TSDF-based Change Detection for Consistent Long-Term Dense Reconstruction and Dynamic Object Discovery

Marius Fehr[1], Fadri Furrer[1], Ivan Dryanovski[2],
Jürgen Sturm[2], Igor Gilitschenski[1], Roland Siegwart[1] and Cesar Cadena[1]

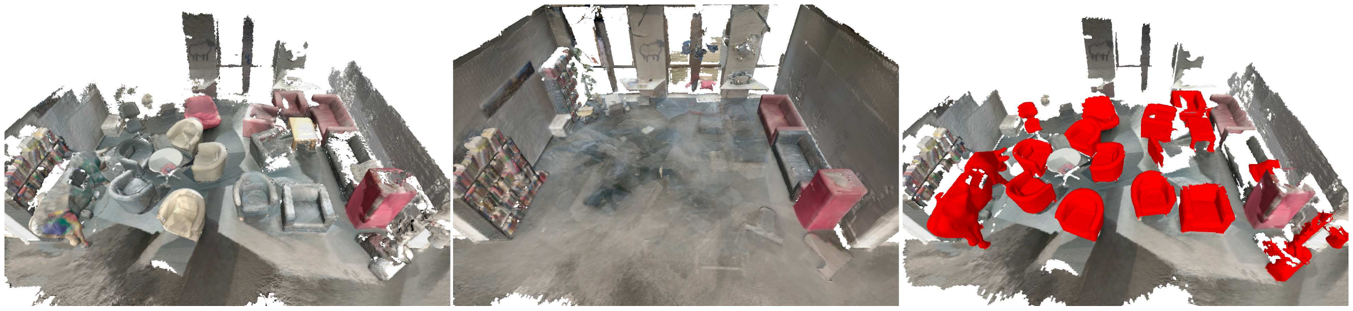


Fig. 1: **Change Detection Algorithm:** *(left)* One of 10 reconstructed scene observations. *(center)* Reconstruction of the static environment after 10 observations. *(right)* Discovered dynamic objects.

***Abstract*— Robots that are operating for extended periods of time need to be able to deal with changes in their environment and represent them adequately in their maps. In this paper, we present a novel 3D reconstruction algorithm based on an extended Truncated Signed Distance Function (TSDF) that enables to continuously refine the static map while simultaneously obtaining 3D reconstructions of dynamic objects in the scene. This is a challenging problem because map updates happen incrementally and are typically incomplete. Previous work on change detecting typically performs change detection on point clouds, surfels or maps, which are not able to distinguish between unexplored and empty space. In contrast, our TSDF-based representation naturally contains this information and thus allows us to more robustly solve the scene differencing problem. We demonstrate the algorithms performance as part of a system for unsupervised object discovery and class recognition. We evaluated our algorithm on challenging datasets that we recorded over several days with RGB-D enabled tablets. To stimulate further research in this area, all of our datasets will be made publicly available[3].**

I. INTRODUCTION

The ability of a robot to perceive and map the 3D structure of its environment is crucial to many applications such as life-long mapping and navigation. However, incrementally incorportating new 3D observations into a consistent map is difficult in the presence of dynamic objects. Furthermore, observations are noisy and generally incomplete so that reasoning about change (and its extent) is a non-trivial task. In this work, we propose an extension to TSDF based mapping that allows us to estimate consistent 3D reconstructions of the static map. Furthermore, our approach enables object discovery as an ongoing process during the robot's long-term day-to-day operation. Besides robotics, our work is also applicable to many other domains such as 3D building scanning where one wants all dynamic objects to be removed or warehouse logistics where it is relevant to find leftover objects and generally model the flow of goods.

We exploit the intrinsic properties of the volumetric TSDF-grid representation of 3D geometry to efficiently and robustly detect changes between different observation of the same scene and cluster and segment these changes into object candidates.

Our approach does not rely on viewpoint-dependent data such as camera poses, images and depth maps, once the reconstruction of the scene is complete. It also solves existing limitations such as assuming non-overlapping scene changes and completely overlapping observations. We furthermore employ an incremental object database to match and store the discovered objects and refine their 3D model with every discovered object instance.

The key contributions of this paper are:

- A novel estimation technique for computing consistent 3D reconstructions in dynamic environments based on a TSDF,
- a change detection algorithm that allows us to segment dynamic objects,
- a qualitative and quantitative evaluation of unsupervised object discovery and matching that combines the proposed change detection algorithm with an incremental object database, and
- a large corpus of datasets recorded with a low cost, mobile RGB-D tablet that we provide to community for further research and evaluation in long-term mapping in changing environments.

This reasearch was funded by Google Tango.

[1]Autonomous Systems Lab, ETH Zurich, Switzerland: `{marius.fehr, fadri.furrer}@mavt.ethz.ch`, `{igilitschenski, rsiegwart, cesarc}@ethz.ch`

[2]Google Inc.: `{idryanovski, jsturm}@google.com`

[3]Dataset: `github.com/ethz-asl/change_detection_ds`

II. RELATED WORK

There has been some research into solving the problem of long-term autonomous mapping and navigation for robots. Many approaches agree that in order to operate in a dynamic world the robots representation of the environment needs to either model these dynamics or has to be able to distinguish static from dynamic map elements. There have been different grid based approaches to model the dynamic world for the purpose of improving 2D mapping. Walcott-Bryant *et al.* [1] introduces dynamic pose graph SLAM by adding time as an additional dimension and using change detection on an occupancy grid to identify and discard obsolete poses. Saarinen *et al.* [2] approximates the dynamic environment using an occupancy grid map with the assumption that each cell is an independent Markov chain.

With the advent of low-cost 3D sensors the same problem arose also in consistent dense long-term mapping and reconstruction and lead to an increasing amount research in this field. Andreasson *et al.* [3] applies a 3D version of the Normal Distribution Transform (NDT) to detect not only geometry but also color changes between a reference model and a new observation. Saarinen *et al.* [4] recently further improved the efficiency and real-time capability of the system. Even though the NDT allows the computation of probabilistic changes it is not suitable for surface reconstruction and hence will not allow the extraction of object surface models.

Recent publications furthermore focused on exploiting the dynamics of the environment to learn about the scene or more specifically about the dynamic objects. Herbst *et al.* [5] employs probabilistic sensor models for depth, color and surface normals and segments a Markov Random Field (MRF) using graph cuts to extract dynamic objects in surfel-based reconstructions. Their work employs spectral clustering to compute object classes, however the performance significantly decreases if the number of classes is unknown in advance. In more recent work Herbst and Fox [6] demonstrated a system for real-time dynamic object segmentation and modelling on desk scenes. It keeps track of multiple TSDF-based reconstructions of the scene that are split or merged based on frame-to-model change detection. Finman *et al.* [7] exploit change detection using a free space ray-casting algorithm to learn the best segmentation method for every dynamic object. This facilitates segmenting dynamic objects in scenes where they have not moved yet. More recent in [8] they show how these segmented objects can then be used for place recognition.

Ambrus *et al.* [9] proposed a change detection system that computes a meta-room, i.e. a 3D model of the static environment, from point cloud based 3D reconstructions recorded autonomously by a robotic platform. Their algorithm assumes complete observation overlap and uses occlusion checks to distinguish real changes from segments that have been occluded by dynamic objects. The objects are then clustered by proximity across observations and shape using global descriptors. The meta-room algorithm has very recently been applied by Fäulhammer *et al.* [10] in a system to navigate a robotic platform towards a change cluster to obtain a better, more detailed object model in the process. The algorithms and data structures of [5], [7] and [9] require recomputation or approximation of free-space/occupancy information or visibility/occlusion checks, whereas our approach exploits the intrinsic properties of TSDFs, hence no additional such effort is required.

At this point it is important to point out that there are different time scales for the dynamics in an environment. A concept explored by Biber and Duckett [11] by introducing a dynamic 2D laser maps that represent the environment and its dynamics over multiple time scales. Our work as well as [5], [7] and [9] focus solely on change that happens in between observations. In contrast, Schmidt *et al.* [12] make use of Signed Distance Function (SDF) submaps to track articulated objects that move during a single observation.

Other work focuses on using the results of change detection algorithms to analyse the spatial-temporal behaviour of changes. Ambrus *et al.* [13] further improve their object matching performance by modelling the spatial-temporal distribution of change segments. Krajnik et al. [14] assumes periodicity of the changes and calculates the frequency spectra of long-term observations of the environment to compute the probability of a certain state and hence predict its future state.

III. METHODOLOGY

In Section III-A we describe how we obtain the aligned 3D reconstructions. In Section III-B and III-C we introduce the change detection algorithm and dynamic objects segmentation. In Section III-D we summarize the incremental object database we employ for object discovery.

A. 3D Reconstruction

1) Input Data: The proposed algorithm processes aligned 3D reconstructions and therefore is built on top of a SLAM system that is used to ensure the robots long-term localization. It provides accurate trajectories and it allows us to align and optimize new ones with respect to the previously recorded map.

Hence, we assume every observation $O_i \in \mathbb{O}$, where $N = |\mathbb{O}|$, consists of an aligned visual-inertial pose-graph based map with RGB-D measurements associated with each camera position.

2) TSDF Representation: After alignment, the RGB-D data is used to create a 3D reconstruction represented in a TSDF in the world coordinate frame. The TSDF is stored in a volumetric grid of resolution r. This allows us to directly compare the TSDF voxel values of each observation. A single cell (voxel) of the volumetric grid, indexed with $\iota \in \mathbb{Z}^3$, stores the signed distance function value $f \in [-t, +t]$, where t is the truncation distance. It also keeps track of the voxel weight $w \in \mathbb{R}_0^+$ and its RGB color c. A new depth measurement d that observes point P from camera position X is integrated into the grid by ray tracing in between X and P and computing the truncated signed distance. We define

s to be the signed distance between voxel center V and point P, which is positive in front and negative behind the observed surface and can be truncated as follows:

$$f_d = \begin{cases} t, & sdf \geq t \\ sdf, & sdf \in [-t, t] \end{cases} \quad (1)$$

Hence, for every voxel the new measurements for f_d, c_d and w_d are fused with the existing values as follows:

$$\begin{aligned} f_{n+1} &= \frac{f_n \cdot w_n + f_d \cdot w_d}{w_n + w_d} \\ c_{n+1} &= \frac{c_n \cdot w_n + c_d \cdot w_d}{w_n + w_d} \\ w_{n+1} &= w + n + w_d \end{aligned} \quad (2)$$

The volumetric grid employs a voxel hashing scheme [15] to improve the scalability of the reconstruction. A marching cube algorithm [16] is used for surface reconstruction.

After reconstructing the scene observations all the voxels are filtered by applying a threshold τ to the voxel weight w. This removes sections of the reconstruction that have received very few measurements, which also covers highly dynamic objects passing in front of the sensor.

B. Change detection

1) Initialization: The algorithm keeps track of the following data:

- A multi-layered volumetric grid $\mathcal{M}$ that stores the TSDF grid M_i and the voxel indices I_i for each observation such that:

$$\forall i \in [0, N] \quad \mathcal{M}(i, \iota) = \begin{cases} M_i(\iota), & \textit{if } \iota \in I_i \\ \varnothing, & \textit{otherwise} \end{cases} \quad (3)$$

- A set of all voxel indices containing measurements: $\mathcal{I} = \bigcup_{i \in [0,N]} I_i$
- A volumetric TSDF grid $\mathcal{S}$ and voxel indices $I_{\mathcal{S}}$ that represent the current best estimate of the static part of the world.
- A label grid $\mathcal{D}$ to store intermediate voxel label results: $\mathcal{D}(\iota) \in \{\textit{static}, \textit{dynamic}\}\ \forall \iota \in \mathcal{I}$.

With the first observation O_0 the reconstruction M_0 is computed and stored in $\mathcal{M}$ and $\mathcal{I}$ is initialized to I_0. $\mathcal{S}$ and $\mathcal{D}$ are initialized to the first observation such that $\mathcal{S}(\iota) = M_0(\iota)$ and $\mathcal{D}(\iota) = \textit{static}\ \forall \iota \in \mathcal{I}$.

2) Update Overview: The update consists of incorporating a new observation into the static reconstruction (See Fig. 2). A new observation O_i is made and aligned against the previous observations based on the sparse map using keypoint-based visual loop closure and batch optimization. The 3D reconstruction M_i is computed and stored in $\mathcal{M}$ and the voxel indices $\mathcal{I}$ are updated with I_i. From this point onward, the original trajectory and viewpoint dependent data, such as camera poses, images and depth maps is not needed any more for change detection. However the sparse, pose-graph-based map and the keypoints need to be saved and ideally summarized to allow for long term robot localization and trajectory alignment.

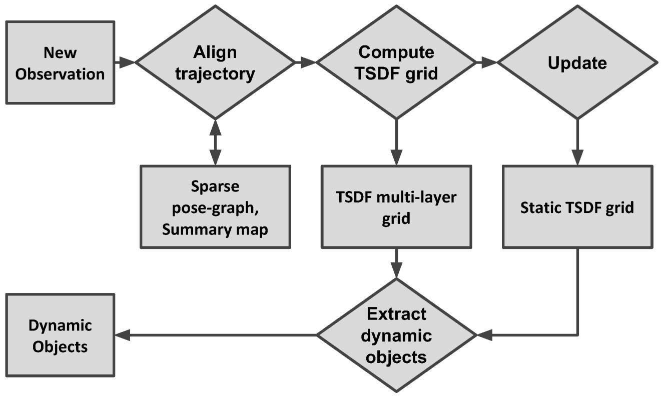


Fig. 2: **Change Detection Algorithm Overview**

After alignment the new reconstruction M_i is compared to $\mathcal{S}$ and coarse dynamic clusters are identified. The static map is first updated by merging all the voxel outside the dynamic clusters using the default weighted average update scheme (Eq. 2). For all voxels within the dynamic cluster we apply a more elaborate merging strategy to cope with occlusion and noise and in the process segment the remaining static geometry from the objects. The resulting, more fine grained, voxel labels are then applied to the surface mesh of the 3D reconstruction M_i and the segmentation is refined using a region-growing algorithm. Finally we use a connected-components algorithm to segment the changes into individual objects.

3) Identifying Dynamic Clusters: The scene differencing happens in two steps: First we update the label grid $\mathcal{D}$ whose labels are coarsely identifying the dynamic clusters, while filtering out dynamic labels that originate from noise. To that end we compare the new observation M_i with the previous static reconstruction $\mathcal{S}$ and compute:

$$\delta_\iota = |f_{M_i(\iota)} - f_{\mathcal{S}(\iota)}| \qquad \forall \iota \in I_{\mathcal{S}} \cap I_i \quad (4)$$

By applying the threshold θ we obtain the label grid $\mathcal{D}$, such that $\mathcal{D}(\iota) = \textit{dynamic}$ if $\delta_\iota > \theta$. In order to reduce false positive *dynamic* labels caused by sensor and systematic noise we filter $\mathcal{D}$ using a 3D erosion kernel. We define the kernel at voxel index ι

$$K_\iota = \{\kappa \mid |\kappa - \iota| \leq [\rho, \rho, \rho]\} \quad (5)$$

and for label λ

$$\text{count}(\iota, \lambda) = |\{\kappa \mid \kappa \in K_\iota,\ \mathcal{D}(\kappa) = \lambda\}| \quad (6)$$

A *dynamic* label at $\mathcal{D}(\iota)$ is therefore removed if $\textit{count}(\iota, \textit{dynamic}) \leq \alpha * |K|$. We then apply a bigger dilation kernel to the remaining *dynamic* labels to recover the eroded geometry. Fig. 3 shows the filtering process on the label map computed from two different observations.

4) Updating the Static Reconstruction: After identifying the dynamic clusters by labeling them in $\mathcal{D}$ we are able to compute the consistent reconstruction of the static part of the environment. To this end we exploit the intrinsic properties of the TSDF which allows us to elegantly solve the following two problems encountered by other 3D representation such as point clouds, meshes and surfels: *How do we identify*

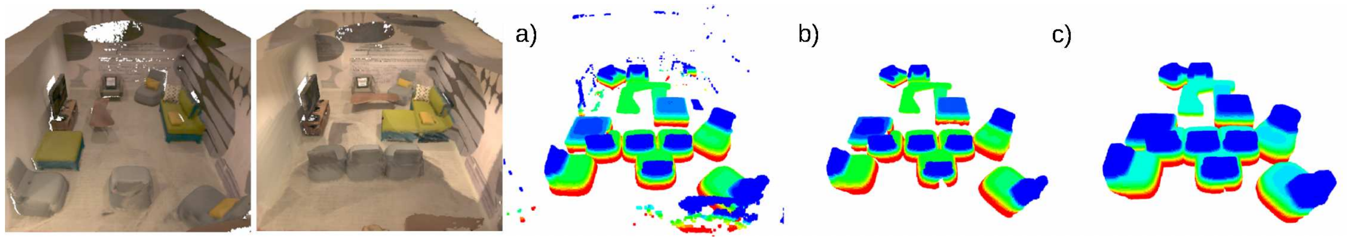


Fig. 3: **Filtering of label grid $\mathcal{D}$:** Example of the first coarse change detection step applied to two reconstructions (*left*). **a) - c)** show the state of the label grid $\mathcal{D}$. For better visibility $\mathcal{D}$ is colored by height. **a)** Voxel labels that are obtained by applying a threshold θ to the raw TSDF difference of the two reconstructions (*left*). **b)** The labels are filtered using an erosion kernel to reduce false positives caused by sensor and systematic noise in the reconstructions. **c)** A dilation kernel is applied to recover the original geometry.

changes originating from differences in scene coverage of the observations? and *How do we identify changes caused by occlusion of dynamic objects?* This occluded geometry is needed to complete the static reconstruction.

The former can either be solved by assuming complete overlap of the observations [9] or by recomputing the free space information [7]. The TSDF grid already models free space ($f = t \wedge w > 0$) and unobserved space ($w = 0$) and therefore solves this problem with no additional computational effort.

The second problem was solved in previous work [9] by projecting the detected change segments onto a sphere around the viewpoint and checking for occlusion. However this prohibits overlapping changes, e.g. a dynamic object partially occluding another. With the TSDF grid we can solve this without any viewpoint dependent data: Let us assume we have N noise-free observations resulting in N perfectly aligned TSDF grids. Hence by comparing the TSDF values $f_{(\iota,i)}$ for every voxel at index ι the best estimate of the static map can simply be computed as follows:

$$S(\iota) = \max(f_{(\iota,i)} \mid i \in [0, N] \wedge w_{(\iota,i)} > 0) \tag{7}$$

An intuitive explanation is that adding an object can only ever **decrease** the signed distance from each voxel to the next surface, whereas removing an object can only ever **increase** it. With this in mind the static reconstruction $\mathcal{S}$ is updated as follows: In a first step all voxels that are labeled *static* in $\mathcal{D}$ are merged with $\mathcal{S}$ such that:

$$\forall \kappa \in \{\iota \mid \iota \in \mathcal{I} \wedge \mathcal{D}(\iota) = \textit{ static}\}$$

$$\mathcal{S}_{new}(\kappa) = \begin{cases} \mathcal{S}(\kappa), & if\ w_{\mathcal{M}(N,\kappa)} = 0 \\ \mathcal{M}(N,\kappa), & if\ w_{\mathcal{S}(\iota)} = 0 \\ \mathcal{S}(\kappa) \diamond \mathcal{M}(N,\kappa), & otherwise \end{cases} \tag{8}$$

where the voxel merging operation $\diamond$ is defined by the TSDF update Eq. 2. For all voxel labeled *dynamic* we modify Eq. 7 into an algorithm that accounts for the fact that we do **not** have noise-free measurements and perfect alignment.

$$\forall \kappa \in \{\iota \mid \iota \in \mathcal{I} \wedge \mathcal{D}(\iota) = \textit{ dynamic}\}$$

$$\mathcal{S}_{new}(\kappa) = \begin{cases} \mathcal{S}(\kappa), & if\ f_{\mathcal{S}(\kappa)} > (f_{\mathcal{M}(N,\kappa)} + \theta) \\ \mathcal{M}(N,\kappa), & if\ f_{\mathcal{M}(N,\kappa)} > (f_{\mathcal{S}(\kappa)} + \theta) \\ \mathcal{S}(\kappa) \diamond \mathcal{M}(N,\kappa), & if\ |f_{\mathcal{M}(N,\kappa)} - f_{\mathcal{S}(\kappa)}| \le \theta \\ \mathcal{S}(\kappa) \diamond \mathcal{M}(N,\kappa), & if\ f_{\mathcal{S}(\kappa)} = t \wedge f_{\mathcal{M}(N,\kappa)} = t \\ \mathcal{S}(\kappa), & if\ f_{\mathcal{S}(\kappa)} = t \wedge f_{\mathcal{M}(N,\kappa)} \neq t \\ \mathcal{M}(N,\kappa), & if\ f_{\mathcal{S}(\kappa)} \neq t \wedge f_{\mathcal{M}(N,\kappa)} = t \end{cases} \tag{9}$$

Intuitively this algorithm approximates the static TSDF value by merging a new value if it is close to the current estimate. If a new value is found that is significantly larger it replaces the current estimate.

C. Dynamic Object Segmentation

In order to extract the dynamic objects we compare the reconstruction M_i to our current best estimate of the static reconstruction $\mathcal{S}$ and compute a refined label grid D_i.

$$\forall \kappa \in \{\iota \mid \iota \in I_i \wedge \mathcal{D}(\iota) = \textit{ dynamic}\}$$

$$D_i(\kappa) = \begin{cases} \textit{dynamic}, & if\ \ f_{M_i(\kappa)} < f_{\mathcal{S}(\kappa)} \\ & or\ \ w_{\mathcal{S}(\kappa)} = 0 \\ \textit{static}, & otherwise \end{cases} \tag{10}$$

The voxel labels are then applied to the vertices of the surface mesh R_i. These vertex labels are then refined using a simple mesh region-growing algorithm [17]. A region is initialized to a single mesh face. The algorithm adds a neighboring face to the region if its normal does not significantly deviate from the regions current average normal. If a region cannot grow any more a new region is initialized. Finally small regions are merged with neighboring regions with similar average normal. The final set of regions are then used to grow the dynamic label across all the region if the ratio of dynamic faces exceeds a threshold ϕ. Then the mesh is segmented based on the labels. In order to get individual objects we apply a simple connected component algorithm, i.e. all dynamic surfaces that are connected are considered a single object. Finally all objects are filtered by discarding planar objects based on their singular values. Furthermore we impose a minimum size in terms of number of vertices on the objects.

D. Incremental Object Database

The last component of the proposed system merges and tracks the extracted dynamic objects and improves their 3D models with every object instance that is discovered. To that end we employ the unsupervised incremental object database described in [18]. For the readers convenience we will briefly summarize the referenced system.

The system starts from an empty object database, no previously scanned objects or any other training is involved. A new object, represented by its 3D point cloud, is added to the database where additional properties are computed, e.g. scale, confidence, normals, 3D keypoints (ISS [19]) and descriptors (FPFH [20]) and a list of poses of matched object instances. In a cleaning/merging step the new object is matched against the database object (1 to n) or a full database merging step is started (n to n). The database treats identical objects in a single observation as different instances of the same object. The object matching can be summarized as follows:

- 3D feature-based initial alignment using RANSAC.
- Refinement of the alignment using ICP.
- The best database matches is determined based on ICP score and inlier ratio.
- A scale- and confidence-aware surface reconstruction algorithm [21] is used to merge the set of matching objects and improve the object model.

IV. EXPERIMENTS AND RESULTS

A. Setup

The proposed system was implemented based on the following existing components:

We use Google Tango tablets as sensor for all our datasets, most notably its RGB-D sensor which has a resolution of $320 \times 180px$ and an operating range of $0.4 - 4.0m$. Hence we use the Google Tango [22] visual-inertial odometry and mapping framework for both the initial trajectory estimates and the alignment of individual observations. The framework creates sparse, pose-graph based maps with associated IMU measurements, feature tracks based on lens-invariant Freak [23] descriptors and triangulated feature matches. The sparse maps are aligned and refined using visual keypoint based loop closure [24] and batch optimization. The dense 3D reconstruction is based on the Google Tango framework as well, which is closely related to its OpenSource version OpenChisel [25].

In order to evaluate our system we recorded three challenging indoor datasets:

a) living-room: This is the baseline dataset and consists of 9 hand-held trajectories in a controlled indoor environment, see Fig. 3 (left). It provides nearly 100% observation overlap and also provides depth measurements from a large variety of viewpoints for most objects resulting in 3D models with a high coverage. The scene changes not only overlap in between observations but the dynamic objects also come in contact with different other objects.

b) office: This dataset consists of 4 observations of a controlled office environment recorded from a single point at the center of the room using a tripod, see Fig. 4 (top). Hence the overlap of the observations is close to 100%. Its purpose is to be able to compare our approach to the meta-room algorithm which assumes a robotic platform with a pan-tilt RGB-D sensor unit that scans a single, convex room from a central point.

c) lounge: This is the most challenging dataset and consists of 10 hand-held trajectories in an uncontrolled, challenging environment, a highly frequented meeting area/office lounge over the course of two weeks where objects are shifted on a daily basis, see Fig. 1. The observation overlap varies between approx. $50-100\%$ and many dynamic objects are only partially observed.

Table I lists the parameters used to process each dataset.

TABLE I

Parameter	***office***	***living-room***	***lounge***
Grid resolution: r **[m]**	0.02		
Truncation distance: t **[m]**	0.1		
TSDF diff threshold: θ **[m]**	0.05	0.07	
Erosion kernel: ρ_e **[voxel]**	3	5	
Erosion kernel: α	0.5		
Dilation kernel: ρ_d **[voxel]**	5	7	
Region-growing ratio: ϕ	0.25		
Minimum TSDF voxel weight: τ	10		

For all experiments we align and reconstruct all observations as described Section III. The reconstructions are then used to compute the consistent static reconstruction. Subsequently the most recent static reconstruction is used to extract the dynamic objects for all the observations. Hence the result consists of a single static reconstruction and a set of dynamic object candidates for every observation. The extracted dynamic objects candidates are then quantitatively evaluated by counting the correctly identified dynamic object and by analyzing over- and under-segmentation of the objects based on a hand-labeled ground truth. This means we counted the number of objects that have been split into multiple object candidates and the ones that have been combined into a single object candidate. It is important to note that this segmentation metric also counts objects that have been perceived as multiple objects due to the partial observation and objects that have been in close contact with other object and without further information are indistinguishable from a single object. We furthermore evaluate the convergence of the static reconstruction by comparing each intermediate result to a manually cleaned up version of the final reconstruction. For a qualitative evaluation of the proposed system for object discovery, we gradually insert all object candidates into the incremental object database and perform a cleaning/merging step as described in Subsection III-D.

B. Experiments

We will evaluate the full change detection algorithm for all datasets using the above described metrics. A quantitative and qualitative comparison of our change detection algorithm to the meta-room algorithm [9] was performed on the *office* dataset. In order to demonstrate how the results of our change detection algorithm can be used as for object discovery, we will show qualitative result of the generated object classes and improved 3D models after inserting the objects into the object database (See Section III-D).

C. Results

1) Change Detection: Fig. 5 shows qualitative results of the change detection algorithm for all 3 datasets, more specifically the results of the dynamic object segmentation (in *red*) for one of the observations (*top*) and the final state of the reconstruction of the static environment after processing all observations (*bottom*).

The algorithm successfully detects and removes all dynamic objects from the reconstruction and converges to a consistent 3D model of the static environment. Highlighted in blue are some minor artifacts resulting from imperfect segmentation at the object/static environment boundaries.

TABLE II: Evaluation of change detection based on number of correctly detected object changes.

Dataset	TP	FP	FN	Precision	Recall
living-room (ours)	62	0	1	100%	98%
lounge (ours)	137	0	12	100%	92%
office (ours)	23	0	0	100%	100%
office (meta-room)	20	1	3	95%	87%

Table II shows the quantitative results of the change detection by evaluating the number of correctly extracted dynamic objects. On the evaluated datasets we successfully maintained 100% precision while reaching high recall rates. Most false negatives are caused by filtering out planar objects such as the couch table in *living-room* or a partially reconstructed cushion or table in *lounge*, see Fig. 1. These misclassification errors happen at the very end of the pipeline, hence the static reconstruction is still updated correctly.

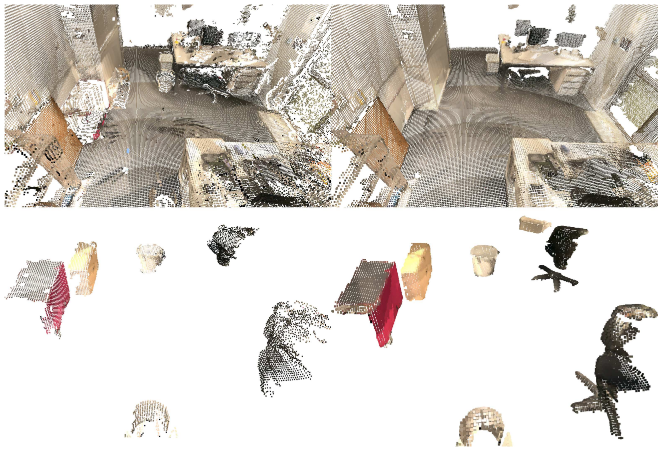


Fig. 4: **Comparison of our approach *(right)* to the meta-room [9] algorithm *(left)*.** Static scene after processing all 4 observations (top). Our approach exhibits a cleaner and more complete segmentation and a better completion of the scene geometry that was occluded by objects. Our extracted objects (*bottom right*) are more complete and some of the smaller changes are missing with the meta-room based algorithm (*bottom left*). See table II for details.

Fig. 4 shows the the comparison between our approach and the meta-room algorithm [9] applied to the *office* dataset. Our static reconstruction exhibits fewer outliers and thus appears slightly cleaner than the one generated by meta-room. We attribute this at least partially to our TSDF weight based filtering as opposed to the Statistical Outlier Rejection (SOR) filtering employed by meta-room, which filters based on the distribution of the point neighbors. Futhermore we observe that for the meta-room algorithm segments that were occluded by dynamic objects appear sparser than with our approach and most segmented dynamic objects leave point artifacts behind.

2) Static Reconstruction: Fig. 6a, 6b and 6c show the absolute and relative number of points whose point-to-plane distance to the reference reconstruction is smaller than θ. For *living-room* and *office* we observe the expected convergence, as these controlled datasets were designed such that all static surfaces that were occluded by dynamic objects are eventually observed and thus can be added to the static reconstruction. The dataset *lounge* requires all observations to converge but despite the fact it was recorded in an uncontrolled environment the final static reconstructions contains only one small partial object artifact (center of the room, highlighted in *blue*) and minor artifacts near the static surface.

3) Object Segmentation: In contrast to the previous sections that focused on evaluating the change detection and static reconstruction computation, in this section we will evaluate the object segmentation, i.e. how well the segments represent complete objects. Table III shows the over- and undersegmentation of objects. Our algorithm has a clear tendency to oversegment objects as a direct consequence of the decision to use a connected components algorithm on the labelled mesh. The meta-room on the other hand employs a Euclidean clustering algorithm to obtain change segments and performs significantly better on the *office* dataset. Fig. 7 shows examples of merged object models as well as under- and oversegmented objects. Fig. 8 shows how the region-growing algorithm is used to propagate dynamic labels across nearly planar regions and results in a better, more complete object segmentation.

4) Object Discovery: Fig. 7 shows examples of object candidates that have been merged by the incremental

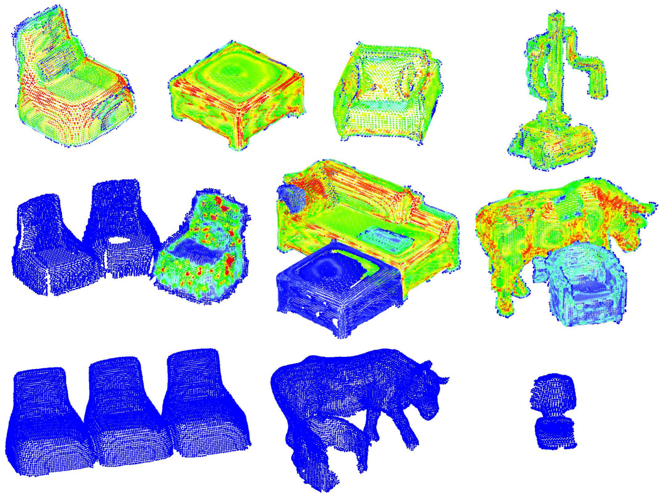


Fig. 7: Examples of merged database object models from under- and oversegmented object candidates. (*top*) correctly segmented and merged (*center*) undersegmented and partially matched (*bottom*) under- or oversegmented without matches.

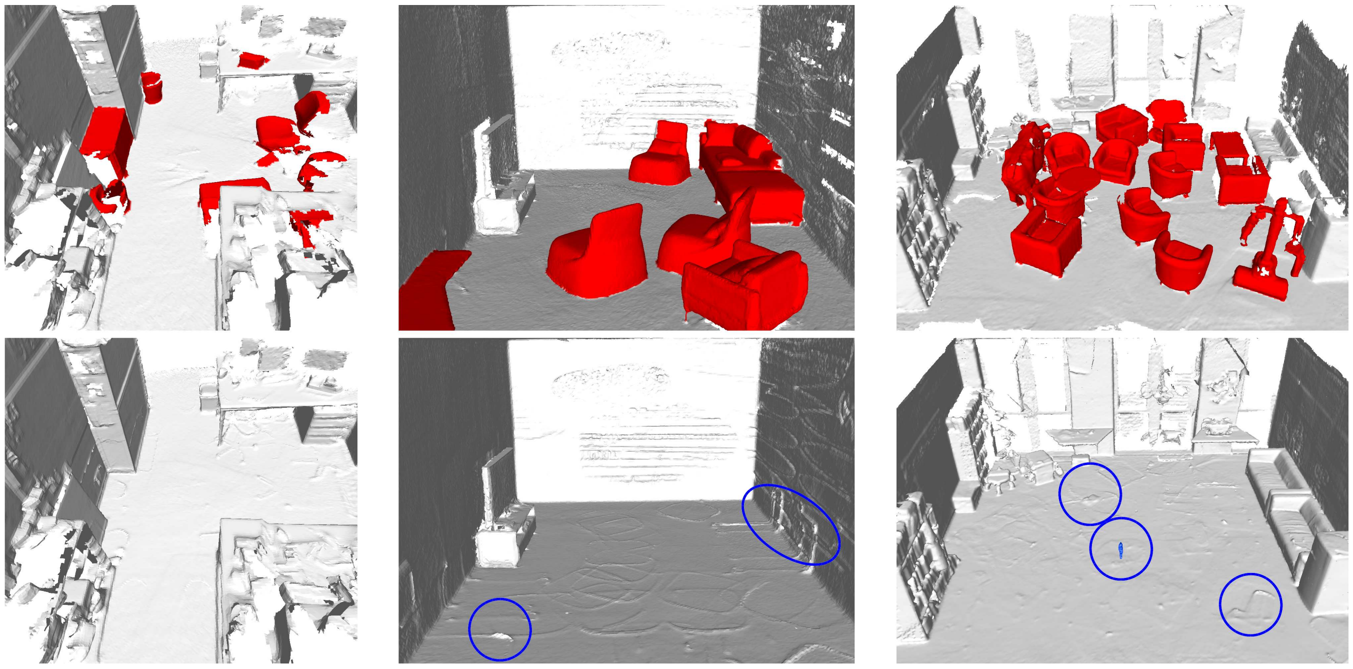
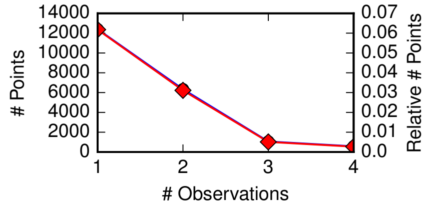
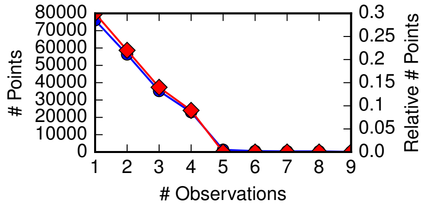


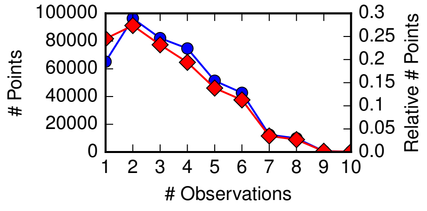
Fig. 5: **Change detection result for all datasets:** (*left*) *office* (controlled) (*center*) *living-room* (controlled) (*right*) *lounge* (uncontrolled) (*top*) Discovered and segmented dynamic objects, highlighted in *red*. (*bottom*) Computed reconstruction of the static environment after 4 (*office*), 9 (*living-room*) and 10 (*lounge*) observations. Artifacts, i.e. remaining dynamic surfaces, are highlighted in *blue*. See table II for the quantitative change detection evaluation.



(a) Office Dataset.



(b) Living-room Dataset.



(c) Lounge Dataset.

Fig. 6: **Convergence of the Static Reconstruction:** Comparison of static reconstruction to a manually cleaned-up reconstruction. Number of points (absolute in blue, relative in red) with a point-to-plane distance greater than θ (5cm).

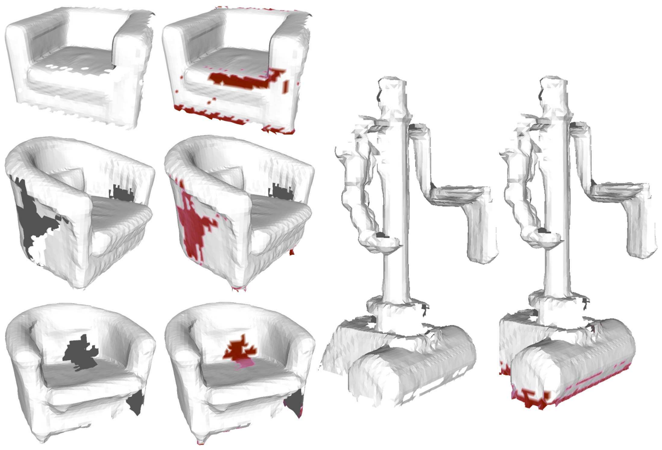


Fig. 8: Segmented dynamic objects with (*right*) and without (*left*) label refinement/propagation using region-growing. The refined surfaces are highlighted in red.

TABLE III: Evaluation of object segmentation. For the office dataset we compare our approach to the meta-room [9]

Dataset/Algorithm	Over-segmented segments	Under-segmented segments
living-room (ours)	0	5 / 9%
lounge (ours)	30 / 23%	15 / 11%
office (ours)	26 / 68%	0
office (meta-room)	2 / 9%	0

database, colored by confidence (from low (*blue*) to high (*green* to *red*)). The *living-room* dataset contains 9 distinct objects (by shape), the database merged 56 object candidate segments into 15 database objects. The *lounge* dataset contains 17 distinct objects (deformed bean bag and robot arms are separate objects), the database merged 134 object candidate segments into 48 database objects.

V. DISCUSSION

We have presented a change detection algorithm that exploits the intrinsic properties of TSDF grids to solve the scene differencing problem and maintain a consistent 3D model of the static environment. Our approach does not

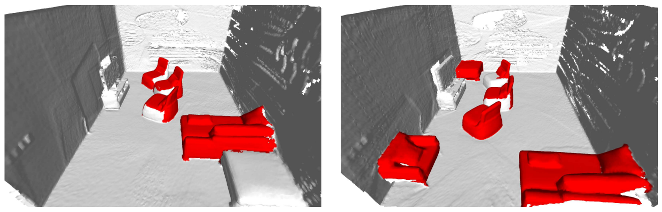


Fig. 9: **Limitation:** Even though region growing improves the segmentation it cannot fully solve segmentation in case of few observations (2) and overlapping changes.

require viewpoint dependent data once the observations are reconstructed and filtering based on TSDF weight and 3D erosion kernel has proven to be effective in removing noise and outliers. We furthermore presented a method to cluster and extract the scene changes that employs a region-growing algorithm to improve the segmentation. We demonstrated the application of our change detection algorithm by using an incremental object database to match the dynamic objects and compute merged 3D models.

We acknowledge the fact that our change detection approach is dependent on a correct, pose-graph-based alignment of the observations. However unlike other approaches we refrain from further refining the scene alignment using 3D features or ICP, because from our experience the trajectory-based alignment is more likely to exhibit non-rigid/local misalignment that cannot be solved with a simple global transformation.

Fig. 9 demonstrates the limit of the segmentation despite region-growing refinement. The most limiting factor of the proposed system however is still the over- and undersegmentation of objects that complicates creating a consistent object database. We deliberately chose to use connected component clustering over euclidean clustering obtaining potential oversegmentations as we believe it is easier to merge object segments in the database than to solve segment splits.

When planar segments are extracted from valid structural changes we reject them because they are highly ambiguous as object candidates for matching and merging with other objects in the database.

We furthermore want to point out that the proposed system does not offer a complete semantic mapping solution because there are no semantic labels assigned for the matched objects.

In future work we would like to better identify and separate under- and oversegmented objects by using the observations TSDF grids to enforce visibility constraints on the improved 3D objects. This could not only provide a quality measure but also allow us to further improve and filter the database objects.

References

[1] A. Walcott-Bryant, M. Kaess, H. Johannsson, and J. J. Leonard, "Dynamic pose graph slam: Long-term mapping in low dynamic environments," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 1871–1878, IEEE, 2012.

[2] J. Saarinen, H. Andreasson, and A. J. Lilienthal, "Independent markov chain occupancy grid maps for representation of dynamic environment," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3489–3495, IEEE, 2012.

[3] H. Andreasson, M. Magnusson, and A. Lilienthal, "Has something changed here? autonomous difference detection for security patrol robots," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3429–3435, IEEE, 2007.

[4] J. P. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, "3d normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments," *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1627–1644, 2013.

[5] E. Herbst, X. Ren, and D. Fox, "Rgb-d object discovery via multi-scene analysis," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4850–4856, IEEE, 2011.

[6] E. Herbst and D. Fox, "Toward Online 3-D Object Segmentation and Mapping," *2014 IEEE International Conference on Robotics and Automation*, pp. 3193–3200, 2014.

[7] R. Finman, T. Whelan, M. Kaess, and J. J. Leonard, "Toward lifelong object segmentation from change detection in dense rgb-d maps," in *Mobile Robots (ECMR), 2013 European Conference on*, pp. 178–185, IEEE, 2013.

[8] R. Finman, L. Paull, and J. J. Leonard, "Toward object-based place recognition in dense rgb-d maps," in *ICRA Workshop Visual Place Recognition in Changing Environments, Seattle, WA*, 2015.

[9] R. Ambruş, N. Bore, J. Folkesson, and P. Jensfelt, "Meta-rooms: Building and maintaining long term spatial models in a dynamic world," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1854–1861, IEEE, 2014.

[10] T. Fulhammer, R. Ambru, C. Burbridge, M. Zillich, J. Folkesson, N. Hawes, P. Jensfelt, and M. Vincze, "Autonomous learning of object models on a mobile robot," *IEEE Robotics and Automation Letters*, vol. 2, pp. 26–33, Jan 2017.

[11] P. Biber, T. Duckett, *et al.*, "Dynamic maps for long-term operation of mobile service robots.," in *Robotics: science and systems*, pp. 17–24, 2005.

[12] T. Schmidt, R. Newcombe, and D. Fox, "Dart: dense articulated real-time tracking with consumer depth cameras," *Autonomous Robots*, vol. 39, no. 3, pp. 239–258, 2015.

[13] R. Ambrus, J. Ekekrantz, J. Folkesson, and P. Jensfelt, "Unsupervised learning of spatial-temporal models of objects in a long-term autonomy scenario," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 5678–5685, IEEE, 2015.

[14] T. Krajník, J. P. Fentanes, G. Cielniak, C. Dondrup, and T. Duckett, "Spectral analysis for long-term robotic mapping," in *International Conference on Robotics and Automation (ICRA)*, 2014.

[15] M. Niessner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3D Reconstruction at Scale Using Voxel Hashing," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 169:1—-169:11, 2013.

[16] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," in *ACM siggraph computer graphics*, vol. 21, pp. 163–169, ACM, 1987.

[17] A. Shamir, "A survey on mesh segmentation techniques," *Computer Graphics Forum*, vol. 27, no. 6, pp. 1539–1556, 2008.

[18] F. Furrer, T. Novkovic, A. Gawel, M. Fehr, T. Sattler, M. Pollefeys, and R. Siegwart, "Unsupervised incremental construction of 3D object databases from RGB-D data," 2016.

[19] Y. Zhong, "Intrinsic shape signatures: A shape descriptor for 3d object recognition," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pp. 689–696, IEEE, 2009.

[20] R. B. Rusu, Z. C. Marton, N. Blodow, and M. Beetz, "Persistent point feature histograms for 3d point clouds," in *Proc 10th Int Conf Intel Autonomous Syst (IAS-10), Baden-Baden, Germany*, pp. 119–128, 2008.

[21] S. Fuhrmann and M. Goesele, "Floating scale surface reconstruction," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, p. 46, 2014.

[22] Google, "ATAP Project Tango Google," Feb. 2014.

[23] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*, pp. 510–517, Ieee, 2012.

[24] S. Lynen, T. Sattler, M. Bosse, J. Hesch, M. Pollefeys, and R. Siegwart, "Get out of my lab: Large-scale, real-time visual-inertial localization," 2015.

[25] M. Klingensmith, I. Dryanovski, S. Srinivasa, and J. Xiao, "Chisel: Real time large scale 3d reconstruction onboard a mobile device," in *Robotics Science and Systems 2015*, July 2015.