Odom/IMU/ 2D LiDAR	✓[1]	✓	✗	18 trajectories, 18 scenes, $\sim$ 70k frames

the object poses due to measurement noise and ambiguous association, leading to incorrect updates.

D. Detecting Incremental Changes

Detecting incremental changes in the scene is crucial to achieve long-term robot operation. Existing methods can be broadly classified into image-based or geometry-based approaches. Earlier image-based methods such as the works of Rosin [39] and Radke *et al.* [40] compare the intensities of aligned images to detect pixel-level changes. They are usually sensitive to view-angle variation and illumination changes. Recent research applies deep learning to achieve more robust change detection. In the works of Zhan *et al.* [41], Guo *et al.* [42], and Varghese *et al.* [43], the authors use supervised learning to train models to detect pixel or patch-level differences. However, such learned image-based methods require densely annotated training samples and can still be sensitive to view angle and appearance changes.

Instead, researchers leverage 3D geometric information to tackle these challenges. Alcantarilla *et al.* [44] warp coarsely registered camera images around dense reconstructions of the scene to mitigate view-angle differences. Fehr *et al.* [1] update a TSDF map by directly calculating voxel-level differences between signed distance functions of the previous reconstruction and the new measurement, pruning voxels above an error threshold. However, directly comparing geometric information at a voxel-level is prone to localization and measurement noise. Yew *et al.* [45] first use deep learning to perform non-rigid point cloud registration to mitigate localization drift, and then compare the registered point clouds to detect voxel-level changes. These methods still lack object-level information which can lead to semantically inconsistent map updates.

Recently, there has been work to detect and handle incremental changes at an object level. Gomez *et al.* [46] propose an offline method with an object pose-graph for mapping indoor semi-static environments. Data association and change estimation are performed offline between runs to update object persistence and poses. Gomez *et al.* model objects as cuboid bounding volumes instead of full 3D reconstructions, and only estimate class-level stationarity priors using a heuristic update rule, contrary to our estimate of object-level stationarity via Bayesian inference. Furthermore, they employ a centroid distance-based metric to detect object changes, which is difficult to obtain when only partial observations and incomplete object models are available.

Schmid *et al.* [13] recently proposed a framework that also aims to construct and maintain a dense reconstruction in semi-static environments at an object-level. In [13], each object is represented as a submap in a layered TSDF, with a stationarity score propagated by a heuristic update rule. Though we both aim to solve the same problem, our method features an object-level association module. This allows us to find explicit associations between observations and tracked objects, and estimate geometrically meaningful changes in the scene. On the other hand, [13] simply counts the number of inconsistent voxels, based on the weighted TSDF value differences, over overlapping submaps to determine if an update is required. Moreover, in our framework, object states are represented by a probability distribution and an update is performed via a Bayesian approach, allowing for more robustness against measurement and localization noise.

E. Indoor Changing Scene Datasets

To facilitate research in long-term autonomy in indoor changing scenes, a variety of datasets have been released. The datasets, both real and simulated, are captured in different environments such as indoor offices [22, 23], homes [1, 11, 12, 22, 20, 21], malls [24], and warehouses [25]. A comparison of these datasets is seen in Table I. Many of the datasets are collected using handheld devices rather than on a robot platform. The lack of additional sensors such as inertial measurement units (IMUs), odometry, and LiDAR, makes

[1]The included semantic masks are generated by a trained model. We also provide a human-annotated training set for fine-tuning. Please see the Supplementary Material for more details.

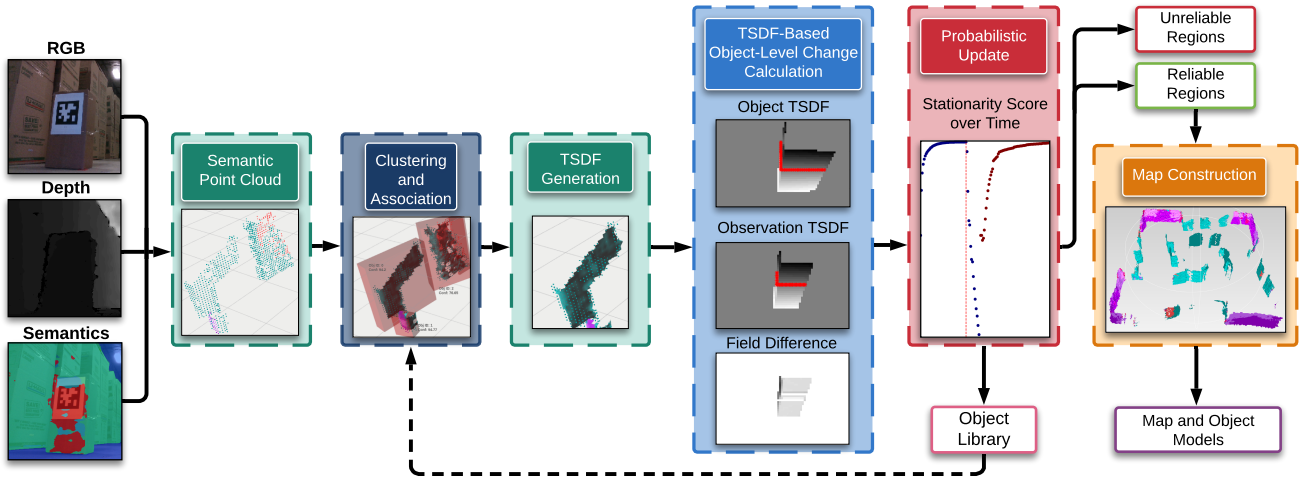

Fig. 2: Our object-aware map update framework (Section III) for semi-static environments. The system takes in RGB-D frames (Section III-B), each one semantically annotated and converted to a semantic point cloud (Section III-C). The point cloud is clustered into observations (Sections III-D), and associated to mapped objects (Section III-E). A TSDF-based object-level change estimation is performed between the associated observation-object pairs, followed by a joint probabilistic update of the geometric change and stationarity score (Section III-F). Objects with a high stationarity score are used to generate the new map (Section III-G), and the low-score objects are discarded.

these datasets difficult to extend beyond mapping tasks (e.g., simultaneous localization and mapping (SLAM)). Moreover, a real-world dataset in a warehouse environment is lacking.

III. System Description

A. Overview and Assumptions

In this work, we consider map maintenance in long-term, semi-static scenarios. Our goal is to track object-level change in the scene, reconstructing the objects when sufficient evidence of their change has been collected, to produce a final map that reflects the up-to-date configuration. In this section, we provide an overview of each part of the POCD framework. The following assumptions were made in this work:

1) The operating space is a bounded, indoor environment, such as a warehouse or a retail store with rigid objects.
2) High-level prior knowledge of the environment is available, such as object type and range of object dimensions.
3) The poses of the robot can be obtained from an existing localization system such as in [29].
4) Scene changes are due to the addition, removal, or planar motion of objects between robot traversals.

Though the focus of this work is on semi-static environments, we also identify objects that are dynamic, to improve the overall robustness of the system. A flow diagram of the proposed framework is shown in Figure 2.

B. Pipeline and Representations

We now introduce our object-aware pipeline which aims to improve long-term map quality in semi-static indoor environments. The system inputs are a sequence of colour and depth frames $\mathcal{F} = \{\mathbf{F}_t\}_{t=1\ldots T}$ taken from a RGB-D camera $\mathcal{C}$, and 6-DoF world-to-camera transformations, $\mathcal{T}^{CW} = \{\mathbf{T}_t^{CW} = \{\mathbf{p}_t^{CW}, \mathbf{q}_t^{CW}\}\}_{t=1\ldots T}$, with 3D position $\mathbf{p}_t^{CW}$ and orientation $\mathbf{q}_t^{CW}$ at each timestep t, provided by an independent localization system. The framework maintains a library of mapped objects $\mathcal{O} = \{\mathbf{O}_i\}_{i=1\ldots I}$ where each object $\mathbf{O}_i$ contains:

- a 4-DoF global pose, $\mathbf{T}_i^{OW} = \{\mathbf{p}_i^{OW}, \phi_i^{OW}\}$,
- a point cloud from accumulated depth data, $\mathbf{P}_i$, and the resulting TSDF reconstruction, $\mathbf{M}_i$,
- a bounding box, $\mathbf{B}_i$, that is aligned with the major and minor axes of the object reconstruction,
- a semantic class, c_i,
- a state probability distribution, $p(l_i, v_i)$, to model the object-level geometric change, $l_i \in \mathbb{R}$, and the stationarity (likelihood of the object being stationary), $v_i \in [0, 1]$.

The pipeline starts with an empty library. Since our experiments are conducted in a warehouse scenario, we assume the objects are restricted to planar motions. In particular, the objects can only rotate around the z-axis with a heading ϕ^{OW}. The system outputs a dense, up-to-date semantic TSDF map containing the current configuration of the environment. We choose a dense reconstruction as it contains rich geometric information that can be used for many robotic tasks such as path planning, obstacle avoidance, and robot-object interaction.

C. Semantic Segmentation from RGB Images

To extract semantically consistent instances from a RGB-D frame, $\mathbf{F}_t$, we utilize a semantic segmentation network

Fig. 3: A sample semantically labelled training image taken in a Clearpath Robotics warehouse, with a subset of the training labels.

(DeepLabV3 [47]) which provides pixel-level semantic predictions for 16 classes. The network was trained on a proprietary dataset of the warehouse environment provided by Clearpath Robotics, as seen in Figure 3.

Previous works require either instance or panoptic segmentation on their input images [7, 9, 10] in order to identify unique object instances. However, obtaining instance-level segmentation is difficult in real-world warehouse and retail settings where the robot will face visually identical and cluttered objects, such as boxes and pallets, that are contiguous with one another. Therefore, we adopt a semantic segmentation method which is representative of such challenging scenes.

D. 3D Observations from RGB-D Frames

We extract a set of observations $\mathcal{Y}_t = \{\mathbf{Y}_{t,j}\}_{j=1\dots J}$ from the segmented frame $\mathbf{F}_t$, where each observation, $\mathbf{Y}_{t,j}$, contains:

- a 4-DoF global pose, $\mathbf{T}_{t,j}^{YW} = \{\mathbf{p}_{t,j}^{YW}, \phi_{t,j}^{YW}\}$,
- a point cloud, $\hat{\mathbf{P}}_{t,j}$, extracted from $\mathbf{F}_t$,
- a bounding box, $\mathbf{B}_{t,j}$, that is aligned with the major and minor axes of the object reconstruction,
- a semantic class, $c_{t,j}$,
- and a stationarity class, $s_{t,j}$.

To obtain $\mathcal{Y}_t$, we convert $\mathbf{F}_t$ into a semantically segmented point cloud, $\mathbf{P}_t^{\text{raw}}$, transforming it to align with the world frame based on the current camera pose, $\mathbf{T}_t^{CW}$. We first remove the ground-plane and reject outliers from $\mathbf{P}_t^{\text{raw}}$. Since our target environments are warehouses and retail spaces, many planar features exist in the scene. Thus, we search for planes in $\mathbf{P}_t^{\text{raw}}$, and enforce semantic consistency over the co-planar points via a majority voting scheme to reject noise from incorrect semantic segmentation. This yields a filtered point cloud, $\mathbf{P}_t^{\text{filt}}$. Note that other shape priors can be integrated based on prior knowledge of the target environments.

Using Euclidean clustering, we extract semantically consistent clusters, $\{\hat{\mathbf{P}}_{t,j}\}_{j=1\dots J}$, from $\mathbf{P}_t^{\text{filt}}$. An observation, $\mathbf{Y}_{t,j}$, is spawned for each cluster, $\hat{\mathbf{P}}_{t,j}$. To retrieve the 4-DoF pose, $\mathbf{T}_{t,j}^{YW}$, of $\mathbf{Y}_{t,j}$, we follow a two-step approach. z-axis aligned 3D bounding cuboid, $\mathbf{B}_{t,j}$, is fit to $\hat{\mathbf{P}}_{t,j}$, with its x and y axes aligned with the major and minor axes of the flattened $\hat{\mathbf{P}}_{t,j}$ using principle component analysis (PCA), to acquire its heading, $\phi_{t,j}^{YW}$. The translation, $\mathbf{p}_{t,j}^{YW}$, is set to the centroid of $\mathbf{B}_{t,j}$. Observations with overlapping bounding cuboids are then merged, and semantic consistency is once again enforced.

A semantic class, $c_{t,j}$, is assigned to each observation, $\mathbf{Y}_{t,j}$, based on the enforced semantic prediction over the points. Finally, a stationarity class, $s_{t,j}$, can be mapped from $c_{t,j}$, where $s_{t,j} = s(c_{t,j}) \in \{0, 1\}$, by leveraging prior class property knowledge. The value of $s_{t,j} = 0$ denotes a dynamic object and the value of $s_{t,j} = 1$ denotes a static object. For example, an object with $c_{t,j} = robot$ will have $s_{t,j} = 0$ whereas an object with $c_{t,j} = shelf$ will have $s_{t,j} = 1$.

E. Object-Level Data Association

The extracted observations, $\mathcal{Y}_t$, are compared to existing objects in the mapped object library, $\mathcal{O}$. Point-to-plane ICP, based on the Point Cloud Library (PCL) [48], is first run between each object-observation pair, $\{\mathbf{O}_i, \mathbf{Y}_{t,j}\}$, with a centroid distance below a generous association distance threshold, θ_{dist}, to find both the relative transformation, $\mathbf{T}_{ij} = \{\mathbf{p}_{ij}, \phi_{ij}\}$, and geometric dissimilarity, ϵ_{ij} (percent of outliers after ICP convergence), for each pair. A cost matrix, $\mathbf{C}$, is then constructed between all feasible pairs based on a weighted sum of the relative pose change and semantic consistency:

$$\mathbf{C}(i,j) = \lambda_1 \left\|\mathbf{p}_{ij}\right\|_2 + \lambda_2 |\phi_{ij}| + \lambda_3 (1 - \mathbb{I}[c_i = c_j]) \quad (1)$$

Object-observation pairs with a geometric dissimilarity, ϵ_{ij}, greater than the similarity threshold, θ_{sim}, not within the association distance threshold, θ_{dist}, or with a cost, $\mathbf{C}(i,j)$, above a cut-off threshold, θ_{cutoff}, are given a cost of infinity. The resulting cost matrix is run through the greedy Hungarian algorithm to find an optimal association. Unassociated observations are added to the mapped object library with an initial state probability distribution (with geometric change expectation, $\mathbb{E}[l] = 0$, and stationarity expectation, $\mathbb{E}[v] = v_{\text{class}}$), and are marked as *new* objects. For observations that have been assigned to an existing object in the library, the new observation is stored and the object is marked as *update-pending*. Objects with at least θ_{vis} percent of their points within the current camera frustum, but with no associated observations, are marked as *unobserved*. Note that, since the Hungarian algorithm has a polynomial complexity of $O(n^3)$, we only perform data association within the distance threshold, θ_{dist}, to reduce the computational cost under large scenes. For objects displaced by a large distance, we believe it is not necessary to find the actual correspondence to update the map.

F. Probabilistic Stationarity and Change Update

As *update-pending* and *unobserved* objects are identified, their state distributions are updated via Bayesian inference, as discussed in Section IV-D. For an *update-pending* object, $\mathbf{O}_i$, we then estimate the magnitude of geometric change, $\Delta_i \in \mathbb{R}$, as discussed in Section IV-A, to construct the likelihood distribution for the Bayesian update. If Δ_i is within half of the measurement standard deviation, σ, which we deem a successful geometric verification, the observation is integrated into the object's TSDF, $\mathbf{M}_i$. Else, the observation is discarded, as it is no longer consistent with the object model from previous observations. For an *unobserved* object, a large pseudo-change is used to penalize its stationarity. Similar to the voxel-level confidence clamping trick used in [18], we do not perform the object-level state update if the new observation brings an object's stationarity score above an upper bound, v_{max}, to ensure responsiveness to changes.

G. Object and Map Update

Once all *update-pending* and *unobserved* objects are updated, the stationarity score expectation, $\mathbb{E}[v_i]$, of each object is checked against a heuristic-based stationarity threshold, θ_{stat}. If the expected score falls below the threshold, all voxels in map that are associated with the object's TSDF, $\mathbf{M}_i$, are reinitialized and the object is erased from the object library, $\mathcal{O}$. All *new* objects are integrated into the library and the map.

IV. Methodology

In this section, we present the details of our contributions, including how we estimate changes between object-observation pairs, justifications behind our probabilistic modelling, and the derivation of our Bayesian update rule.

A. Object-Level Geometric Change Estimation

We first describe our approach to estimate the object-level geometric change using TSDFs. The method is inspired by [49] and [50], where the authors propose to improve 3D reconstruction quality by minimizing the cumulative back-projection depth error between RGB-D images and the TSDF model. Here, we ray-trace from the current camera pose to obtain a scalar-valued, zero-mean error measure between two TSDFs. For each object-observation pair, $\{\mathbf{O}_i, \mathbf{Y}_{t,j}\}$, we first transform the point cloud of the object, $\mathbf{O}_i$, namely $\mathbf{P}_i$, and the point cloud of the observation, $\mathbf{Y}_{t,j}$, namely $\hat{\mathbf{P}}_{t,j}$, to the camera optical frame, $\mathcal{C}$. Two TSDFs are constructed around the two point clouds by ray-tracing through the camera optical center, $\mathbf{o}$. We define the change of $\mathbf{O}_i$ with respect to $\mathbf{Y}_{t,j}$ over voxels, $\mathbf{\Omega}$, as:

$$\Delta_{t,ij} = \text{sign}(\mathbf{p}_z^{\mathcal{C}}) \frac{\lambda_{\text{diff}}}{|\mathbf{\Omega}|} \sum_{\mathbf{v} \in \mathbf{\Omega}} |\text{tsdf}(\hat{\mathbf{P}}_{t,j}, \mathbf{v}) - \text{tsdf}(\mathbf{P}_i, \mathbf{v})| \quad (2)$$

where λ_{diff} is a scaling factor and $\mathbf{\Omega}$ is the intersection of the non-zero voxel sets of the two TSDF grids:

$$\mathbf{\Omega} = \{\mathbf{u} \mid \text{tsdf}(\hat{\mathbf{P}}_{t,j}, \mathbf{u}) > 0 \wedge \text{tsdf}(\mathbf{P}_i, \mathbf{u}) > 0\}.$$

Here, $\mathbf{p}_z^{\mathcal{C}}$ is the z component of the 3D translation found by the ICP between $\mathbf{P}_i$ and $\hat{\mathbf{P}}_{t,j}$ in the camera optical frame, $\mathcal{C}$, where the z-axis points forward. Thus, $\mathbf{p}_z^{\mathcal{C}}$ is positive if $\hat{\mathbf{P}}_{t,j}$ is in front of $\mathbf{P}_i$, and negative otherwise, from the camera's point of view. Therefore, $\Delta_{t,ij}$ takes a positive or negative value depending on the actual geometric change. If the object stays at the same pose, $\Delta_{t,ij}$ is approximately 0. For *unobserved* objects, we define a large pseudo-change of $\Delta_{t,ij} = \Delta_{\text{max}}$ to force the removal of *unobserved* objects.

In comparison to other metrics discussed in Section II, our approach directly estimates how much the object has changed in the metric space. It utilizes full geometric information of the objects and also takes camera view angle, potential occlusions, and effects of partial observations into account.

B. Object-Level State and Process Model

To support object-level reasoning of the environment dynamics, we introduce a novel probabilistic object state representation. For each object, $\mathbf{O}_i \in \mathcal{O}$, and its previously associated observations, $\{\mathbf{Y}_{t,j}\}_{t=1\ldots T}$, we can extract a sequence of high-level measurement features, $\{\mathbf{z}_{t,j}\}_{t=1\ldots T}$ (see Section IV-C). We would like to jointly estimate the object's geometric change, $l_i \in \mathbb{R}$, and stationarity score, $v_i \in [0, 1]$. Dropping the object index, i, and observation index, j, for clarity, this object-level joint state probability distribution is given by:

$$p\left(l, v \mid \mathbf{z}_1 \ldots \mathbf{z}_T\right) \quad (3)$$

This joint distribution can be updated iteratively as new measurements arrive, by following a Bayesian update rule:

$$\overbrace{p\left(l, v \mid \mathbf{z}_1 \ldots \mathbf{z}_T\right)}^{\text{Posterior}} \propto \overbrace{p\left(\mathbf{z}_T \mid l, v, \mathbf{z}_1 \ldots \mathbf{z}_{T-1}\right)}^{\text{Measurement Likelihood}} \overbrace{p\left(l, v \mid \mathbf{z}_1 \ldots \mathbf{z}_{T-1}\right)}^{\text{Prior}} \quad (4)$$

Assuming that geometric change, l, and stationarity score, v, are conditionally independent, we adopt a probability model, first proposed in [51] and later verified in [52, 53, 54, 55] for volumetric fusion and sparse visual SLAM, to parametrize the state distribution in Equation 3 as the product of a Gaussian distribution for l, and a Beta distribution for v:

$$\begin{aligned} p\left(l, v \mid \mathbf{z}_1 \ldots \mathbf{z}_T\right) &:= q\left(l, v \mid \mu_T, \sigma_T, \beta_T, \alpha_T\right) \\ &:= \mathcal{N}(l \mid \mu_T, \sigma_T^2)\text{Beta}(v \mid \alpha_T, \beta_T) \end{aligned} \quad (5)$$

In all previous works, the Gaussian-Beta parametrization has been used in pixel-level depth estimation and inlier/outlier identification. The same parametrization also fits nicely into our object-level change detection framework, as l is expected to center around an underlying change measure (0 for stationary objects), and v estimates the likelihood of the object being stationary. In Equation 5, μ_T and σ_T^2 represent the mean and variance of l respectively, and α_T and β_T are the number of observed inlier and outlier measurements with respect to the model (see Section IV-C). Finally, object pruning decisions can be made based on the expectation of the stationarity, $\mathbb{E}[v]$.

C. Measurement Likelihood Model

In our setup, each measurement feature, $\mathbf{z}_t$, contains the geometric change measure, Δ_t, as calculated in Section IV-A, and the stationarity class, s_t, as determined in Section III-D:

$$\mathbf{z}_t = \{\Delta_t, s_t\} \quad \forall t = 1 \ldots T. \quad (6)$$

Applying Bayes rule to the *Measurement Likelihood* term in Equation 4, the measurement likelihood becomes:

$$\begin{aligned} &p\left(\mathbf{z}_T \mid l, v, \mathbf{z}_1 \ldots \mathbf{z}_{T-1}\right) \\ &= p\left(\Delta_T, s_T | l, v, \Delta_1, s_1 \ldots \Delta_{T-1}, s_{T-1}\right) \\ &\propto p\left(\Delta_T \mid l, v, \Delta_1, s_1 \ldots \Delta_{T-1}, s_{T-1}, s_T\right) \\ &\quad \times p\left(s_T \mid l, v, \Delta_1, s_1 \ldots \Delta_{T-1}, s_{T-1}\right) \end{aligned} \quad (7)$$

Further, we assume that the geometric terms, Δ_t and l, and the visual-semantic term, s_t, are independent, and the measurements, $\mathbf{z}_t$, are conditionally independent given current l and v estimates. Equation 7 can then be simplified as the product between a geometric consistency likelihood, $p\left(\Delta_T \mid l, v\right)$, and a stationarity likelihood, $p\left(s_T \mid v\right)$:

$$\begin{aligned} &p\left(\Delta_T, s_T \mid l, v, \Delta_1, s_1 \ldots \Delta_{T-1}, s_{T-1}\right) \\ &\propto p\left(\Delta_T \mid l, v\right) p\left(s_T \mid v\right) \end{aligned} \quad (8)$$

We categorize the measurements in one of two ways: as an inlier measurement where the object did not move and the measured geometric change is normally distributed around the current estimate l (initially 0), or as an outlier measurement where the object is moved and the change is uniformly distributed in an interval, $[-\Delta_{\text{max}}, \Delta_{\text{max}}]$. Inspired by [51], where the authors use a Gaussian-Uniform mixture to model uncer-

tainties in pixel-level depth measurements, we also model the object-level geometric consistency likelihood as a Gaussian-Uniform mixture weighted by the stationarity score, v:

$$p\left(\Delta_T \mid l, v\right) := v\mathcal{N}(\Delta_T \mid l, \tau^2) + (1-v)\mathcal{U}(\Delta_T \mid -\Delta_{\max}, \Delta_{\max}) \tag{9}$$

where the measurement variance, τ^2, can be determined experimentally using both static and changed reference objects.

On the other hand, the stationarity class, $s_T = s(c_T)$, can be intuitively considered as a sample from a Bernoulli process controlled by the object's stationarity score, v:

$$s_T \sim \text{Bernoulli}(v)$$

Note that the Beta stationarity prior of Equation 4 can be considered as a conjugate prior to the Bernoulli stationarity likelihood here. As a result, both the geometric consistency measurement, Δ_T, and the stationarity measurement, s_T, contribute to the update of the stationarity score, v. To balance the relative importance between Δ_T and s_T, we introduce an adaptive factor, k, depending on s_T and Δ_T:

$$p\left(s_T \mid v\right) := \text{Bernoulli}(s_T \mid v)^k \tag{10}$$

The factor, k, acts as a weight in the Beta stationarity update rule for the posterior, as will be shown in Section IV-D. It aids in adjusting the model behaviour. For example, the model should adapt quickly when a large geometric change is measured for dynamic objects, such as robots, while being more conservative to static objects, such as shelves.

D. Probabilistic Update Rule

Combining the measurement models, Equation 9 and Equation 10, and the parametrized state model, Equation 4, we can derive a Bayesian update rule. If the prior (i.e. the model after processing measurements $\{z_t\}_{t=1\ldots T-1}$) is parametrized by $(\mu, \sigma, \alpha, \beta)$, the true posterior would have the form:

$$p(l, v \mid \Delta_T, s_T, \mu, \sigma, \alpha, \beta) = \eta p(\Delta_T \mid l, v) p(s_T \mid v) q(l, v \mid \mu, \sigma, \alpha, \beta) \tag{11}$$

for some normalization factor, η. The true posterior can be rearranged to take the form:

$$\begin{aligned} p(l, v \mid \Delta_T, s_T, \mu, \sigma, \alpha, \beta) = \\ C_1\mathcal{N}(l \mid m, \gamma^2)\text{Beta}(v \mid \alpha + ks_T + 1, \beta + k(1-s_T)) \\ +C_2\mathcal{N}(l \mid \mu, \sigma^2)\text{Beta}(v \mid \alpha + ks_T, \beta + k(1-s_T) + 1) \end{aligned} \tag{12}$$

which is a weighted mixture of two Gaussian-Beta distributions. The weights, C_1 and C_2, are the probability of the measurement, $\mathbf{z}_T$, being an inlier and outlier, respectively. The first term models the effect of fusing an inlier measurement, where the geometric change, l, is updated with a new mean, m, and variance, γ^2. The stationarity score, v, also adapts with a k-weighted update rule. The second term models the effect of fusing an outlier measurement, where only the stationarity score, v, adapts.

Note that Equation 12 cannot be written as a single Gaussian-Beta distribution exactly. As in [51, 52, 55], we find a new parametrization for an approximated Gaussian-Beta posterior $q(l, v \mid \mu', \sigma', \alpha', \beta')$ by matching the first and second moments for l and v to the true posterior in Equation 12. The approximated posterior can be found analytically, enabling efficient online inference. We refer the reader to the Supplementary Material for the full derivation.

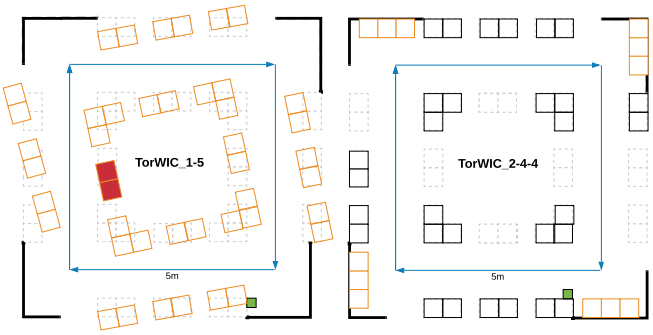

Fig. 4: The TorWIC_1-5 and TorWIC_2-4-4 ground truth schematics, respectively. Green box: fixed AprilTag for drift reference; black: stationary box or fence; orange: shifted boxes; dotted grey: original position of shifted boxes. The evolution of the red-filled box wall's stationarity score $\mathbb{E}[v]$ on the left is shown in Figure 7.

Fig. 5: An RGB image comparison of the robot revisiting the same spot (AprilTag) during the TorWIC 2_4_4 route. Changes include 3 stacks of boxes added in front of the fence, and an additional box wall to the right of the fence. The ground truth schematic of this changed route can be seen in Figure 4, right.

V. Experimental Results

A. Experimental Setup

We verify the performance of our framework both qualitatively and quantitatively by comparing the map reconstruction of POCD to several baseline methods, each representative of a category of dynamic environment mapping approaches:

- Kimera [5]: Assumes the world is static and adopts a naive TSDF update rule.
- MaskFusion [7]: Actively tracks and updates all potentially dynamic objects.
- Fehr *et al.* [1]: Performs voxel-level corrections by comparing new observations against the constructed map.
- Panoptic Multi-TSDFs [13]: Maintains objects as static submaps and performs consistency checks to prune those that have changed.

To demonstrate the capabilities of POCD, we evaluate on two datasets. Due to the lack of real-world semi-static datasets, we create one for the purpose of this problem (Section V-B). Furthermore, we use the ToyCar dataset from [12] to show that our framework not only handles semi-static changes, but is also robust to dynamics. As the work of Fehr *et al.* is not open-source, it was implemented on top of Voxblox [2] based on [1].

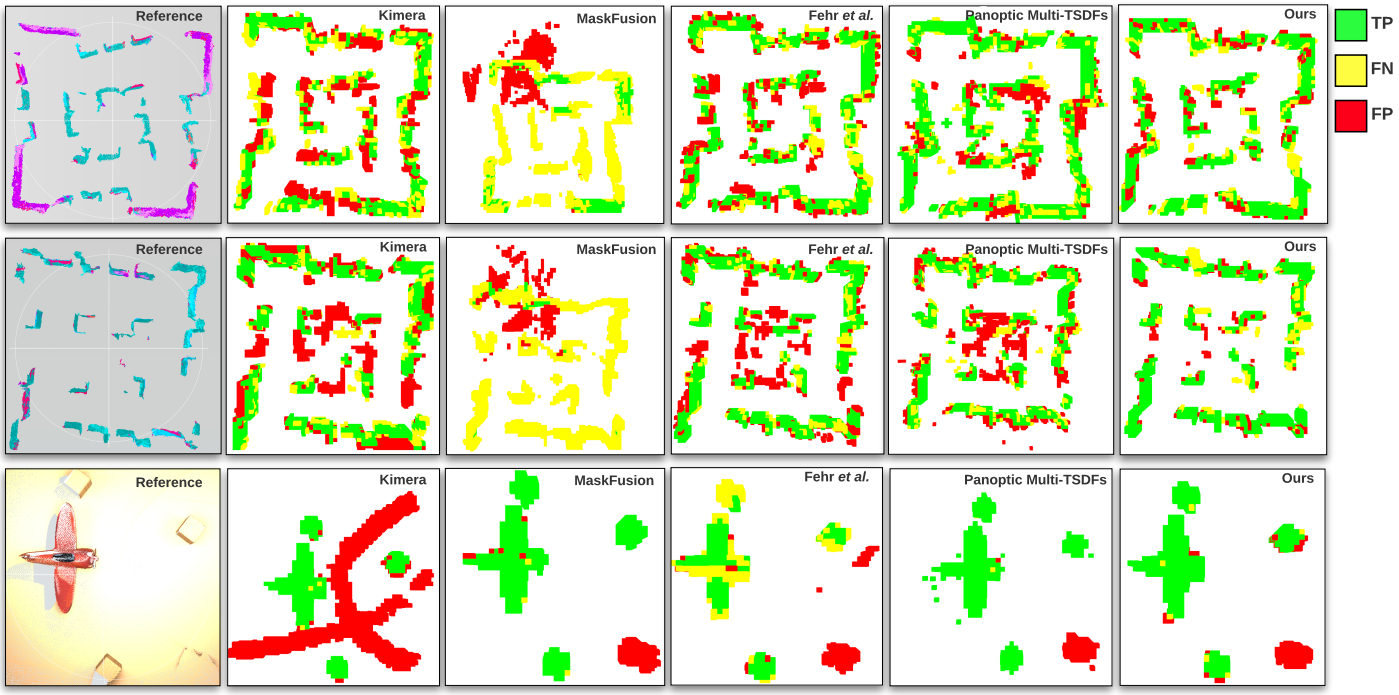

Fig. 6: Bird's-eye-view qualitative 3D reconstruction results of the **top row:** TorWIC_1-5 route, **middle row:** TorWIC_2-4-4 route, and **bottom row:** ToyCar scenario. The reconstruction produced by POCD is compared against that of Kimera, MaskFusion, Fehr *et al.*, and Panoptic Multi-TSDFs. The green, yellow, and red sections represent true positives, false negatives, and false positives, respectively. The first image is the reference map of the routes' final scenario. A voxel size of 20 cm for the TorWIC routes and 6 cm for the ToyCar scenario is used to compute the metrics.

We fine-tune the parameters of the baselines on our dataset whenever possible for the best reconstruction. We implement POCD on top of Kimera [5].

B. The TorWIC Dataset

We release TorWIC, a real-world warehouse dataset collected on a OTTO 100 Autonomous Mobile Robot equipped with a RealSense D435i RGB-D camera, a 2D Hokuyo UAM501 LiDAR, a wheel encoder, and an inertial measurement unit (IMU). The robot setup can be seen in Figure 3, the sensor specifications are listed in Table 3, and sample sensor data can be seen in Figure 4, of the Supplementary Material.

We collected 18 real-world trajectories with 4 types of changes, each with a unique warehouse-like scenario, containing the following objects:

- 2m-tall walls made out of cardboard boxes
- 3m-tall tarp-covered fences at the end of each hallway

The robot makes multiple passes of each of the 18 scenarios to accumulate sufficient sensor coverage. Reference poses are obtained by running a proprietary LiDAR-based SLAM algorithm for each trajectory. The scenarios differ by the removal, addition, shift, or rotation of boxes or fences. Since each trajectory starts and ends in the same spot, identifiable by the AprilTag, they can be stitched in different ways using the provided script to create longer routes that present changed object locations over time. Figure 4 shows the scenario changes for two sample routes, TorWIC_1-5 (trajectory 1_1 and 1_5 stitched) and TorWIC_2-4-4 (trajectory 2_2, 2_4, and 4_1 stitched), and Figure 5 shows two RGB images captured by the robot along the TorWIC_2-4-4 route. A high level breakdown of the trajectories can be found in Table 4 of the Supplementary Material. A document with detailed descriptions of each scenario is provided[2].

C. Real-World Experiment in a Semi-Static Environment

We first perform qualitative and quantitative evaluations against the four baseline methods on two routes from the TorWIC dataset, as shown in Figure 4. Since a ground truth map is not available, and each method reconstructs the scene differently, we create a set of reference maps by processing the final scenario through each method. For a qualitative evaluation, we visually inspect the reconstruction against the corresponding reference map. For a quantitative evaluation, we overlay and voxelize both the reference and reconstructed maps and count overlapping and inconsistent voxels to compute the precision, recall, and false positive rate (FPR) for each method at the voxel level. We list the parameters used in POCD in the Supplementary Material.

The top two rows of Figure 6 contain the bird's-eye-views of the reference and reconstructions of the two TorWIC routes. Quantitative evaluation results are listed in Table II. When exporting the final map, we downsample and threshold voxel weights to prune the low confidence regions such that all methods produce a similar recall. Here, precision showcases

[2]https://github.com/Viky397/TorWICDataset/blob/main/TorWIC_Dataset.pdf

TABLE II: Quantitative mapping results on the TorWIC dataset.

TorWIC_1-5	Precision ↑	Recall (TPR) ↑	FPR ↓
Kimera	60.6	74.2	7.7
MaskFusion	54.5	24.4	3.5
Fehr *et al.*	76.3	76.6	3.8
Panoptic Multi-TSDFs	77.4	**78.9**	4.2
POCD (ours)	**80.2**	78.7	**3.0**
POCD Improvement	2.8	-0.2	0.5
TorWIC_2-4-4	**Precision ↑**	**Recall (TPR) ↑**	**FPR ↓**
Kimera	55.8	78.3	8.5
MaskFusion	23.8	5.5	3.0
Fehr *et al.*	67.4	80.7	5.3
Panoptic Multi-TSDFs	62.7	77.1	5.3
POCD (ours)	**87.0**	**81.2**	**1.6**
POCD Improvement	19.6	0.5	1.4

the framework's coverage of true objects, and FPR showcases how well objects are updated after a scene change. Therefore, the precision and FPR should be compared for performance evaluation. It can be seen that POCD produces the most visually accurate map, with minimal ghost artifacts or double walling. It also produces the highest precision and lowest FPR.

When compared to the baselines, Kimera does not handle shifted objects, leaving many artifacts such as double walls in the reconstruction. MaskFusion relies on both geometric and semantic consistency to track objects. However, when faced with partial observations of visually similar boxes, it fails to track the box wall due to ambiguous associations and inaccurate object motion estimates, resulting in an incorrectly constructed map with a low precision and recall, despite being given the reference robot poses. The method of Fehr *et al.* is able to partially correct the map for regions inside the camera's field-of-view (FOV), but is still left with artifacts and ghosts outside the camera's FOV, due to the absence of an object-level understanding of the environment. Panoptic Multi-TSDFs is also able to partially correct the map. In contrast to the method of Fehr *et al.*, it maintains object-level information, thus is able to perform consistent map updates for partially visible objects. However, this method lacks an explicit object association module and detects changes by counting inconsistent voxels over overlapping submaps. As a consequence, it behaves similarly to the naive TSDF update when an object has moved too far from its initial position and no association is established.

As discussed in Section III-F, POCD propagates a belief distribution for the stationarity score, v, for each object in the scene as the robot accumulates more measurements. Figure 7 shows the evolution of the stationarity score's expectation for a sample box wall. As the robot revisits the box wall, its estimate of the object being stationary adapts. As the scene changes, its stationarity drops below the threshold, θ_{stat}, and is erased and recreated at its new location. The drop in stationarity seen around 400 seconds is due to inaccuracies in poses obtained during dataset collection. However, the measurement is still considered an inlier by the measurement model and thus integrated into the object's reconstruction, increasing the stationarity score after more observations are made.

Finally, we study POCD's effectiveness on map maintenance after a scene change. Figure 8 plots the precision, recall,

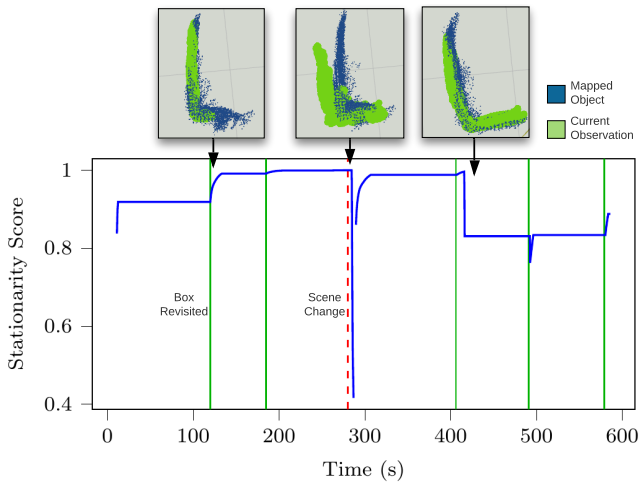

Fig. 7: Evolution of the expected stationarity, $\mathbb{E}[v]$, of the red-filled boxes in Figure 4, as the robot makes multiple loops. The stationarity score adapts each time the box is revisited along the robot's trajectory, indicated by the green lines. The expectation increases or decreases depending on how well the new observations match the mapped object model, with the mapped object and current observation visible at three separate times. The object is discarded and reconstructed when the scene changes, indicated by the red dotted line.

and FPR of the reconstructed map over time. We perform two runs with POCD after the scene change: one continued run starting with a prior map constructed before the scene change, and another reference run without any prior maps. The reference run reconstructs the post-change scene from scratch and should produce the highest quality map achievable for all shifted objects. As seen in Figure 8, POCD is able to steadily increase both precision and recall, and reduce FPR, as the robot moves through the environment. The precision starts off low for the continued run as the percentage of change in the environment is large. Therefore, the box walls reconstructed in the prior map no longer reflect the new scenario. However, a large portion of the voxels in the prior map, such as those belonging to the unchanged fences, are still valid, leading to

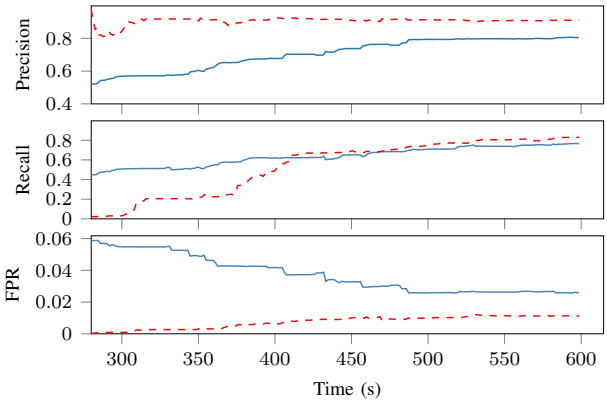

Fig. 8: Evolution of the precision, recall, and false positive rate (FPR) of POCD after the scene has changed, on the TorWIC_1-5 trajectory, starting from a map of the previous scenario (solid blue line), and from scratch (dashed red line).

TABLE III: Quantitative mapping results on the ToyCar dataset.

ToyCar	Precision ↑	Recall (TPR) ↑	FPR ↓
Kimera	24.9	97.4	13.4
MaskFusion	80.2	97.8	1.8
Fehr *et al.*	62.0	51.3	2.0
Panoptic Multi-TSDFs	**82.0**	**99.2**	**1.2**
POCD (ours)	79.4	96.9	1.9
POCD Improvement	-2.6	-2.3	0.7

a high recall at the start and throughout the run. The final map has a comparable recall to that of the reference, with a slightly lower precision and slightly higher FPR due to regions not seen during the second trajectory. This evaluation shows how our framework effectively corrects inconsistencies in the map when sufficient observations have been made.

D. Robustness Against Dynamics

We will now evaluate the methods' robustness to dynamic objects, specifically in comparison to the works of Fehr *et al.*, and Panoptic Multi-TSDFs, which also focus on mapping in semi-static scenes. For this experiment, we use the simulated ToyCar dataset from [12], as they provide a ground truth camera trajectory, depth, and instance-level segmentation images. Similar to the real-world experiment, we construct a set of reference background maps using each method, by masking out the two dynamic toy cars. We tune parameters for each method to produce the best visually reconstructed map, while maintaining a similar recall for a fair comparison.

The third row of Figure 6 contains the bird's-eye-view of both reference and reconstructed scenes of the ToyCar dataset, with the respective quantitative analyses listed in Table III. Kimera does not handle dynamics in the environment, leaving a trail behind the two toy cars as they drive. In contrast to the real-world experiment, MaskFusion is able to produce a highly accurate scene reconstruction as it is now provided with instance-level segmentation, thus enabling accurate motion estimation of the toy cars. The work of Fehr *et al.* is able to produce a clean reconstruction as well, though it does not operate at an object-level. Therefore, voxels that belong to complex surfaces are pruned aggressively, resulting in poor object completeness and a low precision and recall. Moreover, artifacts are left behind the toy cars when they are blocked from the camera. Panoptic Multi-TSDFs achieves the most accurate final reconstruction. POCD produces a clean final reconstruction comparable to that of MaskFusion and Panoptic Multi-TSDFs. Note that the toy car is masked as false positive at its final position (bottom right corner) in all reconstructions, as it is not present in the reference background map.

While the methods that attempt to handle changes (MaskFusion, Fehr *et al.*, Panoptic Multi-TSDFs, POCD) achieve comparable reconstructions of the final scene, their reconstruction quality for the dynamic toy cars varies based on the assumptions they adopt. In Figure 9, we compare the reconstruction quality of one of the toy cars during the run. Kimera assumes a static scene, thus failing to handle the toy car. MaskFusion is specifically designed to handle dynamic objects, producing the most visually accurate model and a smooth trajectory. The other three methods target semi-static environments, and do not explicitly handle dynamic objects:

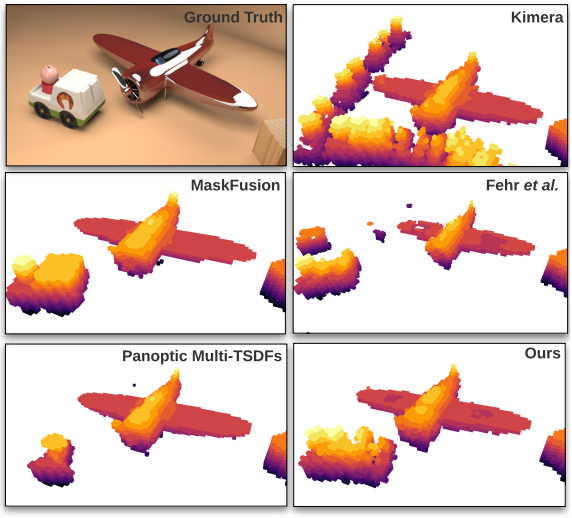

Fig. 9: Qualitative comparison of the toy car reconstruction during its motion. Note that Kimera does not handle any dynamics whereas MaskFusion is designed for dynamic scenes.

all these methods erase and reconstruct the toy car at its new position after sufficient observations are gathered. The work of Fehr *et al.* is able to correctly construct and update this toy car as it is never occluded. Though, as map updates are performed at a voxel-level, this approach performs ineffectively under occlusions, as can be seen by the resulting trail left behind the other toy car shown in the middle-right row.

Panoptic Multi-TSDFs assumes all objects are static and does not perform explicit object-level association. Hence, when the toy car moves out of its original bounding region, a correct association is not established and the object-level map update degrades to the traditional voxel-level TSDF update behaviour. As seen in the bottom-left row of Figure 9, the resulting reconstruction is eroded when the toy car leaves its submap. Alternatively, POCD features an object-level association module allowing it to achieve correct associations even in the presence of dynamics. In contrast, POCD can robustly update the toy car at an object-level, even when it has moved a large distance. Note that as we focus on semi-static changes in this work, POCD is not able to perform incremental object pose tracking for dynamic objects, as updates to object poses only occur when sufficient confidence is achieved that the object has moved. We plan to explore explicit handling of dynamics as an extension to this work.

E. Runtime Analysis

We evaluate the performance of our framework on a Linux system with an AMD Ryzen R9-5900X CPU with 24 threads at 3.7GHz. Our runtime analysis does not account for segmentation, which is performed offline, as it is not the focus of this work. The average runtime performance is measured on the TorWIC_1-5 route over all frames. With an input resolution of 640×360 and a voxel size of 5 cm, the average per-frame computation time for point clustering and object-observation association, object state update and model integration, and map maintenance is 118.30 ms, 49.80 ms and 50.72 ms, respec-

tively. Together, we achieve an average FPS of 4.57, compared to 22.50 FPS for the original Kimera Semantics codebase, which our system builds on, and 5-6 FPS for Panoptic Multi-TSDFs [13]. The most expensive operation is clustering and association, varying our frame rate from 1.25 FPS when 43 objects are mapped, to 34.24 FPS when only two objects are mapped. Some box walls are mapped as multiple objects due to a limited sensor FOV. POCD is amenable to online operation as map maintenance is not required at every frame. Note that our codebase is not optimized for computation time, and we expect to achieve significantly higher performance by utilizing parallel processing and GPUs.

VI. Conclusion

In this paper, we present POCD, a novel online, object-aware map maintenance framework that simultaneously tracks and reconstructs incremental changes in the environment by leveraging both geometric and semantic information. POCD performs Bayesian updates to propagate a joint state distribution for each tracked object in the scene. We experimentally verify the robustness of POCD against several state-of-the-art baselines on two datasets, including TorWIC, a real-world warehouse dataset that we release with this work. The results suggest that object-level knowledge of the environment, explicit object tracking, consistent object-level change estimation, and different levels of dynamics-handling are all crucial for reliable long-term map maintenance.

References

[1] Marius Fehr, Fadri Furrer, Ivan Dryanovski, Jürgen Sturm, Igor Gilitschenski, Roland Siegwart, and Cesar Cadena. TSDF-based change detection for consistent long-term dense reconstruction and dynamic object discovery. In *2017 IEEE International Conference on Robotics and automation (ICRA)*, pages 5237–5244. IEEE, 2017.

[2] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto. Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1366–1373, 2017.

[3] Thomas Whelan, Michael Kaess, Maurice F. Fallon, Hordur Johannsson, John J. Leonard, and John B. McDonald. Kintinuous: Spatially extended kinectfusion. In *AAAI 2012*, 2012.

[4] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[5] Antoni Rosinol, Marcus Abate, Yun Chang, and Luca Carlone. Kimera: an open-source library for real-time metric-semantic localization and mapping. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[6] Thomas Whelan, Renato F Salas-Moreno, Ben Glocker, Andrew J Davison, and Stefan Leutenegger. Elasticfusion: Real-time dense SLAM and light source estimation. *The International Journal of Robotics Research*, 35(14): 1697–1716, 2016.

[7] Martin Rünz and Lourdes de Agapito. Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 10–20, 2018.

[8] Ioan Andrei Bârsan, Peidong Liu, Marc Pollefeys, and Andreas Geiger. Robust dense mapping for large-scale dynamic environments. *CoRR*, abs/1905.02781, 2019. URL http://arxiv.org/abs/1905.02781.

[9] Michael Strecke and Jörg Stückler. EM-fusion: Dynamic object-level SLAM with probabilistic data association. In *Proceedings IEEE/CVF International Conference on Computer Vision 2019 (ICCV)*, pages 5864–5873. IEEE, October 2019.

[10] Binbin Xu, Wenbin Li, Dimos Tzoumanikas, Michael Bloesch, Andrew J. Davison, and Stefan Leutenegger. MID-fusion: Octree-based object-level multi-instance dynamic SLAM. *2019 International Conference on Robotics and Automation (ICRA)*, pages 5231–5237, 2019.

[11] Johanna Wald, Armen Avetisyan, Nassir Navab, Federico Tombari, and Matthias Niessner. Rio: 3d object instance re-localization in changing indoor environments. In *Proceedings IEEE International Conference on Computer Vision (ICCV)*, 2019.

[12] Martin Rünz and Lourdes Agapito. Co-fusion: Real-time segmentation, tracking and fusion of multiple objects. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017.

[13] Lukas Maximilian Schmid, Jeffrey A. Delmerico, Johannes L. Schönberger, Juan I. Nieto, Marc Pollefeys, Roland Siegwart, and César Cadena. Panoptic multi-TSDFs: a flexible representation for online multi-resolution volumetric mapping and long-term dynamic scene consistency. *CoRR*, abs/2109.10165, 2021. URL https://arxiv.org/abs/2109.10165.

[14] Felix Endres, Jurgen Hess, Jurgen Sturm, Daniel Cremers, and Wolfram Burgard. 3-d mapping with an RGB-D camera. *Robotics, IEEE Transactions on*, 30:177–187, 02 2014.

[15] Mona Gridseth and Timothy Barfoot. Towards direct localization for visual teach and repeat. In *2019 16th Conference on Computer and Robot Vision (CRV)*, pages 97–104, 2019.

[16] Maik Keller, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision - 3DV 2013*, pages 1–8, 2013.

[17] Jakob Engel, Jörg Stückler, and Daniel Cremers. Large-scale direct SLAM with stereo cameras. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1935–1942, 2015.

[18] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 2013.

[19] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136. IEEE, 2011.

[20] Wenbin Li, Sajad Saeedi, John McCormac, Ronald Clark, Dimos Tzoumanikas, Qing Ye, Yuzhong Huang, Rui Tang, and Stefan Leutenegger. Interiornet: Mega-scale multi-sensor photo-realistic indoor scenes dataset. In *British Machine Vision Conference (BMVC)*, 2018.

[21] Edith Langer, Timothy Patten, and Markus Vincze. Robust and efficient object change detection by combining global semantic information and local geometric verification. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[22] Rares Ambrus, John Folkesson, and Patric Jensfelt. Unsupervised object segmentation through change detection in a long term autonomy scenario. *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 1181–1187, 2016.

[23] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[24] Xuesong Shi, Dongjiang Li, Pengpeng Zhao, Qinbin Tian, Yuxin Tian, Qiwei Long, Chunhao Zhu, Jingwei Song, Fei Qiao, Le Song, Yangquan Guo, Zhigang Wang, Yimin Zhang, Baoxing Qin, Wei Yang, Fangshi Wang, Rosa H. M. Chan, and Qi She. Are we ready for service robots? the OpenLORIS-Scene datasets for lifelong SLAM. In *2020 International Conference on Robotics and Automation (ICRA)*, pages 3139–3145, 2020.

[25] Jin-Man Park, Jae-hyuk Jang, Sahng-Min Yoo, Sun-Kyung Lee, Ue-hwan Kim, and Jong-Hwan Kim. ChangeSim: Towards end-to-end online scene change detection in industrial indoor environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021.

[26] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto. Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery. *IEEE Robotics and Automation Letters*, 4(3):3037–3044, July 2019.

[27] Renato F. Salas-Moreno, Richard A. Newcombe, Hauke Malte Strasdat, Paul H. J. Kelly, and Andrew J. Davison. SLAM++: Simultaneous localisation and mapping at the level of objects. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1352–1359, 2013.

[28] J. McCormac, R. Clark, Michael Bloesch, A. Davison, and Stefan Leutenegger. Fusion++: Volumetric object-level SLAM. *2018 International Conference on 3D Vision (3DV)*, pages 32–41, 2018.

[29] Antoni Rosinol, Marcus Abate, Jingnan Shi, and Luca Carlone. 3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans. In *Robotics: Science and Systems*, 07 2020.

[30] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[31] Irene Ballester, Alejandro Fontán, Javier Civera, Klaus H. Strobl, and Rudolph Triebel. DOT: dynamic object tracking for visual SLAM. *CoRR*, abs/2010.00052, 2020.

[32] Ting Sun, Yuxiang Sun, Ming Liu, and Dit-Yan Yeung. Movable-object-aware visual SLAM via weakly supervised semantic segmentation. *ArXiv*, abs/1906.03629, 2019.

[33] Chao Yu, Zuxin Liu, Xinjun Liu, Fugui Xie, Yi Yang, Qi Wei, and Fei Qiao. DS-SLAM: A semantic visual SLAM towards dynamic environments. *CoRR*, abs/1809.08379, 2018.

[34] Philipp Ruchti and Wolfram Burgard. Mapping with dynamic-object probabilities calculated from single 3d range scans. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6331–6336, 2018.

[35] Gian Diego Tipaldi, Daniel Meyer-Delius, and Wolfram Burgard. Lifelong localization in changing environments. *The International Journal of Robotics Research*, 32:1662 – 1678, 2013.

[36] Ryo Hachiuma, Christian Pirchheim, Dieter Schmalstieg, and H. Saito. Detectfusion: Detecting and segmenting both known and unknown dynamic objects in real-time SLAM. In *BMVC*, 2019.

[37] Chenjie Wang, Bin Luo, Yun Zhang, Qing Zhao, Lu-Jun Yin, Wei Wang, Xin Su, Yajun Wang, and Chengyuan Li. DymSLAM: 4d dynamic scene reconstruction based on geometrical motion segmentation. *IEEE Robotics and Automation Letters*, 6:550–557, 2021.

[38] Margarita Grinvald, Federico Tombari, Roland Y. Siegwart, and Juan I. Nieto. TSDF++: A multi-object formulation for dynamic object tracking and reconstruction. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14192–14198, 2021.

[39] Paul L. Rosin. Thresholding for change detection. *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pages 274–279, 1998.

[40] Richard J. Radke, Srinivas Andra, Omar Al-Kofahi, and Badrinath Roysam. Image change detection algorithms: a systematic survey. *IEEE Transactions on Image Processing*, 14:294–307, 2005.

[41] Yang Zhan, Kun Fu, Menglong Yan, Xian Sun, Hongqi Wang, and Xiaosong Qiu. Change detection based on deep siamese convolutional network for optical aerial images. *IEEE Geoscience and Remote Sensing Letters*, 14:1845–1849, 2017.

[42] Enqiang Guo, Xin sha Fu, Jiawei Zhu, Min Deng, Yu Liu, Qing Zhu, and Haifeng Li. Learning to measure change: Fully convolutional siamese metric networks for scene change detection. *ArXiv*, abs/1810.09111, 2018.

[43] Ashley Varghese, Jayavardhana Gubbi, Akshaya Ramaswamy, and P. Balamuralidhar. Changenet: A deep learning architecture for visual change detection. In *ECCV Workshops*, 2018.

[44] Pablo Fernández Alcantarilla, Simon Stent, Germán Ros, Roberto Arroyo, and Riccardo Gherardi. Street-view change detection with deconvolutional networks. *Autonomous Robots*, 42:1301–1322, 2018.

[45] Zi Jian Yew and Gim Hee Lee. City-scale scene change detection using point clouds. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13362–13369, 2021.

[46] Clara Gomez, Alejandra C. Hernandez, Erik Derner, Ramon Barber, and Robert Babuška. Object-based pose graph for dynamic indoor environments. *IEEE Robotics and Automation Letters*, 5(4):5401–5408, 2020.

[47] Liang-Chieh Chen, G. Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *ArXiv*, abs/1706.05587, 2017.

[48] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–4. IEEE, 2011.

[49] Edgar Sucar, Kentaro Wada, and Andrew J. Davison. Nodeslam: Neural object descriptors for multi-view shape reconstruction. *2020 International Conference on 3D Vision (3DV)*, pages 949–958, 2020.

[50] Jingwen Wang, Martin Rünz, and Lourdes de Agapito. DSP-SLAM: Object oriented SLAM with deep shape priors. *2021 International Conference on 3D Vision (3DV)*, pages 1362–1371, 2021.

[51] George Vogiatzis and Carlos Hernández. Video-based, real-time multi-view stereo. *Image and Vision Computing*, 29(7):434–441, 2011.

[52] Wei Dong, Qiuyuan Wang, Xin Wang, and Hongbin Zha. Psdf fusion: Probabilistic signed distance function for on-the-fly 3d data fusion and scene reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 701–717, 2018.

[53] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. SVO: Fast semi-direct monocular visual odometry. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, 2014.

[54] Matia Pizzoli, Christian Forster, and Davide Scaramuzza. Remode: Probabilistic, monocular dense reconstruction in real time. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2609–2616, 2014.

[55] Jun Yang, Dong Li, and Steven L. Waslander. Probabilistic multi-view fusion of active stereo depth maps for robotic bin-picking. *IEEE Robotics and Automation Letters*, 6:4472–4479, 2021.