SLAM with Object Discovery, Modeling and Mapping

Siddharth Choudhary, Alexander J. B. Trevor, Henrik I. Christensen, Frank Dellaert

***Abstract*— Object discovery and modeling have been widely studied in the computer vision and robotics communities. SLAM approaches that make use of objects and higher level features have also recently been proposed. Using higher level features provides several benefits: these can be more discriminative, which helps data association, and can serve to inform service robotic tasks that require higher level information, such as object models and poses. We propose an approach for online object discovery and object modeling, and extend a SLAM system to utilize these discovered and modeled objects as landmarks to help localize the robot in an online manner. Such landmarks are particularly useful for detecting loop closures in larger maps. In addition to the map, our system outputs a database of detected object models for use in future SLAM or service robotic tasks. Experimental results are presented to demonstrate the approach's ability to detect and model objects, as well as to improve SLAM results by detecting loop closures.**

I. Introduction

Service robots operating in human environments require a variety of perceptual capabilities, including localization, mapping, object discovery, recognition, and modeling. Each environment may have a unique set of objects, and a set of object models may not be available a-priori. Object discovery approaches have been proposed to address this need [15], [3] . Simultaneous Localization and Mapping (SLAM) is also required for service robot to be able to map new environments and navigate within them. We propose an approach to online object discovery and modeling, and show how to combine this with a SLAM system. The benefits are twofold: an object database is produced in addition to the map, and the detected objects are used as landmarks for SLAM, producing improved mapping results.

Localization and navigation are two basic problems in the area of mobile robotics. Map based navigation is a commonly used method to navigate from one point to another, where maps are commonly obtained using simultaneous localization and mapping (SLAM). Durrant-Whyte and Bailey provide a survey of the SLAM literature and the state of art in [8], [2]. Recent SLAM systems use graph optimization technique to jointly estimate the entire robot trajectory and landmarks using sparse optimization techniques [10], [7], [24].

However, traditional feature based maps are composed of low level primitives like points and lines which are mostly used to model space based on its geometric shape [5]. These maps lack the semantic information necessary for performing wider range of tasks, which may require higher level representation such as object models. Dense metric maps also have a lot of redundant information like 3D points in floor and uninformative textured surfaces which do not facilitate robot localization. In addition, dense maps built using depth cameras can be memory intensive as well.

The authors are with Institute for Robotics and Intelligent Machines, Georgia Institute of Technology. This work was financially supported by ARL MAST CTA, project 36666A9, and Boeing Corporation.

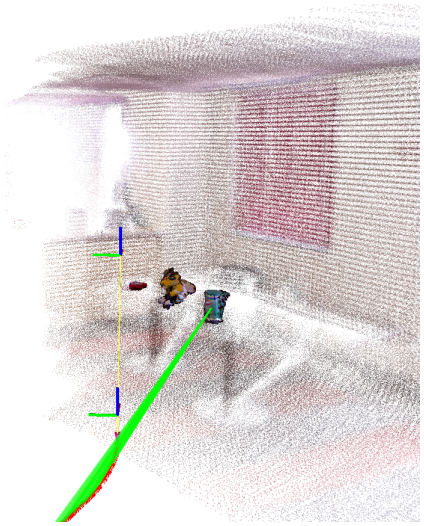


Fig. 1: Snapshot of the process at one instant. The robot trajectory is shown in red. Green lines add constraints between the object landmarks and the robot poses they are seen from. Light background shows the aggregated map cloud generated using the current SLAM solution.

In contrast, maps augmented with objects confer a number of advantages for mapping the environment. Features, e.g., objects and planes, can be represented using a compact representation and provide a richer description of the environment. It is simpler to include prior constraints at the object level than at the lower feature level. The semantic information available for an object provides better cues for data association as compared to a 3D point cloud. An object would not be represented by a 3D point but rather by a 3D point cloud. Joint optimization over all the camera poses and objects is computationally cheaper than the joint optimization over all the 3D points and cameras since, as there are many fewer objects compared to the number of 3D points in a map.

Hence, *semantic mapping* has gathered a lot of interest from the robotics community. Kuipers [18] modeled the environment as a spatial semantic hierarchy, where each level expresses states of partial knowledge corresponding to different level of representations. Ranganathan and Dellaert [22] presented a 3D generative model for representing places using objects. The object models are learned in a supervised manner. Nüchter and Hertzberg [20] described an approach to semantic mapping by creating a 3D point cloud map

of the environment and labeling points using the different semantic categories like floor, wall, ceiling or door. Pronobis et al. [21] proposed a complete and efficient representation of indoor spaces including semantic information. They use a multi-layered semantic mapping representation to combine information about the existence of objects in the environment with knowledge about the topology and semantic properties of space such as room size, shape and general appearance.

Some recent semantic mapping work has focused on using higher level landmarks such as objects. Rogers et al. recognize door signs and read their text labels such as room numbers, which are used as landmarks in SLAM [13]. Trevor et al. used planar surfaces corresponding to walls and tables as landmarks in a mapping system [27]. More recently, the SLAM++ system proposed by Salas-Moreno et al. [23] trained domain specific object detectors corresponding to repeated objects like tables and chairs. The learned detectors are integrated inside the SLAM framework to recognize and track those objects resulting in semantic map. Similarly Kim et al. [16] uses learned object models to reconstruct dense 3D models from single scan of the indoor scene.

Object discovery and scene understanding are also related to our approach. Karpathy et al. [15] decompose a scene into candidate segments and ranks them according to their objectness properties. Collet et al. used domain knowledge in the form of metadata and use it as constraints to generate object candidates [3]. Using RGBD sensor, Koppula et al. [17] used graphical models capturing various image feature and contextual relationship to semantically label the point cloud with object classes and used that on a mobile robot for finding objects in a large cluttered room. Hoiem and Savarese [11] did a survey of additional recent work in the area of 3D scene understanding and 3D object recognition.

All of these approaches either learn object models in advance, or discover them in a batch process after the robot has scanned the environment and generated a 3D model or a video sequence. In contrast, our main contribution is to integrate object discovery and mapping in a SLAM framework, following an online learning paradigm. Considering the advantages of object augmented maps, we too use objects as landmarks in addition to other features. However, we discover objects in an incremental fashion as the robot moves around in the environment. In contrast to other approaches, we do not train object detectors ahead of time, but we discover objects and train on object representation in an online manner. As the robot moves through the environment, it continuously segments the scene into non planar and planar segments (Section II-A). All the non planar segments associated across frames are considered as object hypotheses. Every new object is represented using the 3D descriptors and matched against the other objects in the given map (Section II-B). The transformation between the new object and the matched object is used as a constraint when optimizing the whole map. This is considered as a loop closure constraint when the robot sees the same object after some time and is used to optimize the complete trajectory (non sequential detection). It also helps in reducing the segmentation errors by adding a constraint between an under-segmented object part and the full object model, which results in better object modeling (recurring detection). When the new object does not match any of the previous objects, the object representation for the new object is saved for future use. Figure 1 shows a screenshot from our system that illustrates our object landmarks.

To demonstrate the effectiveness of our approach, we show results on the object discovered in the process and the resulting object augmented map generated by it (Figure 4). We also compare the robot trajectory estimated with and without the use of object landmarks during loop closure (Figure 7).

II. Our Approach

As the robot moves around in an indoor environment, we continuously map the environment using the SLAM system presented by Trevor et al. [27], [28]. ICP along with odometry information available during each frame is used to infer the robot pose trajectory. We use the Georgia Tech Smoothing and Mapping (GTSAM) library to optimize the robot poses and landmarks [6], [14]. In GTSAM, the SLAM problem is described as a factor graph where each factor represents a constraint between the variables (robot poses, objects, planes etc.). We add some additional factors in between the objects and pose-objects as we discover and recognize them. This is described in Section II-A and II-B. It is jointly optimized by performing maximum a-posteriori (MAP) inference over the factor graph.

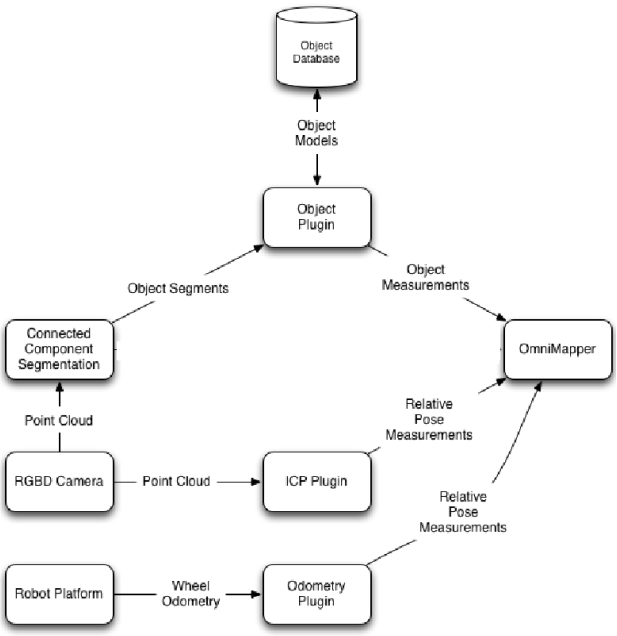


Fig. 2: Flowchart of the data processing pipeline.

Each RGBD frame is segmented using connected component segmentation to generate planes and non-planar segments [29]. We consider all the non-planar segments as the object segments. Each non planar segment from

each frame is propagated across frames to generate object hypothesis (Section II-A). The proposed object hypothesis $\mathbf{O}$ are represented using 3D feature descriptors augmented with map information (Section II-B). Object representation helps in identifying loop closure when the robot sees the same object and for producing better object models when an object part undersegmented in a few frames matches to the full object model. Object recognition is done by finding the object representation with the maximum number of inlier correspondences with the current object (Section II-B). If an object does not find the minimum number of inlier correspondences with any of the saved representations, it saves the representation of the current object. This follows an online learning framework, where the robot identifies potential objects and matches to other objects in the map, if none of them matches it hypothesizes that the potential object is a new object and therefore saves the representation. Figure 2 shows the flowchart of the complete system. Trevor et al. provide a detailed description of the mapping framework for integrating different sensor plugins [28].

In the following discussion we assume that the objects do not move during the experiment and there are few or no repetitive objects. We provide the details to our approach in the sections below.

A. Object Discovery

Segmentation is an important step used in many perception tasks like object detection and recognition. We use segmentation to extract objects of interest from each RGBD frame and propagate them across frames to find the set of segments resulting in one object.

1) Per Frame Segmentation: We assume that in indoor environments, objects are supported by homogeneous planes such as walls, floors, table tops. We consider all the homogeneous planes as planar landmarks and the remaining non-planar regions as the regions corresponding to potential objects of interest. To segment such scenes, we employ the organized connected component segmentation approach of Trevor et. al [29]. Given a point cloud, surface normals are computed for each point in each frame using the technique of Holzer et. al [12]. Using these normals, connected component corresponding to surfaces with smoothly varying normals are found. Least squares plane fit is used to recover planar segments having more than the minimum number of inliers and low curvature.

Points corresponding to non-planar regions are further segmented using different segmentation techniques. We experiment with connected component segmentation [29] and graph based segmentation [9] to segment out objects. We use connected component segmentation considering its better runtime performance and comparable accuracy to graph based segmentation. The remaining clustered regions are further filtered to exclude clusters that are either too small or too large, to exclude noise and large furniture. In this work, we consider only clusters that include at least 1000 points, and have a bounding box volume of less than 1 cubic meter. These values were determined empirically.

2) Segment Propagation: We use a graph matching strategy similar to the method proposed by Couprie et al. [4]. Our approach works as follows. During each frame we use the aggregated segmentation result until the previous frame S_{t-1} and the segmentation result from the current frame S_t to produce the final segmentation in the current frame S_t^*. This is done by matching segments aggregated until the previous frame with the segments in the current frame.

Since the robot is mapping the environment and estimating its trajectory using other sources of information like odometry and ICP algorithm, we have a prior estimate of the camera pose available at each frame. We transform the current frame according to the estimated pose. The segments from all the previous frames are transformed according to their respective camera poses and aggregated into one point cloud. The current transformed frame is then matched against the aggregated segments to propagate segment labels.

We represent each segment as the centroid of the point locations corresponding to that segment. In contrast to RGB images, depth is an important component of RGBD images. We found that centroid of a segment effectively represents that segment and using color based information does not improve result by much. For each segment centroid in the current frame, we find the nearest centroid in the aggregated segments. If the centroid in aggregated segments matches to more than one segment in the current frame, we choose the segment in the current frame which is the closest to the corresponding aggregated segment. This results in a *two-way matching* of segments. The segments rejected by this strategy are given new labels.

However, since we are matching against the whole map, a new segment in the current frame can match against a far away segment because we match each segment to its nearest segment centroid in the map. To avoid this, we perform an additional check by computing the bounding box $\mathbf{B_t}$ of the current segment and the bounding boxes of the matching segment $\{\mathbf{B_o}\}$ in the map. If the intersection of the two bounding boxes is small, we do not match the current segment to the corresponding segment in the map and instead give it a new label. The intersection is computed using Jaccard index which is given by Equation 1.

$$J(\mathbf{B_t}, \mathbf{B_o}) = \frac{|\mathbf{B_t} \cap \mathbf{B_o}|}{|\mathbf{B_t} \cup \mathbf{B_o}|} \tag{1}$$

Given the set of segments having the same label, we consider the object corresponding to the set of segments to be completely modeled if no new segment is added to the same label and the object is outside the view frustum of the camera. The object point cloud is created by aggregating the segments from all matched frames transformed according to their respective camera poses.

Once the object is modeled, non-linear constraints are added between the object landmarks and the corresponding robot poses. In the SLAM system, each object $\mathbf{O}$ is represented by the centroid of its point cloud $\mathbf{C} \in \mathbb{R}^3$ (in the map frame). Given a matching object segment S having the centroid $\mathbf{C_s} \in \mathbb{R}^3$ (in the robot frame) and the corresponding

robot pose $\mathbf{X} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$, the object measurement function $f(\mathbf{X}, \mathbf{O})$ is given by

$$f(\mathbf{X}, \mathbf{O}) = R\mathbf{C} + t \tag{2}$$

$f(\mathbf{X}, \mathbf{O})$ transforms the object centroid in the robot frame. Assuming a Gaussian noise λ with covariance matrix Λ, $\mathbf{C_s}$ is given by

$$\mathbf{C_s} = f(\mathbf{X}, \mathbf{O}) + \lambda \tag{3}$$

$$P(\mathbf{C_s}|\mathbf{X}, \mathbf{O}) \propto \exp -\frac{1}{2}\|f(\mathbf{X}, \mathbf{O}) - \mathbf{C_s}\|^2_{\mathbf{\Lambda}} \tag{4}$$

Constraints added according to the above measurement model jointly optimizes for the object location and the robot trajectory. The object recognition (Section II-B) considers all such objects given by the object discovery and adds constraints between the matching objects.

3) Object Refinement: In a separate thread, we merge different objects given the current robot trajectory and object location estimate. Each object represented using the CSHOT descriptor (Section II-B) [25] is matched against other nearby objects given the current estimated map. Each object is matched against the nearby objects whose Jaccard index (Equation 1) is non-zero when compared with the bounding box of the current object. The matching objects are merged to form one object. This step helps merging the under-segmented object parts not matched by segment propagation. We don't add object object constraint or perform optimization in this step and it only helps in improved object model generation given the current state of the SLAM solution.

B. Object Representation and Recognition

Each object point cloud given by the object discovery is uniformly sub-sampled to using a voxel grid with minimum leaf size equals to 3 cm to generate a set of keypoints $\mathbf{K}$. We found that using 3 cm as the minimum leaf size gave a proper balance between the accuracy and the computation time. The normals at each keypoint is estimated using a radius of 1 cm. Normal computation utilizes the full point cloud and not the sub-sampled cloud. Considering a good balance between the recognition accuracy and time complexity, we use CSHOT (or SHOTCOLOR) descriptor to describe each point corresponding to a segment [1], [25]. The resulting descriptor is denoted as $\mathbf{D}$.

All point coordinates corresponding to the object point cloud are represented in the global map frame, so that we can exploit the spatial context while matching a new object to the stored representations. We store the centroid (in the map frame) of the un-sampled point cloud $\mathbf{C}$. Each object $\mathbf{O}$ is represented using the keypoints $\mathbf{K}$, CSHOT descriptor $\mathbf{D}$ and the centroid $\mathbf{C}$ as shown in equation 5.

$$\mathbf{O} = \{\mathbf{K}, \mathbf{D}, \mathbf{C}\} \tag{5}$$

Every new object $\mathbf{O}_i$ estimated using object discovery (Section II-A) is matched to all the nearby objects from the current map. For every new object we compute the object representation $\{\mathbf{K}_i, \mathbf{D}_i, \mathbf{C}_i\}$. Since the object $\mathbf{O}_i$ is considered as a landmark, we can estimate its current location $\mathbf{C}_i$ and uncertainty in the estimate $\mathbf{\Sigma}_i$ given the current SLAM solution. We utilize this information $(\mathbf{C}_i, \mathbf{\Sigma}_i)$ to find out the set of potential objects $\mathbf{\Omega}$ that are likely to match to the new object. We consider all the objects $\mathbf{\Omega}$ whose centroid lie within twice the covariance radius $\mathbf{\Sigma}_i$ of the new object $\mathbf{O}_i$ as the set of potential matching objects. More formally, the set of potential objects ($\mathbf{\Omega}$) is given as:

$$\mathbf{\Omega} = \forall_{\omega \in \mathbf{\Theta}} \left\{ (\mathbf{C}_\omega - \mathbf{C}_i)^T \mathbf{\Sigma_i^{-1}} (\mathbf{C}_\omega - \mathbf{C}_i) < 2 \right\} \tag{6}$$

Here the ω and $\mathbf{C}_\omega$ represents a potential matching object and the corresponding centroid. $\mathbf{\Theta}$ represents the set of all objects. $\mathbf{C}_i$ and $\mathbf{\Sigma}_i$ represents the centroid of the new object and its covariance matrix, respectively.

The new object $\mathbf{O}_i$ is matched to a potential object $\omega \in \mathbf{\Omega}$ by finding the correspondences between keypoints $\mathbf{K}_i$ of the new object and the keypoints $\mathbf{K}_\omega$ of the potential object ω. The correspondences between $\mathbf{K}_i$ and $\mathbf{K}_\omega$ are estimated by finding the nearest neighbors in both directions resulting in a two-way match. First of all, we compute the nearest neighbors of feature descriptors $\mathbf{D}_i$ with respect to all the feature descriptors $\mathbf{D}_\omega$ belonging to the potential object ω. The same is done in reverse for all the descriptors $\mathbf{D}_\omega$ in the potential object with respect to $\mathbf{D}_i$. Let the nearest neighbor of a keypoint $k \in \mathbf{K}_i$ in ω is ζ. If the nearest neighbor of keypoint $\zeta \in \mathbf{K}_\omega$ in $\mathbf{O}_i$ is k, we accept that correspondence between the two objects (two-way match). Those correspondences are then further refined using geometric verification. If the number of refined correspondences is less than 12, the object is not considered as a match to the new object. We found that using 12 reduces the number of false positive matches. Among all the matching object representations, we find the object $\mathbf{O}^*$ which has the maximum number of inlier correspondences with respect to the new object $\mathbf{O}_i$.

Assuming that most of the objects don't move during the mapping time, we use the spatial context stored with the object to make the search more efficient. The representations are searched in the order of geometric distance $\|\mathbf{C}_i - \mathbf{C}_\omega\|$ from the new object location. In case the new object does not match to any of the stored representations, we assume that it is an unseen object and save its representation to disk.

In case we find a match to the new object, the two major use cases are as follows:

- If the matching object is seen after a certain period of time when the robot returns to the same location, we declare it as *loop closure detection*.
- If the matching object is one of the under-segmented object parts, we add constraint between the under-segmented part and the new object in order to merge them as the optimization progresses. It is used by *object refinement* (Section II-A.3) thread to generate a refined model.

Given the new object $\mathbf{O}_i$, the matching object $\mathbf{O}^*$, the centroid of the new object $\mathbf{C}_i$ and the centroid of the matching object $\mathbf{C}^*$, we add a non linear constraint between the new object and the matching object.

The object-object measurement function $h(\mathbf{O}, \mathbf{O}^*)$ is given by,

$$h(\mathbf{O}_i, \mathbf{O}^*) = \mathbf{C}^* \tag{7}$$

The ideal distance between $\mathbf{C}^*$ and $\mathbf{C}_i$ should be zero since both the centroids belong to the same object separated apart due to the error incurred by SLAM or object discovery. Assuming Gaussian measurement noise γ with covariance matrix Γ,

$$\mathbf{C}_i = h(\mathbf{O}_i, \mathbf{O}^*) + \gamma \tag{8}$$

$$P(\mathbf{C}_i|\mathbf{O}_i, \mathbf{O}^*) \propto \exp -\frac{1}{2}\|h(\mathbf{O}_i, \mathbf{O}^*) - \mathbf{C}_i\|^2_\Gamma \tag{9}$$

In both the cases, loop closure detection or object refinement, the non linear constraint tries to minimize the gap between matching object and the new object. Minimizing the distance between the new object and the matching object gives better object models and at the same time optimizes the robot trajectory.

III. Experimental Results

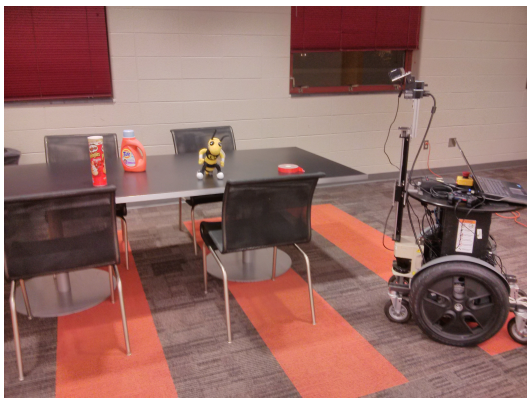


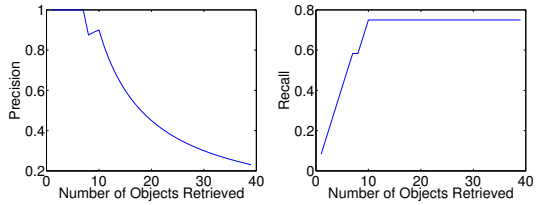
Fig. 3: The robot and a snapshot of the scene where the experiment was conducted.

A. Robot Platform

The robot platform used in this work consists of a Segway RMP-200 mobile base, which has been modified to be statically stable. It is equipped with a SICK LMS-291 for obstacle avoidance and navigation, although not used for this work. Kinect depth camera is used to collect point cloud information. Computation is performed on an on board laptop; however, our experiments are run on desktop machines using log data gathered from the robot. To evaluate our system, we collected data from the Georgia Tech Institute for Robotics and Intelligent Machines. The floor plan of the institute is shown in Figure 7c and 8c. The robot and a snapshot of the scene is shown in Figure 3.

B. Object Discovery

In an experiment, the robot is tele-operated along a trajectory shown in Figure 4a to test the object modeling capabilities. Twelve objects were used in this experiment. Figure 4b shows one particular frame from the trajectory



(a) Precision (b) Recall

Fig. 6: Precision and Recall curves of Object Discovery

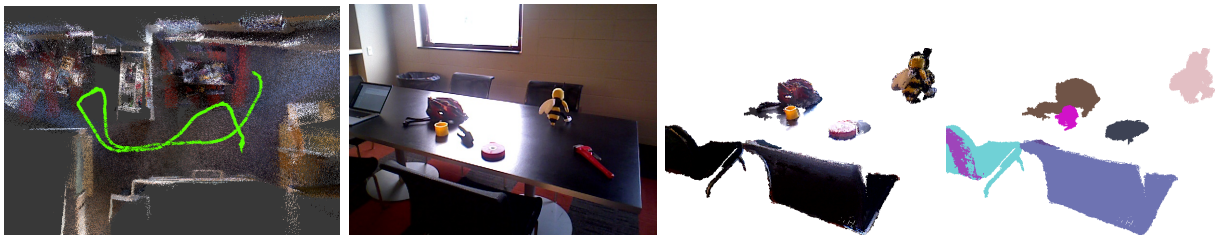
	#Objects	#True Positive Objects Discovered	Percentage Reduction in #Objects Discovered after Object Refinement
1	12	9	30%
2	5	4	27.28%
3	5	4	24.56%

TABLE I: Number of true positive objects discovered and the effect of object refinement for merging under-segmented objects.

which shows the objects kept on the table. Figure 4c shows the corresponding object segments estimated by segmentation propagation (Section II-A.2). Figure 4d shows the same frame with object labels given to discovered objects after object refinement (Section II-A.3). Each color represents a different object. The objects labels are well defined with each label representing a different object. Labeling error in the back of the left chair is due to per-frame segmentation error which gives it a new label. However the segmentation errors are not propagated across many frames.

Figure 5 shows the top four objects discovered sorted by number of frames it is seen in. Figure 5a shows the image of the discovered objects and figure 5b shows the corresponding reconstructed model. We found that the noisy objects detected due to segmentation failures are not propagated across many frame and the quality of the discovered object is directly proportional to the number of frames it is seen in. This is analogous to the 3D point where a point visible from many images has a well defined location and it is more likely to be seen in a new image. Similarly an object seen across many frames is better modeled and it is more likely to recognize this object in a new frame. Assuming 12 objects, Figure 6 shows the precision and recall curves representing the quality of the retrieved objects when retrieved according to the number of frames they are seen in. As we can see most of the true positive objects are seen across many frames. In general, storing top 15 objects includes all the true positive discovered objects.

Table I shows the results of object discovery and the effect of object refinement for merging under-segmented objects. Row one represents the object discovery experiment where the robot is tele-operated along a trajectory shown in Figure 4a. The remaining rows correspond to the loop closure experiments described in Section III-D. As we can see the from the results, the object discovery pipeline is able to discover most of the objects used in the experiments. The number of discovered objects on an average is reduced by



(a) (b) (c) (d)

Fig. 4: Various stages of the object discovery pipeline. (a) The map of the scene and the trajectory along which the robot is tele-operated. (b) One particular frame from the trajectory showing the objects kept on the table. (c) The corresponding object segments estimated by segmentation propagation (Section II-A.2). (d) The same frame with labels given to discovered objects after object refinement (Section II-A.3). Each color represents a different object.

(a)

(b)

Fig. 5: The top four objects discovered sorted in an descending order of the number of frames it is seen in. (a) Snapshot of the discovered objects. (b) The corresponding reconstructed object models.

25% on using object refinement which merges the under-segmented object parts.

C. Object Recognition

Luis A. Alexandre evaluated different 3D descriptors for object and category recognition and showed that CSHOT (or SHOTCOLOR) worked the best considering its recognition accuracy and time complexity [1], [25]. CSHOT has an object recognition accuracy of 75.53% [1]. While doing object recognition we only consider the objects which lie within twice the uncertainty radius of an object (Equation 6). It helps in reducing the false positive matches. Other than this we assume that the objects do not move during the experiment and are non-repetitive (except chairs). All these increase the recognition performance. We do not explicitly remove the false positive data association and assume it to be unlikely considering the uncertainty constraint and static, non-repetitive assumption.

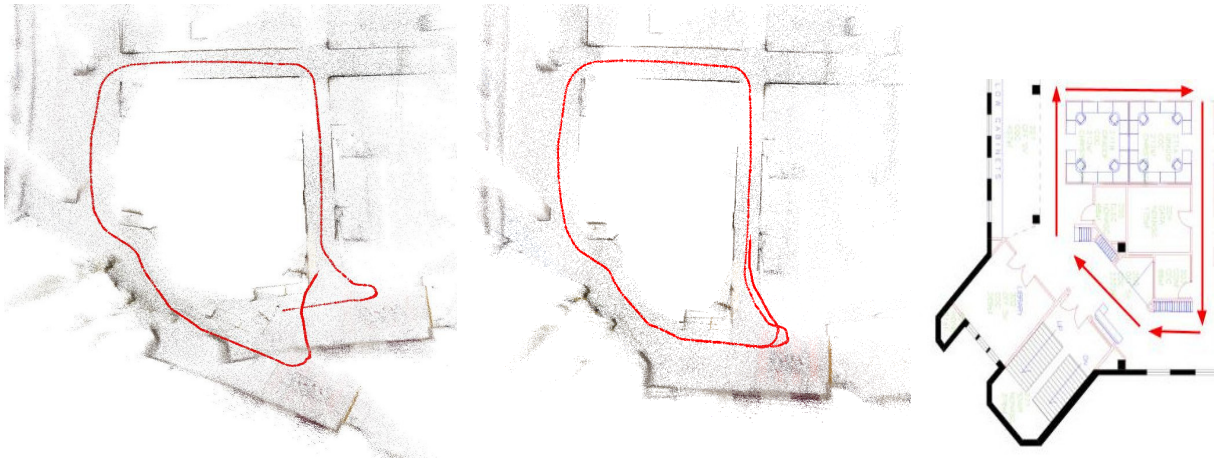
D. Loop closure detection

To test the loop closing capabilities, the robot is tele-operated in the cubicle area forming a loop as shown in Figure 7c and 8c. In the first experiment, the robot moves through the atrium twice with objects kept on the table. Objects kept on the table are used as landmarks when closing the loop. Figure 7c shows the approximate trajectory on which the robot is run.

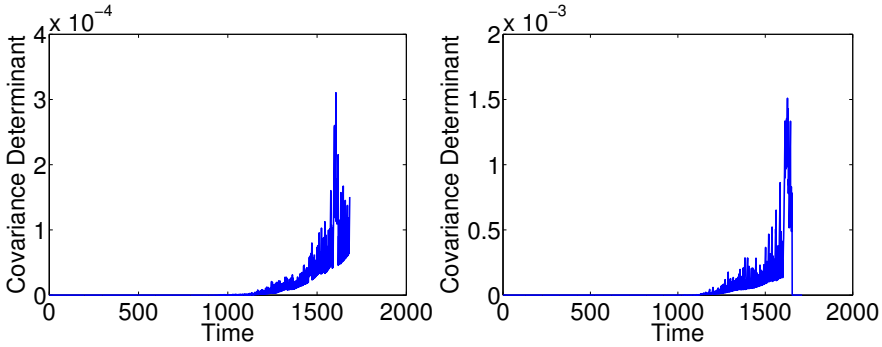
In Figure 7a we see the robot trajectory estimated using only odometry and ICP. Object landmarks are not used. There is an error in the bottom right section when the robot returns to the same location. Loop closures fails to be detected with ICP alone. Figure 7b shows the robot trajectory estimated when object landmarks are used for loop closure. The object-object constraints joins the bottom right location when loop closure is detected. Figure 7d shows the covariance determinant of the latest pose when using objects for loop closure as compared to not using objects. In the left plot, we see that the covariance determinant keeps on rising where as the in the right plot, we see a sudden fall in the value of covariance determinant representing a loop closure. The actual values of the determinant depends on the noise model used.

In the second experiment, the robot is tele-operated for a longer time period to form longer length trajectory. Figure 8c shows the approximate trajectory of this run.

In Figure 8a we see the robot trajectory estimated when using no object landmarks. There is an error in the estimated

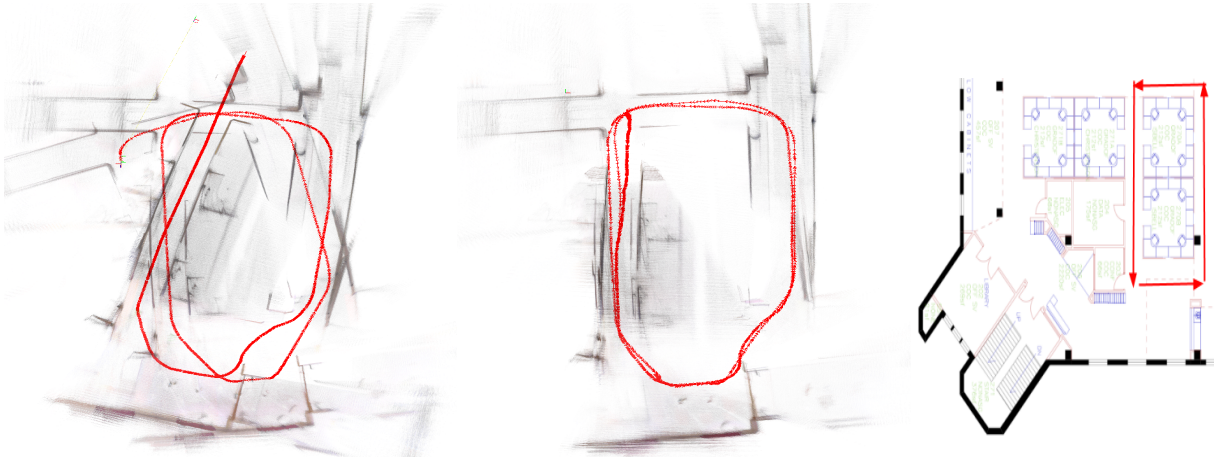


(a) (b) (c)



(d)

Fig. 7: Trajectories estimated with and without using objects as landmarks during loop closure. (a) Robot trajectory estimated when using no object landmarks. There is an error in the bottom right section when the robot returns to the same location. (b) Robot trajectory estimated when using object landmarks for loop closure. The object-object constraints joins the bottom right location when loop closure is detected. (c) Shows the approximate ground truth trajectory w.r.t the floor plan. (d) Covariance determinant of the latest pose w.r.t time when not using objects (left) as compared to when using objects for loop closure (right). There is a sudden fall in the value of covariance determinant due to a loop closure event.



(a) (b) (c)

Fig. 8: Trajectories estimated with and without using objects as landmarks during loop closure. (a) Robot trajectory estimated when using no object landmarks. (b) Robot trajectory estimated when using object landmarks for loop closure. (c) Shows the approximate ground truth trajectory w.r.t the floor plan.

trajectory due to the error in odometry and ICP estimates propagated across the full trajectory. Figure 8b shows the robot trajectory estimated when using object landmarks for loop closure. Object-object loop closure constraint is able to reduce the error in the trajectory estimate.

IV. Conclusions and Future Work

In this paper, we proposed an approach to simultaneously discover objects and produce their models along with the map. Objects provide a richer description of the environment and can be more effective for data association. We use the discovered objects as landmarks and the data association among them assist with the loop closure in large environments. The pipeline follows an online learning framework to avoid any pre-training on the object models.

Our approach is able to discover good quality object models in an unsupervised manner and use those objects as landmarks during loop closure. We showed the results of object discovery (Table I), the discovered models (Figure 4) and compared the resulting models with the actual snapshot of the objects (Figure 5). We also showed that using object-object constraints during loop closure estimated a better trajectory as compared to the trajectory estimated when not using the object-object constraints. The comparison of the robot trajectory estimated with and without the use of object landmarks is shown in Figures 7 and 8.

However in the existing pipeline, we cannot handle moving or repetitive objects. We plan to integrate JCBB to handle false positive matches caused by moving and repetitive objects [19]. Moving objects can also be handled using the expectation maximization technique as proposed by Rogers et al. which allows a landmark to be dynamic [13]. Better object discovery techniques can be used to detect repetitive objects and cluster them [3]. Repetitive or moving objects when detected can only be used to coarsely localize a robot depending on the object's mobility region. For example, objects like a kitchen utensil is generally found in a kitchen and can be used to infer the robot's coarse location inside the house but it cannot be used to precisely estimate its pose. As a future work, we plan to include a variable representing the object's mobility in order to achieve robust localization in the case of moving objects.

We also plan to add interactive object modeling and labeling system for the objects where a user can add semantic information about the discovered objects like the actions which can be performed on this object [26].

References

[1] L. A. Alexandre. 3D descriptors for object and category recognition: a comparative evaluation. In *Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vilamoura, Portugal, October 2012.

[2] T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (SLAM): part II. *IEEE Robotics Automation Magazine*, 13(3):108–117, 2006.

[3] A. Collet Romea, B. Xiong, C. Gurau, M. Hebert, and S. Srinivasa. Exploiting domain knowledge for object discovery. In IEEE, editor, *IEEE International Conference on Robotics and Automation (ICRA)*, May 2013.

[4] C. Couprie, C. Farabet, Y. LeCun, and L. Najman. Causal graph-based video segmentation. In *IEEE Internatinal Conference on Image Processing (ICIP)*, 2013.

[5] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007.

[6] F. Dellaert. Factor Graphs and GTSAM: A Hands-on Introduction. Technical Report GT-RIM-CP&R-2012-002, GT RIM, Sept 2012.

[7] F. Dellaert and M. Kaess. Square root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing. *Int. J. Rob. Res.*, 25(12):1181–1203, Dec. 2006.

[8] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: Part I. *IEEE Robotics Automation Magazine*, 13(2), 2006.

[9] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004.

[10] J. Folkesson and H. I. Christensen. Graphical SLAM - a self-correcting map. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 383–389, New Orleans, Apr. 2004. IEEE.

[11] D. Hoiem and S. Savarese. *Representations and Techniques for 3D Object Recognition and Scene Interpretation*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2011.

[12] S. Holzer, R. B. Rusu, M. Dixon, S. Gedikli, and N. Navab. Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2684–2689, 2012.

[13] J. G. R. III, A. J. B. Trevor, C. Nieto-Granda, and H. I. Christensen. Simultaneous localization and mapping with learned object recognition and semantic data association. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1264–1270, 2011.

[14] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental Smoothing and Mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008.

[15] A. Karpathy, S. Miller, and L. Fei-Fei. Object discovery in 3d scenes via shape analysis. In IEEE, editor, *IEEE International Conference on Robotics and Automation (ICRA)*, May 2013.

[16] Y. M. Kim, N. J. Mitra, D.-M. Yan, and L. J. Guibas. Acquiring 3d indoor environments with variability and repetition. *ACM Trans. Graph.*, 31(6):138, 2012.

[17] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena. Semantic labeling of 3d point clouds for indoor scenes. In *NIPS*, pages 244–252, 2011.

[18] B. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119:191 – 233, 2000.

[19] J. Neira and J. Tardos. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, Dec 2001.

[20] A. Nüchter and J. Hertzberg. Towards semantic maps for mobile robots. *Robot. Auton. Syst.*, 56(11):915–926, Nov. 2008.

[21] A. Pronobis and P. Jensfelt. Large-scale semantic mapping and reasoning with heterogeneous modalities. In *IEEE International Conference on Robotics and Automation (ICRA)*, may 2012.

[22] A. Ranganathan and F. Dellaert. Semantic modeling of places using objects. In *Robotics: Science and Systems*, 2007.

[23] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. SLAM++: Simultaneous localisation and mapping at the level of objects. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[24] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[25] F. Tombari, S. Salti, and L. Di Stefano. A combined texture-shape descriptor for enhanced 3d feature matching. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 809–812, 2011.

[26] A. Trevor, I. Rogers, J.G., A. Cosgun, and H. Christensen. Interactive object modeling & labeling for service robots. In *08th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 421–421, 2013.

[27] A. Trevor, J. Rogers, and H. Christensen. Planar surface SLAM with 3D and 2D sensors. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3041–3048, 2012.

[28] A. Trevor, J. Rogers, and H. Christensen. Omnimapper: A modular multimodal mapping framework. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[29] A. J. B. Trevor, S. Gedikli, R. B. Rusu, and H. I. Christensen. Efficient organized point cloud segmentation with connected components. In *Semantic Perception Mapping and Exploration (SPME)*, May 2013.