Semantic SLAM with Autonomous Object-Level Data Association

Zhentian Qian, Kartik Patath, Jie Fu, Jing Xiao

***Abstract*— It is often desirable to capture and map semantic information of an environment during simultaneous localization and mapping (SLAM). Such semantic information can enable a robot to better distinguish places with similar low-level geometric and visual features and perform high-level tasks that use semantic information about objects to be manipulated and environments to be navigated. While semantic SLAM has gained increasing attention, there is little research on semantic-level data association based on semantic objects, i.e., object-level data association. In this paper, we propose a novel object-level data association algorithm based on bag of words algorithm [1], formulated as a maximum weighted bipartite matching problem. With object-level data association solved, we develop a quadratic-programming-based semantic object initialization scheme using dual quadric and introduce additional constraints to improve the success rate of object initialization. The integrated semantic-level SLAM system can achieve high-accuracy object-level data association and real-time semantic mapping as demonstrated in the experiments. The online semantic map building and semantic-level localization capabilities facilitate semantic-level mapping and task planning in a priori unknown environment.**

I. Introduction

In order for robots to interact intelligently with the real world and perform high level tasks, semantic information of its surroundings must be acquired. Traditional visual or visual-inertial simultaneous localization and mapping (SLAM) algorithms extract low-level geometric features such as corners, lines and surface patches from image sequence to build sparse or dense point cloud map. However, low-level geometric map is insufficient for more sophisticated tasks, such as getting a book from a particular desk or carrying meal to a particular nightstand for a patient. It is often necessary to add semantic information into the map, which motivates the need for semantic SLAM. While semantic SLAM has gained increasing attention in the literature (see Related Work), there is little research on semantic-level data association based on semantic objects. This paper introduces an integrated system for performing semantic-level SLAM, addressing the key problem of object-level data association. This endeavour is our first step towards a visual semantic SLAM algorithm which would tightly integrate geometric and semantic information.

The major contributions of our paper are the following:

- an autonomous object-level data association algorithm utilizing both geometric and appearance information of the object;
- a novel object initialization scheme improving the success rate of object initialization;
- an integrated, keyframe-based, real-time semantic SLAM system without the use of pre-built object database. In other words, there is no need to survey the environment to build a database of object shape, size and appearance before running our algorithm.

The authors are with the Department of Robotics Engineering, Worcester Polytechnic Institute, Worcester, MA, USA, 01609 e-mail: zqian, kpatath, jfu2, jxiao2@wpi.edu

A. Related Work

In order to bridge the gap between perception and action, the robotics community has taken a keen interest in semantic SLAM. The pioneering work of SLAM++ [2] performs object-level SLAM using a depth camera. The main restriction being that an object database of both 3D shape and global description must be built in advance. Improving on [2], the work in [3] relies solely on monocular input and learns the scale of the map from object models. Nevertheless, the same restriction remains as a pre-built object database is still required. Refs. [4], [5] resort to novel soft data association which in turn circumvents the need of having to assign incoming detected object to objects spawned in map. The drawback is that the object is still represented by 3D points instead of 3D shape. This representation limits the kind of interaction the robot can have with the physical world. Grasp operation, for example, may not be feasible with only point representation.

To address this issue of object representation, one method [6] used dual quadrics to capture the shape and pose of an object and developed a tailored SLAM backend for graph optimization. Dual quadric is a mathematically elegant solution as it can be compactly defined by nine continuous parameters and is also used for object representation in this paper. However, [6] leaves the key problem of object-level data association unsolved. Another method [7] integrates object detection module and RGB-D SLAM and further converts semantically augmented 3D point clouds to Octomap, which makes advanced missions such as grasp point selection possible. Although techniques including multi-threads processing are employed to speed up Octomap creation, their results generally take $50 \sim 200s$ to build the Octomap. Thus, this method falls short in real-time performance.

There are also works that directly add more semantics to the map generated from SLAM. The work in [8] uses a SLAM system to provide correspondences from 2D frame into a 3D map. These correspondences allow the semantic prediction of a Convolutional Neural Network (CNN) from multiple viewpoints to be probabilistically fused into the map. However, the semantic predictions are only loosely added to the map and do not aid the task of SLAM. Similarly,

the work in [9] also fuses 2D semantic segmentation and sparse 3D points from SLAM. This work is closely related to 3D reconstruction in computer vision as the 3D scene is represented in the form of voxels enhanced with semantic labels. Though the time for 3D reconstruction is not given in the paper, considering the size of the voxel representation and the time to perform optimization, real time performance is almost impossible.

II. System Overview

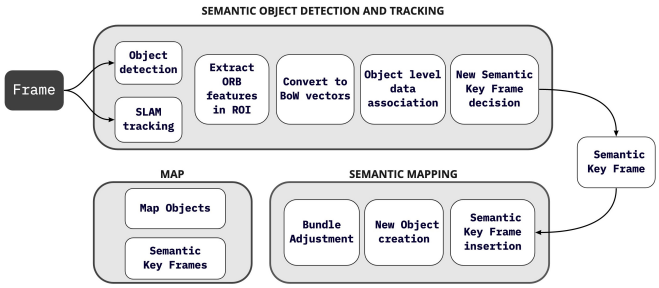

Fig. 1. System overview, showing all the steps performed in semantic object detection and tracking thread and semantic mapping thread. The main components of the map are also shown.

Figure 1 presents an overview of our system composed of two threads: semantic object detection and tracking (Sec. III) and semantic mapping (Sec. IV). The utilized SLAM tracking module is inherited from ORB-SLAM2 [10]. The map contains a set of map objects and semantic keyframes, as well as the correspondence between them. A semantic keyframe stores the map objects it has observed in the frame image. At the same time, each map object records the sequence of semantic keyframes in which this object has been observed. Similar to [6], our semantic SLAM employs semantic objects as landmarks and represents them as dual quadrics. The attributes of map objects and semantic keyframes are described in the Appendix. Next, we provide more details about the algorithms used in these two threads.

III. Semantic Object Detection and Tracking

In this section, we describe the semantic object detection and tracking thread. The key component of this thread is object-level data association—an essential step for robust semantic SLAM. Before we could perform data association, we first perform object detection and extract ORB features in the region of interest (ROI) where objects are detected. These features are subsequently converted to Bag of Words (BoW) [1] vectors to describe the detected objects. Finally, the data association step is performed to match the semantic measurements generated by object detector to map objects.

A. Object Detection and Conversion to BoW vectors

Our algorithm uses a ROS implementation [11] of YOLOv3 [12] as real-time object detector. For every processed RGB or grey image I_t, YOLOv3 outputs a set of semantic measurements $\mathcal{S}_t = \{z_k\}$, where each semantic measurement $z_k = \{b_k, c_k, s_k\}$ is a collection of three attributes, i.e., bounding box b_k, object class c_k, and detection score s_k. We discard semantic measurements with bounding boxes smaller than a certain threshold as we deemed them to hold too little image content to be robustly tracked.

The bag of words algorithm [1] converts the 2D image content inside a bounding box into a 1D BoW vector describing the appearance of the detected objects. The BoW vectors are compact to store and easy to compare. Each entry of the BoW vector is a weighted occurrence count of a particular visual word, i.e., a discretized ORB descriptor space. ORB features are required in the conversion. For each ROI bounded by a bounding box, FAST corners are extracted at a consistent density and the number of corners extracted per ROI varies according to the area of the ROI. For ROI that is textureless or has low contrast and holds few corners, its corresponding semantic measurement and corners extracted in this ROI are discarded. The ORB descriptors are then computed for the remaining FAST corners.

In the next step, we employ visual vocabulary [1], which is the discretization of the descriptor space, to convert the ORB features extracted in each bounding box to BoW vectors. It is structured as a tree with binary nodes which are created by k-medians clustering of training ORB descriptors. The leaves of the tree are the words of the visual vocabulary, weighted with the term frequency - inverse document frequency (tf-idf) according to their relevance in the training corpus. Words with fewer occurrences in the training images are deemed more discriminative and given a higher weight. Instead of using a single vocabulary, we create vocabularies for every object class in our implementation. Our intention is to ensure that each vocabulary is uniquely suited to distinguish the image content in bounding boxes of a particular object class. Each vocabulary is built offline with the ORB descriptors extracted within the ground truth bounding boxes from the COCO dataset images [13]. With $k = 5$ branches and $L = 5$ depth levels, each vocabulary has 3125 words.

When ORB features are extracted from a bounding box of certain object class, we use the matching visual vocabulary to convert them into a BoW vector. The conversion process is described as follows: For all the given ORB features, their descriptor vectors traverse the vocabulary tree from the root to the leaves, selecting at each level the nodes which minimizes the Hamming distance [14]. The BoW vector is simply the weighted occurrence counts of the words, i.e., leaves. The BoW vector provides a way for us to quantify the similarity between the semantic measurements and the map objects based on their appearance.

B. Object-Level Data Association

We propose an object-level data association scheme to match the incoming semantic measurements from YOLOv3 to registered objects in the map. Our method is a two-step frame-to-map matching, described as follows.

The first step is to determine the set of matching object candidates $\mathcal{T}^{ca}$ for a semantic measurement $z_k \in \mathcal{S}_t$. Each map object τ_j contains its class label. For a semantic measurement $z_k = \{b_k, c_k, s_k\}$, only objects with class label c_k are considered. Apart from the requirement on object

class, we introduce geometric checks to rule out objects with similar appearances but incorrect locations. Consider two cases:

Case 1: The map object τ_j has quadric representation. In this case, the projection of the object quadric center should reside inside the bounding box of the semantic measurement z_k, as shown on the right frame of Fig. 2.

Case 2: The map object τ_j does not have quadric representation. In this case, we assume that the 3D point corresponding to the center of the bounding box of the latest observation of τ_j is projected inside the bounding box of z_k. The assumption should hold if the latest observation of τ_j and z_k are not taken from drastically different viewpoint. Based on this assumption, the corresponding epipolar line should go through the bounding box of z_k. The epipolar line is represented by the real line on the right frame of Fig. 2.

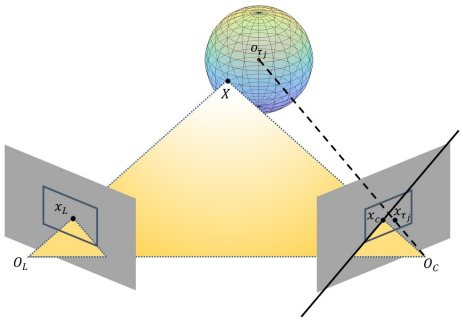

Fig. 2. The geometric requirements imposed on matching object candidates

Once we have determined the set of matching object candidates $\mathcal{T}^{ca}$, we proceed to the second step of data association. This step performs data association based on object appearance. A data association score is calculated between the semantic measurement and every potential matching object. Note that the assignment of semantic measurements to map objects is coupled, since the assignment of one semantic measurement to a particular map object means that map object can no longer be considered as candidate for other measurements. Our system solves the assignment of all semantic measurements by trying to maximize the sum of data association score. The details are given below.

For every map object $\tau_j \in \mathcal{T}^{ca}$, our system goes though semantic keyframes $K_i \in \mathcal{K}^j$, where $\mathcal{K}^j$ stands for the set of semantic keyframes that share observation of the same object τ_j. A L_1-score $s(\mathbf{v}_1, \mathbf{v}_2) = 1 - 0.5\left|\mathbf{v}_1/|\mathbf{v}_1| - \mathbf{v}_2/|\mathbf{v}_2|\right|$ is calculated between the BoW vector stored in the semantic keyframe K_i and the BoW vector corresponding to the bounding box of the semantic measurement z_k. The maximum score among $\mathcal{K}^j$ is considered the data association score c_{kj} between the semantic measurement z_k and the map object τ_j. Next, we formulate the object-level data association as a maximum weighted bipartite matching problem (assignment problem). First, we introduce a set of Boolean decision variables: For each semantic measurement z_k and a map object τ_j, let

$$x_{kj} = \begin{cases} 1, & \text{if } z_k \text{ is assigned to } \tau_j, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

And the assignment problem is formulated as an integer-programming problem as follows:

$$\underset{x_{kj}}{\operatorname{argmax}} \sum_{\{k:z_k \in \mathcal{S}_t^c\}} \sum_{\{j:\tau_j \in \mathcal{T}^{ca}\}} c_{kj} x_{kj} \tag{2}$$

$$\text{s.t.} \sum_{\{j:\tau_j \in \mathcal{T}^{ca}\}} x_{kj} \leq 1, \quad \sum_{\{k:z_k \in \mathcal{S}_t^c\}} x_{kj} \leq 1. \tag{3}$$

where $\mathcal{S}_t^c$ is the set of semantic measurements taken on image I_t with class c. The constraints (3) mean that (a) each semantic measurement can only be assigned to at most one map object and some semantic measurements cannot be assigned to map objects because they are either from new objects that have not yet been observed or due to false detections; and (b) each map object could only be associated with at most one semantic measurement.

The problem defined in (2) can be solved using a cost-scaling push-relabel algorithm [15], [16], [17], readily implemented in [18]. This algorithm has a time complexity of $\mathcal{O}(\sqrt{n}m\log(nC))$, where n and m are the number of nodes and edges in the bipartite graph, C is the largest edge cost (all edge costs need to be converted to integers).

Remark. YOLOv3 could sometimes assign different class labels to the same object in some corner cases. Performing data association only considering objects with the same class label would subsequently lead to wrong matches. Future work would calculate a matching score between the class of the map objects and the class of the semantic measurement instead of simply ruling out map objects of different classes.

IV. Semantic Mapping

In this section, we describe the semantic mapping thread, which is performed on every new semantic keyframe K_i. The semantic keyframe is generated by the semantic object detection and tracking thread every T image frames. When a new semantic keyframe is added, we update the map database to include this keyframe as well as the relations between map objects and semantic keyframes.

A. New Map Object Creation and Initialization

In this section, we propose a novel object initialization scheme. It is observed that sometimes the object initialized based on [6] is behind or intersects with the camera principal plane. We address this problem with a new object initialization scheme to increase the success rate of initialization.

Unlike in geometric SLAM where a map point is created and initialized at the same time, in our approach, a new object τ_j is created and registered in the map if it is observed the first time in the current semantic keyframe. Upon creation, the class of τ_j is determined, the current semantic keyframe observing τ_j is stored, and the number of observations is set to one. However, some attributes of τ_j cannot be initialized by one observation, i.e., the shape,

rotation, and translation (refer to Appendix). Only after a sufficient number of observations have been made on τ_j, its initialization procedure is performed. The initial shape and pose for the quadric representation of τ_j is calculated from semantic measurements z_k stored in a sequence of semantic keyframes $K_i \in \mathcal{K}^j$.

A quadric, in its dual form, is represented by the locus of all planes tangent to the quadric. A tangent plane Π satisfies:

$$\Pi^T \mathbf{Q}^* \Pi = 0 \tag{4}$$

where $\mathbf{Q}^*$ is a 4×4 symmetric matrix defining a quadric and only defined up to scale, and Π is a 4×1 vector corresponding to the coefficients of general form of the plane equation. Note that coefficients of plane equations are the parametrization of all planes mentioned below.

We can represent a generic dual quadric with a 9-vector $\hat{q} = [\hat{q}_1, \cdots, \hat{q}_9]^T$ where each element corresponds to one of the first nine independent elements of symmetric matrix $\mathbf{Q}^*$. The last element of $\mathbf{Q}^*$ is set to -1 to define the scale of $\mathbf{Q}^*$ to be 1. The quadric representation of map object τ_j can be initialized with quadratic programming to fit its defining equation (4). Expanding (4) leads to the following linear equation:

$$\begin{aligned}\big[\Pi(1)^2, 2\Pi(1)\Pi(2), 2\Pi(1)\Pi(3), 2\Pi(1)\Pi(4), \Pi(2)^2,\\ 2\Pi(2)\Pi(3), 2\Pi(2)\Pi(4), \Pi(3)^2, 2\Pi(3)\Pi(4), \Pi(4)^2\big]\\ \cdot [\hat{q}_1, \hat{q}_2, \hat{q}_3, \hat{q}_4, \hat{q}_5, \hat{q}_6, \hat{q}_7, \hat{q}_8, \hat{q}_9, -1]^T = 0.\end{aligned} \tag{5}$$

For each semantic measurement z_k taken on semantic keyframe $K_i \subset \mathcal{K}^j$ and associated with τ_j, we can generate four tangent planes to the quadric of the object τ_j based on its bounding box:

$$\begin{aligned}\Pi^T_{k,xmin} = [1, 0, -x_{k,min}]\mathbf{P}_i,\ \Pi^T_{k,xmax} = [1, 0, -x_{k,max}]\mathbf{P}_i\\ \Pi^T_{k,ymin} = [0, 1, -y_{k,min}]\mathbf{P}_i,\ \Pi^T_{k,ymax} = [0, 1, -y_{k,max}]\mathbf{P}_i.\end{aligned} \tag{6}$$

Here the camera projection matrix $\mathbf{P}_i$ is calculated from the camera pose $\mathbf{T}^i_w$ recorded in K_i. The camera pose $\mathbf{T}^i_w$ is initialized from the odometry measurements.

By collecting all tangent planes generated from the semantic measurements that are associated to object τ_j and substitute them into (5), we obtain a linear system of the form $\mathbf{A}_j \begin{bmatrix} \hat{q} \\ -1 \end{bmatrix} = 0$ with $\mathbf{A}_j$ containing the coefficients of all tangent planes to the quadric of map object τ_j as in (5). For the quadratic programming based initialization process, the objective function is the sum of square errors of the residual on the left hand side of (5). Formally,

$$\min_{\hat{q}} \frac{1}{2}\hat{q}^T \mathbf{H} \hat{q} + f^T \hat{q}. \tag{7}$$

Let

$$\mathbf{B} = \mathbf{A}_j^T \mathbf{A}_j = \begin{bmatrix} \mathbf{B}_{99} & \mathbf{B}_{91} \\ \mathbf{B}_{19} & \mathbf{B}_{11} \end{bmatrix} \tag{8}$$

where $\mathbf{B}_{ij}$ is a submatrix of i rows and j columns. Then $\mathbf{H} = \mathbf{B}_{99}$ and $f = \mathbf{B}_{91}$.

To improve the success rate of object initialization, we propose to include new constraints preventing objects from being incorrectly initialized. The first set of constraints require that the initialized quadric be in front of the camera:

$$(o_{\tau_j} - o_{K_i}) \cdot z_{K_i} \geq 0, \forall K_i \in \mathcal{K}^j \tag{9}$$

where o_{K_i} is the camera center of semantic keyframe K_i, z_{K_i} is the optical axis of the camera of K_i, o_{τ_j} is the quadric center of object τ_j and $o_{\tau_j} = -\begin{bmatrix}\hat{q}_4 & \hat{q}_7 & \hat{q}_9\end{bmatrix}^T$. Both o_{K_i} and z_{K_i} can be extracted from the camera pose $\mathbf{T}^i_w$ of K_i.

The second set of constraints specify that the principal plane of the camera should not intersect with the quadric:

$$\Pi^T_{K_i} \mathbf{Q}^*_{(\hat{q}_j)} \Pi_{K_i} \leq 0, \forall K_i \in \mathcal{K}^j \tag{10}$$

where $\Pi_{K_i} = \begin{bmatrix} z_{K_i} \\ -z_{K_i} \cdot o_{K_i} \end{bmatrix}$ is the principal plane of K_i. It is not to be confused with the tangent planes defined in (6).

The third set of constraints specify that the projection of o_{τ_j} should remain inside the bounding box. For each semantic measurement z_k taken on semantic keyframe $K_i \in \mathcal{K}^j$ and associated with τ_j, we have

$$x_{k,min} \leq u \leq x_{k,max}, \quad y_{k,min} \leq v \leq y_{k,max} \tag{11}$$

where $[u, v, 1]^T = P_i o_{\tau_j}$ is the projection of o_{τ_j} on semantic keyframe K_i.

Constraints (9), (10) and (11) are linear. A quadratic programming problem with linear constraints can be readily solved using convex optimization tools such as CGAL [19].

The solution of the quadratic programming represents a generic quadric surface, not necessarily an ellipsoid; we therefore constrain each quadric to be an ellipsoid by extracting the quadric rotation, translation and shape as shown in (12), using methods introduced in [20]:

$$\mathbf{Q}^* = \begin{bmatrix} \mathbf{R}\,\mathrm{diag}(a^2, b^2, c^2)\mathbf{R}^T - tt^T & t \\ t^T & -1 \end{bmatrix} \tag{12}$$

where $t = [t_1, t_2, t_3]^T$ is the translation vector, $\mathbf{R}$ is the 3×3 rotation matrix and a, b, c are the semi-axes of the ellipsoid.

Hence, we initialize all map objects by solving the quadratic programming problem (7) over the complete set of detections for each map object, and constrain the estimated quadrics to be ellipsoid. The constrained quadrics are deemed failure and discarded if they do not satisfy constraints (9), (10) and (11) or have an average re-projection error greater than 100 pixels. Further analysis of the failure cases indicates that a minimum baseline of 0.02m and a minimum parallax of 5.2° are required for successful initialization. Note that our initialization method (marked by 'Quadratic') yields a higher success rate than the method originally proposed in [6] (marked by 'SVD'), as shown in Fig. 3. Another observation is that quadric initialization has a higher success rate if more observations are utilized; thus, we require a minimum number of ten observations to ensure robust initialization.

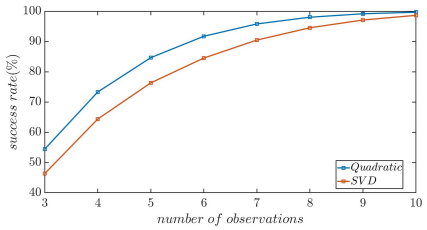

Fig. 3. The initialization success rate

B. Bundle Adjustment

The object initialization process only utilizes the first few observations and the generated quadrics are still sub-optimal. In order to take the full advantage of all the observations, we perform a joint optimization of map objects and camera poses of semantic keyframes in bundle adjustment.

Our data association pipeline would produce few errors, as demonstrated in the experiment section. With accurate data association, the maximum a posterior (MAP) estimation of map objects and camera poses of semantic keyframes can be solved to make improvements on the initialization results by maximizing the product of factors:

$$\mathcal{X}^*, \mathcal{T}^* = \underset{\mathcal{X},\mathcal{T}}{\operatorname{argmax}} \underbrace{\prod_i p(u_{i+1} \mid x_{i+1}, x_i)}_{\text{Odometry}} \cdot \underbrace{\prod_i p(x_i)}_{\text{Pose Prior}} \cdot \underbrace{\prod_j \prod_{\{i:K_i \in \mathcal{K}^j\}} p(z_i^j \mid x_i, \tau_j)}_{\text{Semantic}} \cdot \underbrace{\prod_j p(\tau_j)}_{\text{Object Prior}} \tag{13}$$

where $\mathcal{T} = \{\tau_j\}$ is the set of map objects, $\mathcal{X} = \{x_i = T_w^i\}$ is the set of camera poses of semantic keyframes, u_i is the odometry measurement and z_i^j is the semantic measurement of object τ_j on semantic keyframe K_i. Further, assuming Gaussian measurement and process models and uniform distribution of object and pose, by taking the negative log on the objective function, (13) can be rewritten as a nonlinear least-squares problem:

$$\mathcal{X}^*, \mathcal{T}^* = \underset{\mathcal{X},\mathcal{T}}{\operatorname{argmin}} \sum_i \|h_o(x_{i+1}, x_i) - u_{i+1}\|^2_{\Sigma_u} + \sum_j \sum_{\{i:K_i \in \mathcal{K}^j\}} \|h_s(x_i, \tau_j) - z_i^j\|^2_{\Sigma_z} \tag{14}$$

where $\|\cdot\|_\Sigma$ is the Mahalanobis norm, Σ_u and Σ_z are the covariance matrices of odometry measurements and semantic measurements respectively, h_o and h_s are the sensor models of odometry and semantic measurements respectively. While h_o is already known, h_s needs to be established. Since we perform data association for objects and semantic measurements with the same class label, the predicted object class is always aligned with the measurements. On the other hand, there would be discrepancy between predicted bounding box and the bounding box from associated semantic measurement. The predicted bounding box is calculated from the dual conic projection from object τ_j on semantic keyframe K_i:

$$C_{ij}^* = P_i Q_{\tau_j}^* P_i^T \tag{15}$$

where C_{ij}^* is a 3×3 matrix defining the dual conic projection, P_i is the projection matrix. The predicted bounding box $h_s(x_i, \tau_j)$ is set to be the smallest bounding box containing the part of the conic on the image.

With the established sensor model, the problem described in (14) can be readily solved using modern libraries such as g2o [21]. If a semantic keyframe is inserted when bundle adjustment is busy, a signal is sent to stop bundle adjustment, so that the semantic mapping thread can process the new semantic keyframe as soon as possible.

V. Experimental Evaluation

All experiments are carried out with an AMD Ryzen 3700X (8 cores @ 3.6GHz), a NVIDIA RTX 3060Ti and 16Gb RAM. The proposed semantic slam algorithm takes approximately 0.6Gb RAM. The computation bottleneck lies in the semantic object detection and tracking thread, which takes 32ms on average and allows real-time processing.

A. Data Association

We evaluate the data association performance on the TUM RGB-D fr1_xyz sequence [22]. The original sequence is appended with semantic measurements from YOLOv3. The ground truth object IDs are then manually labeled for those semantic measurements.

First, we clarify how accuracy is evaluated in our experiment. The key idea is to find the correspondences between the object IDs assigned to semantic measurements in ground truth dataset and the IDs assigned by our data association algorithm, as shown in Fig. 4. This task is formulated and solved as a maximum weighted bipartite matching problem. The complete bipartite graph has two disjoint vertices sets U and V. The IDs labeled in the ground truth dataset form the vertices set U. The IDs assigned in the data association algorithm form the vertices set V. The reward function $r(i, j)$ on a particular edge $(i, j) \in U \times V$ is the number of semantic measurements assigned ID $i \in U$ in the ground truth dataset and ID $j \in V$ in the data association algorithm respectively. For example, if our dataset only consists of the one image shown in Fig. 4, then we would have $r(8, 2) = 1$ and $r(8, 5) = 0$ because book object 8 on the left and book object 5 on the right do not share the same bounding box. Assume IDs assigned by our data association algorithm are identical to the IDs in ground truth dataset on every frame, then the total reward we obtain is simply the number of semantic measurements assigned IDs in both the ground truth dataset and the data association algorithm. This reward, denoted by r_{max}, is the maximum possible reward. By solving the maximum weighted bipartite matching problem, we obtain a particular matching. The sum of rewards of all edges inside that matching is considered the rewards obtained by our data association algorithm r_{da}. The ratio r_{da}/r_{max} is considered the accuracy of our data association algorithm.

The results show that the accuracy of our data association algorithm is $790/857 \approx 92.19\%$. The errors can be

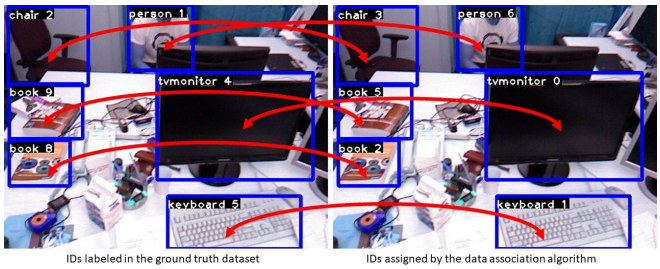

Fig. 4. The correspondences between the IDs labeled in the ground truth dataset and the IDs assigned by the data association algorithm are found by solving the maximum weighted bipartite matching problem.

put into two categories: (I) 60 mismatch cases are due to YOLOv3 assign different object classes to the same object. Since our data association algorithm assumes that semantic measurements with different object classes correspond to different map objects, errors can occur. (II) 9 mismatch cases are caused by errors occurred inside our data association pipeline. Those errors tend to happen for objects with textureless surface, such as the monitor shown in Fig. 4.

B. Results of Semantic Mapping

Evaluated on the TUM RGB-D fr2_desk sequence, the results of semantic mapping are shown in Fig. 5. Fig. 5 shows the projected quadrics of map objects (in green) on the image plane. The blue bounding boxes were results from YOLOv3. For initialization, only the first few observations are utilized. Hence, when observing from frames whose parallaxes with respect to the initialization frames are significant, we can see that the initialization results are far from ideal. On the other hand, after performing the bundle adjustment, we can see that improvements have been made upon the initialization results, and the quadric projections on the image tightly fit the bounding boxes except for the potted plant 0. The quadric representation of this object is optimized to a low-volume ellipsoid. This problem will be addressed in future work by incorporating object shape prior to bundle adjustment. An video file showing this mapping process can be found at https://www.youtube.com/watch?v=K86zXKdobTU.

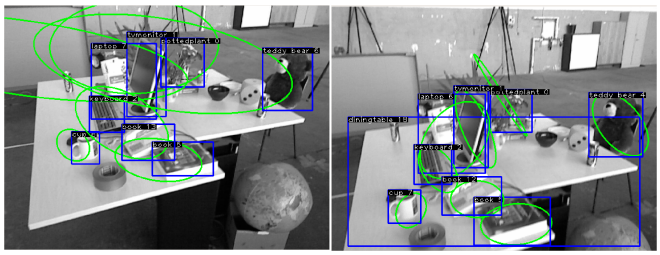

Fig. 5. Initialized map objects (left) and map objects after performing bundle adjustment (right).

Table I summarizes the tracking and mapping results tested on several TUM sequences. The mapping accuracy is evaluated by calculating the reprojection error [23] for all objects (in pixels). The average reprojection error decreases significantly after performing bundle adjustment, suggesting that the optimized quadrics in 3D space are reasonable representations of the actual objects. As for the tracking results, comparing with ORB-SLAM [10], our proposed semantic slam algorithm has larger average absolute trajectory error, due to the noisy nature of the semantic measurements.

TABLE I
TRACKING AND MAPPING RESULTS ON TUM SEQUENCES

Metrics	fr3/long_office_household	fr2/xyz	fr1/desk	fr2/desk
Initial reprojection error	87.8427	78.6246	82.3736	79.7602
Reprojection error after BA	33.9423	28.7704	34.5126	24.8281
Average ATE of proposed method	0.04694	0.04427	0.02328	0.05749
Average ATE of ORB-SLAM2	0.02173	0.00615	0.01760	0.01598

VI. CONCLUSIONS

In this paper, we have developed an integrated, keyframe-based semantic SLAM system without the use of pre-built object database. We have addressed the key problem of object-level data association utilizing both geometric and appearance information of map objects. We have also proposed a novel object initialization scheme to boost the success rate of object initialization. The experimental results show that our data association algorithm has an accuracy of 92.19% and that the generated semantic map is a reasonable presentation of the objects in the environment.

Albeit the good results on data association and mapping, we have observed an increase in tracking errors comparing to ORB-SLAM2. The result suggests the need for combining high-level semantic information and low-level geometric information in cases when high-accuracy in tracking performance is needed. This will be one further step of our work. We will also consider devising a more sophisticated scheme for semantic keyframe selection in future work.

APPENDIX

The attributes of map objects: In each map object τ_j, the following attributes are stored:

- Quadric shape represented by three semi-axes: a, b, c.
- Rotation matrix of the quadric $\mathbf{R}$.
- Translation of the quadric $o_{\tau_j} = t$.
- The class label of the object.
- The set of semantic keyframes $\mathcal{K}^j$ to observe this object.
- The number of observations.

The attributes of semantic keyframes: Each semantic keyframe K_i stores the following information:

- The camera pose $\mathbf{T}_i^w$ with respect to the world frame, in terms of a homogeneous transformation matrix.
- The camera intrinsic matrix.
- Objects observed in this semantic keyframe.
- The set of semantic measurements taken at this frame, associated or not to a map object.
- BoW vectors for the ROI defined by bounding boxes, used to describe observed map objects.

REFERENCES

[1] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[2] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1352–1359.

[3] D. Gálvez-López, M. Salas, J. D. Tardós, and J. Montiel, "Real-time monocular object slam," *Robotics and Autonomous Systems*, vol. 75, pp. 435–449, 2016.

[4] N. Atanasov, S. L. Bowman, K. Daniilidis, and G. J. Pappas, "A unifying view of geometry, semantics, and data association in slam." in *IJCAI*, 2018, pp. 5204–5208.

[5] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic slam," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 1722–1729.

[6] L. Nicholson, M. Milford, and N. Sünderhauf, "Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 1–8, 2018.

[7] L. Zhang, L. Wei, P. Shen, W. Wei, G. Zhu, and J. Song, "Semantic slam based on object detection and improved octomap," *IEEE Access*, vol. 6, pp. 75 545–75 559, 2018.

[8] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "Semanticfusion: Dense 3d semantic mapping with convolutional neural networks," in *2017 IEEE International Conference on Robotics and automation (ICRA)*. IEEE, 2017, pp. 4628–4635.

[9] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg, "Joint semantic segmentation and 3d reconstruction from monocular video," in *European Conference on Computer Vision*. Springer, 2014, pp. 703–718.

[10] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[11] M. Bjelonic, "YOLO ROS: Real-time object detection for ROS," https://github.com/leggedrobotics/darknet_ros, 2016–2018.

[12] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[13] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[14] B. Waggener, W. N. Waggener, and W. M. Waggener, *Pulse code modulation techniques*. Springer Science & Business Media, 1995.

[15] A. V. Goldberg and R. Kennedy, "An efficient cost scaling algorithm for the assignment problem," *Mathematical Programming*, vol. 71, no. 2, pp. 153–177, 1995.

[16] J. R. Kennedy, *Solving unweighted and weighted bipartite matching problems in theory and practice*. Citeseer, 1995, no. 1556.

[17] R. Burkard, M. Dell'Amico, and S. Martello, *Assignment problems: revised reprint*. SIAM, 2012.

[18] L. Perron and V. Furnon, "Or-tools," Google. [Online]. Available: https://developers.google.com/optimization/

[19] The CGAL Project, *CGAL User and Reference Manual*, 5.1 ed. CGAL Editorial Board, 2020. [Online]. Available: https://doc.cgal.org/5.1/Manual/packages.html

[20] C. Rubino, M. Crocco, and A. Del Bue, "3d object localisation from multi-view image detections," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 6, pp. 1281–1294, 2017.

[21] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3607–3613.

[22] J. Sturm, S. Magnenat, N. Engelhard, F. Pomerleau, F. Colas, W. Burgard, D. Cremers, and R. Siegwart, "Towards a benchmark for rgb-d slam evaluation," in *Proc. of the RGB-D Workshop on Advanced Reasoning with Depth Cameras at Robotics: Science and Systems Conf. (RSS)*, Los Angeles, USA, June 2011.

[23] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.