Towards Accurate Loop Closure Detection in Semantic SLAM with 3D Semantic Covisibility Graphs

Zhentian Qian[1], Jie Fu[2], and Jing Xiao[1]

***Abstract*— Loop closure is necessary for correcting errors accumulated in simultaneous localization and mapping (SLAM) in unknown environments. However, conventional loop closure methods based on low-level geometric or image features may cause high ambiguity by not distinguishing similar scenarios. Thus, incorrect loop closures can occur. Though semantic 2D image information is considered in some literature to detect loop closures, there is little work that compares 3D scenes as an integral part of a semantic SLAM system. This paper introduces an approach, called SmSLAM+LCD, integrated into a semantic SLAM system to combine high-level 3D semantic information and low-level feature information to conduct accurate loop closure detection and effective drift reduction. The effectiveness of our approach is demonstrated in testing results.**

***Index Terms*—SLAM, Semantic Scene Understanding, Data Sets for SLAM.**

I. INTRODUCTION

LOOP closure is an essential part of a SLAM system. During a long time motion of a robot, errors often accumulate in the estimated robot poses from SLAM and cause the estimated robot trajectory to drift. Loop closure methods are used to detect whether the robot has returned to a place it has visited before (place recognition) and correct the accumulated localization error if it has. Classic loop closure methods such as the method employed in ORB-SLAM2 [1] rely on geometric features to detect loop closure and compensate for the drifting errors. Such methods can have difficulties distinguishing similar scenarios or environments with symmetry if the low-level geometric features in those environments are very similar. More recently, some researchers have considered using semantic information in SLAM to aid the task of loop closure. In the following, we provide an overview of related approaches.

A. Related Work

1) Semantic SLAM: Semantic SLAM methods are focused on representing, mapping, and localizing 3D objects. The pioneering work of SLAM++ [2] builds object meshes using prior object models and an RGB-D camera. Improving on SLAM++, the work in [3] relies solely on monocular input and learns the scale of the map from prior object models. Lifting the requirement of any prior object models, QuadricSLAM [4] and CubeSLAM [5] build simplified object representations as quadric and cuboid, respectively. However, neither QuadricSLAM nor CubeSLAM addressed the issue of loop closure. In [6], an octomap is used to represent objects based on semantically augmented 3D point clouds from RGB-D SLAM.

2) Semantic loop detection: For loop detection, one line of research seeks to extract semantic information in an image to construct an image descriptor, which is later used to compare and find images from the same scene (i.e., loop detection). The semantic information can be encoded in a vector, as done in Yang *et al.* [7], Hu *et al.* [8] and Merill and Huang [9]. The differences among those approaches lie in how the semantic vector is constructed. For example, the semantic vector generated in [7] encodes the object class information presented in the image, while the semantic vector created in [8] encodes the object class, confidence score, and bounding box shape. On the other hand, Merill and Huang [9] resort to a deep learning method and construct a multi-decoder variational autoencoder (VAE) that encodes the visual appearance and semantic information to construct a global image descriptor.

Semantic information is also used to trim away the features on dynamic objects or non-discriminative features. For example, Chen *et al.* [10] mask out dynamic objects in loop closure candidate images, removing the interference from the instance-level semantic inconsistencies. The remaining features on candidate images are then used for geometric verification [11]. Using semantic segmentation, Mousavian *et al.* [12] only convert features of artificial structures to Bag of Words (BoW) vectors and discard non-discriminative features on vegetation, sky, and road. The generated BoW vectors are used for image comparison and proven to improve the accuracy of location recognition.

Semantic information can also be encoded in a graph. Such a graph represents the observed environment, with semantic landmarks as the vertices and the edges retaining the structure information between semantic landmarks. Wang *et al.* [13] and Cascianelli *et al.* [14] both introduced covisibility graphs for this purpose. To detect loop closures, the overall similarity of a query image and a candidate image is composed of two components: the similarity of landmark vertices and the similarity of the neighboring information of the vertices in

Manuscript received: September, 9, 2021; Revised December, 7, 2021; Accepted January, 3, 2022.

This paper was recommended for publication by Editor Javier Civera upon evaluation of the Associate Editor and Reviewers' comments.

[1] Zhentian Qian and Jing Xiao are with the Robotics Engineering Department, Worcester Polytechnic Institute, Worcester, Massachusetts, USA. zqian, jxiao2@wpi.edu

[2]Jie Fu is with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, Florida, USA. fujie@ufl.edu

Digital Object Identifier (DOI): see top of this page.

each image subgraph. CNN features and landmark region shapes are used to compare landmarks in both papers. While [13] uses random walk descriptors to compare the neighboring information of the nodes in each image subgraph, [14] opts to use adjacency matrix.

However, those methods treat loop detection separately from a running semantic SLAM system: loop detection is not integrated into the SLAM system and does not utilize the multi-modal information generated by the semantic SLAM system. The loop detection methods only use the information presented in image frames.

3) Semantic Loop Closure Integrated in Semantic SLAM: Only a few methods integrate semantic loop closure with semantic SLAM. For example, Liu *et al.* [15] extract a global semantic graph from the dense semantic map built by semantic SLAM based on RGB-D sensing and then use random walk descriptors to match the nodes in the query graph with the nodes in the global semantic graph. Benefiting from the 3D information stored in the object point clouds, the matching is only conducted coarsely. Even with inexact matching, the final localization can still be performed by aligning the semantic points associated with the matched nodes. Li *et al.* [16] use semantic SLAM based on monocular vision to construct cuboid semantic landmarks for objects in the environment. A similarity transformation is computed for two sets of semantic landmarks presented in the candidate and query images, respectively. However, correspondences are not calculated between the two sets of semantic landmarks, and the method exhaustively iterates over all possible matches, which can be time-consuming. Moreover, the method does not consider the structural information of objects in a scene.

B. Contributions

In this paper, we introduce a novel method for loop detection and drift correction in a semantic SLAM system based on monocular vision to take full advantage of both high-level semantic and low-level geometric information. We call our method SmSLAM+LCD (for Semantic SLAM and Loop Closure Detection). The main contributions are:

- A novel approach to organize semantic object information into 3D semantic covisibility graphs.
- A hierarchical loop detection approach. Loop closure candidates are first proposed based on low-level geometric features and then checked by comparing the corresponding 3D semantic covisibility subgraphs to avoid false positives.
- A coarse-to-fine approach to compute the SIM(3) transformation between the candidates for loop closure.
- A virtual dataset and a real-world dataset for testing if loop closure algorithms will produce false positives (available at https://users.wpi.edu/%7Ezqian/#datasets).
- Testing results to demonstrate the effectiveness of the introduced SmSLAM+LCD approach.

II. SYSTEM OVERVIEW

We embed our approach for loop closure in a semantic SLAM system that we developed in [17], featuring semantic object detection, tracking, and mapping. Fig. 1 provides an overview of the system. We highlight the contributions of this paper in blue.

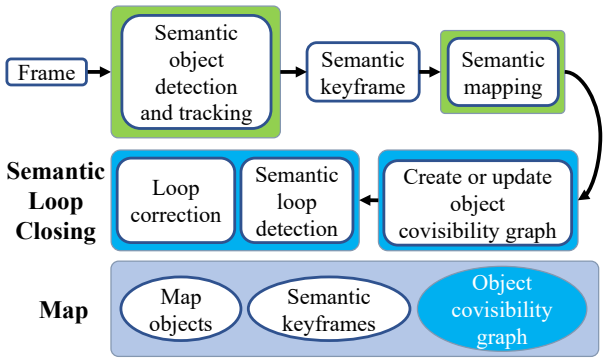

Fig. 1. The SmSLAM+LCD system overview.

First, we briefly review the semantic object detection, tracking, and mapping components [17]. The semantic object detection and tracking component processes the incoming frames from a monocular camera. The SLAM tracking module from ORB-SLAM2 [1] is used to provide visual odometry. At the same time, an object detector (YOLOv3 [18]) is used to detect the objects on the image. ORB features are then extracted in the region of interest (ROI) where objects are detected and subsequently converted to Bag of Words (BoW) [11] vectors to describe the detected objects. The final step is to perform the object-level data association to match the detected objects to the map objects.

The semantic mapping component is performed on every new semantic keyframe K_c generated by the semantic object detection and tracking component. When a new K_c comes in, we update the map database to include this keyframe and the relations between map objects and semantic keyframes.

The mapped semantic objects from each image keyframe are further organized in an object covisibility graph as detailed in Section III. The covisibility graph is also updated based on new observations made.

After the semantic mapping component processes the keyframe K_c, the semantic loop detection component then compares it with the keyframes stored in the map database for possible loop closures, as detailed in Section IV.

III. OBJECT COVISIBILITY GRAPH

A. Graph Definition

An object covisibility graph $G = (V, E)$ is maintained in the map to provide a structured environment representation.

1) Vertices V: The vertices of the covisibility graph are the 3D semantic landmarks built by our semantic SLAM algorithm, that is, the map objects in the map database. Each mapped object is associated with a probability distribution of the object class labels to capture perception uncertainty. The term "map object" or "semantic landmark" will be used interchangeably with "vertex" of the graph in this paper. Each map object contains the following information:

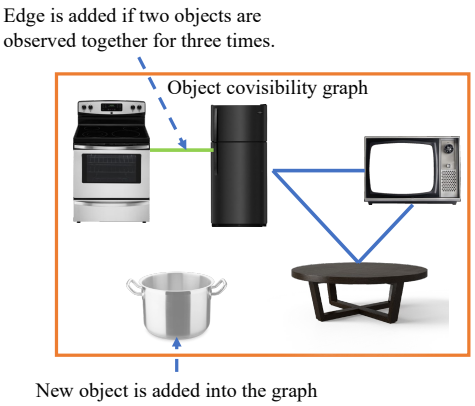

Fig. 2. Updating an example object covisibility graph: a new edge and a new object are added.

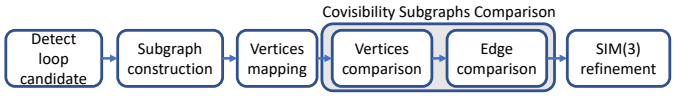

Fig. 3. The semantic loop detection method.

- The length of the principal axes of an ellipsoid that can tightly enclose the map object, denoted by l_a, l_b, l_c, with l_a being the length of the major axis.
- The position of the map object center t.
- Let $\mathcal{L}$ be the set of all object classes and $\mathcal{D}(\mathcal{L})$ be the set of probability distributions over the set $\mathcal{L}$. The vertex stores the probability distribution of the object class $p \in \mathcal{D}(\mathcal{L})$. Given an object class $z \in \mathcal{L}$, the probability of the vertex belonging to z is $p(z)$.
- The set of keyframes sharing observations of the map object.
- A set of BoW vectors calculated from the image patches containing the map object on the keyframes.

2) Edges: An edge is added between two vertices whenever the map objects associated with the two vertices are observed in the same semantic keyframes at least three times.

B. Graph Construction and Maintenance

The object covisibility graph is updated when a new semantic keyframe K_c is created, as shown in Fig. 2.

First, the map objects of the covisibility graph are updated. If a new object is observed in the semantic keyframe K_c and is not included in the covisibility graph, such as the pot in Fig. 2, it is added into the set of vertices. Second, the edges of the covisibility graph are updated. If an arbitrary pair of map objects v_i and v_j have been observed in the same semantic keyframes three times, such as the fridge and washing machine in Fig. 2, an edge is added between them.

IV. SEMANTIC LOOP DETECTION

Fig. 3 outlines our method for semantic loop detection.

A. Detection of Loop Candidates

First, a set of loop closure candidates $\mathcal{K}_l$ are selected from the keyframes in the map. For the current keyframe K_c, keyframe $K_l \in \mathcal{K}_l$ is considered as a loop closure candidate only if the similarity score between K_c and K_l passes a calculated minimum threshold $s_{\min}$. The similarity score is obtained by computing the L_1-score between the two BoW vectors $\mathbf{d}_c$ and $\mathbf{d}_l$ of frame K_c and K_l:

$$s(\mathbf{d}_c, \mathbf{d}_l) = 1 - 0.5\left|\mathbf{d}_c/|\mathbf{d}_c| - \mathbf{d}_l/|\mathbf{d}_l|\right|. \tag{1}$$

We also impose the same temporal consistency in ORB-SLAM2 [1] on generating loop closure candidates.

B. Subgraph Construction

For each loop candidate K_l, our method further validates whether K_l and K_c correspond to the same place by comparing the corresponding covisibility subgraphs. Those covisibility subgraphs are vertex-induced subgraphs of the object covisibility graph stored in the map. Their vertices are only a subset of the map objects in the object covisibility graph, and their edges are directly inherited from the object covisibility graph. The vertices of the covisibility subgraph for a keyframe are the map objects observed on the keyframe; hence only a subset of the map objects are in the covisibility subgraph. We denote the subgraph of K_c as $G_c = (V_c, E_c)$ and subgraph of K_l as $G_l = (V_l, E_l)$.

C. Mapping of Vertices

To compare two covisibility subgraphs, we need to establish the correspondence between the map objects of the two subgraphs. Benefiting from the rich information stored in each map object (refer to Section III), we can obtain the correspondence by solving a maximum weighted bipartite matching problem. For every possible map object pair $v_i \in V_c$ and $v_j \in V_l$, a similarity score is calculated:

$$s_n(i,j) = s_a(i,j) \cdot s_c(i,j) \tag{2}$$

where $s_a(i,j)$ and $s_c(i,j)$ are the appearance similarity and class similarity scores between v_i and v_j, respectively. Empirical study suggests that using a product instead of a weighted sum between the two scores $s_a(i,j)$ and $s_c(i,j)$ yield better results. Let d_i be the BoW vector converted from the image patch containing vertex v_i on keyframe K_c, and d_j be the BoW vector converted from the image patch containing vertex v_j on keyframe K_l. The appearance similarity score $s_a(i,j)$ is computed based on the L_1-score between d_i and d_j, and the formula is the same as (1). The object class similarity $s_c(i,j)$ between v_i and v_j employs the Bhattacharyya distance [19], which quantifies the distance between two class label probability distributions:

$$s_c(i,j) = \sum_{z \in \mathcal{L}} \sqrt{p_i(z) p_j(z)}, \tag{3}$$

where $p_i(\cdot)$ and $p_j(\cdot)$ are probability mass functions stored in vertices i and j, as described in Section III.

Next, we introduce a set of Boolean decision variables. For every possible vertex pair $v_i \in V_c$ and $v_j \in V_l$, let

$$x(i,j) = \begin{cases} 1, & \text{if } v_i \text{ is mapped to } v_j, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

$x(i,j) = 1$ represents that v_i and v_j correspond to the same map object and $x(i,j) = 0$ indicates otherwise. The maximum weighted bipartite matching problem is then formulated as:

$$\max_{x(i,j)\in\{0,1\}} \quad TS = \sum_{i=1}^{|V_c|}\sum_{j=1}^{|V_l|} s_n(i,j)x(i,j), \tag{5}$$

$$\text{s.t.} \sum_{i=1}^{|V_c|} x(i,j) \leq 1,\ \forall j; \quad \sum_{j=1}^{|V_l|} x(i,j) \leq 1,\ \forall i. \tag{6}$$

The constraints (6) mean that a vertex $v_i \in V_c$ can only be mapped to at most one vertex $v_j \in V_l$ and vice versa.

The problem defined in (5) can be solved using a cost-scaling push-relabel algorithm [20], readily implemented in [21]. This algorithm has a time complexity of $\mathcal{O}(\sqrt{n}m\log(nW))$, where n and m are the number of nodes and edges in the bipartite graph, W is the highest edge weight (all edge weights need to be converted to integers).

The map objects mapping solved in this fashion focuses on finding a mapping to maximize the total similarity score TS defined in (5). We calculate and threshold an average similarity score AS based on TS:

$$AS = TS/(\sum_{i=1}^{|V_c|}\sum_{j=1}^{|V_l|} x(i,j)) > \tau_{as}. \tag{7}$$

Loop candidates with an average similarity score AS lower than the threshold τ_{as} is discarded. A low average similarity score shows that the objects presented on the two loop candidate frames vastly differ, indicating two different places.

To eliminate possible false matches, we enforce an additional post-processing step. A match $x(i,j)$ will only be accepted if the similarity score $s_n(i,j)$ is greater than τ_n. Otherwise, if $s_n(i,j) < \tau_n$, $x(i,j)$ is rejected. This is a practice also commonly seen in matching methods for stereo vision [22]. All procedures are summarized in Algorithm 1.

Algorithm 1: Mapping between Map Objects

Input: map object sets V_c and V_l.
Output: all correspondence between map objects in the two sets V_c and V_l.

```
foreach v_i ∈ V_c do
    foreach v_j ∈ V_l do
        calculate the appearance similarity s_a(i,j);
        calculate the object class similarity s_c(i,j);
        calculate the similarity score s_n(i,j);
    end
end
compute all x(i,j) by solving the maximum weighted
  bipartite matching problem;
calculate the average similarity score AS using (7);
if AS < τ_as then return NULL;
foreach v_i ∈ V_c do
    foreach v_j ∈ V_l do
        if s_n(i,j) < τ_n then x(i,j) = 0;
return all x(i,j).
```

D. Comparison of Vertices

With the mapping between map objects established, we proceed to compare the map objects in G_l and G_c. The algorithm for comparing map objects is summarized in Algorithm 2. Note that drifts are accumulated in monocular SLAM and reflected in the pose and shape information stored in the map objects of the covisibility subgraph. To account for the drifts, Algorithm 2 calculates a similarity transformation from the current keyframe K_c to loop keyframe K_l. If the loop candidate frames K_l and K_c pass the map-object comparison test, a coarse SIM(3) transformation will be returned. If nothing is returned, the loop candidate is rejected.

Algorithm 2: Comparison of Map Objects

Input: a set of matched vertex pairs
Output: SIM(3) transformation S_c

```
if N < 3 then return NULL;
N_iter = 0;
while N_iter < maximum number of iterations do
    randomly sample 3 pairs of matched map objects;
    find SIM(3) S_c using the method of Horn;
    // Calculate the number of inliers
    N_M = 0;
    foreach matched vertex pair v_i and v_j do
        transform into common reference frame;
        if reprojection error < max error and
          |l_ai − l_aj| / max(l_ai, l_aj) < τ_s then
            N_M = N_M + 1
    end
    ε = N_M / N;
    N_iter = N_iter + 1;
    if N_M > m_inl and ε > τ_ε then return S_c;
end
return NULL.
```

A SIM(3) transformation has 7 degrees of freedom (DoF), and each pair of matched map objects provide 3 constraint equations. To compute this SIM(3) transformation, we must have a minimum of 3 pairs of matched map objects: $N = \sum_{i=1}^{|V_c|}\sum_{j=1}^{|V_l|} x(i,j) \geq 3$. If the loop candidate frames K_c and K_l contain fewer than 3 pairs of matched map objects, they will be rejected.

Taking the position information from paired map objects in G_l and G_c, we use the RANdom SAmple Consensus (RANSAC) algorithm [23] to find a similarity transformation. Three pairs of matched map objects are randomly selected to compute a similarity transformation using the method of Horn [24]. We then check the number of inliers supported by the computed transformation. For a pair of map objects $v_i \in V_c$ and $v_j \in V_l$ to be considered inliers, they must go through the following checks. First, we check the reprojection errors [25] of the map object centers under the current similarity transformation. Next, we check the scale consistency of v_i and v_j, that is, whether the two map objects are roughly of the same size. The two map objects are first transformed into a common reference frame, and their size is scaled accordingly to facilitate comparison. For scale consistency, we require: $|l_{ai} - l_{aj}|/\max(l_{ai}, l_{aj}) < \tau_s$, with l_{ai} and l_{aj} being the length of the major axis of vertex v_i and v_j in the common reference frame.

If the absolute number of inliers N_M and relative percentage of inliers $\epsilon = N_M/N$ surpasses the preset thresholds, the computed SIM(3) transformation is accepted, and the iteration is terminated. Otherwise, another set of three matched map objects would be randomly selected. If the maximum number of iterations is reached and the similarity transformation is still not computed, the proposed loop candidate fails the vertex comparison test and is rejected.

Note that we do not consider the orientation information or use scale information of the map objects directly in this step as the information contains some ambiguities. For any object with a similar size in two or more dimensions, its orientation cannot be uniquely defined. Multiple quadric representations with varied sizes may exist for the same object.

E. Comparison of Edges

Next, we compare the edges of the co-visibility subgraphs G_l and G_c. Adjacency matrices are constructed for map objects in G_l and G_c that are matched previously, denoted as A_l and A_c. With matches obtained from Section IV-D, we can make the row and column indices of adjacency matrices A_l and A_c the same, i.e., make the subgraphs aligned [26]. The similarity between the two adjacency matrices can be measured by normalized cross-correlation [27]:

$$s_e = \frac{\sum_{i=1}^{|V_c|}\sum_{j=1}^{|V_l|} A_l(i,j)A_c(i,j)}{\sqrt{\sum_{i=1}^{|V_c|}\sum_{j=1}^{|V_l|} A_l^2(i,j)\sum_{i=1}^{|V_c|}\sum_{j=1}^{|V_l|} A_c^2(i,j)}}. \quad (8)$$

The edge comparison succeeds if $s_e > \tau_e$, where τ_e is a pre-defined threshold. The loop with keyframe K_l is only accepted if comparisons of both edges and vertices succeed.

F. SIM(3) Refinement

As the positions of objects cannot be strictly defined, they are not precise; Hence the SIM(3) transformation S_c calculated in Algorithm 2 from map object position is still coarse, as described in Section IV-D. To address this issue, we adopt a coarse-to-fine method to calculate the SIM(3) transformation. In the final step of our semantic loop detection method, we refine the similarity transformation using low-level map points. First, we find a set of matched map points in the two loop candidate frames K_c and K_l. This search is sped up using the BoW vocabulary tree [1]. Next, using the coarse SIM(3) transformation S_c, we can enlarge the matched map points set by performing a guided search. The map points observed on frame K_l are reprojected on K_c based on S_c to find more matches and vice versa. Again, we perform RANSAC iterations on the matched map points to arrive at a new SIM(3) transformation S_f. This SIM(3) transformation S_f is then used to correct the drift errors.

V. EXPERIMENTAL EVALUATION

We have tested our loop closure algorithms in three datasets: TUM RGB-D dataset, virtual dataset, and custom real-world dataset. We have compared our method against the loop detection method used in ORB-SLAM2 [1] and ORB-SLAM3 [28]. The parameter values used in our method are presented in Table I.

TABLE I
PARAMETERS AND THEIR VALUES

τ_{as}	τ_n	τ_s	m_{inl}	τ_ϵ	τ_e
0.3	0.008	0.5	3	0.59	0.5

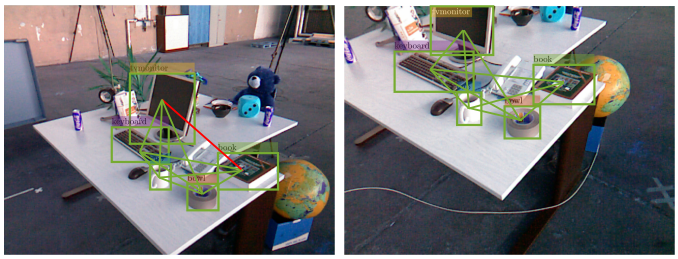

Fig. 4. Covisibility subgraphs on two true loop closure frames (left and right) in fr2/desk sequence. Green and red plots represent matched and mismatched map objects and edges, respectively.

A. Results from TUM Dataset

For the TUM RGB-D dataset, we focus on the fr2/desk and fr3/long_office_household sequences, which contain loop closures and have a certain number of objects in the scene. Since sequences in the TUM dataset are not recorded in similar places, for this test, we focus on the ability of our proposed method to detect loop closures and correct drifts.

1) Detection Results: To test the ability for loop detection, we wrote a script to label the loop closures in the two sequences automatically. For two keyframes to be labeled as a loop closure, their positional difference needs to be smaller than 1m, and their viewing angular difference is smaller than $53°$. To avoid mislabeling adjacent frames as loop closures, we require the ID difference between loop closure frames to be greater than 1,000. Table II shows the detection results from the TUM sequences, including five-time average of the precision and recall metrics. The results suggest that when comparing with ORB-SLAM2 and ORB-SLAM3, our method detects fewer loop closure candidates and hence yields fewer true positives because of stricter criteria, but is able to reach 100% precision in both sequences.

TABLE II
DETECTION RESULTS ON TUM SEQUENCES

DataSet	Metrics	**Ours**	ORB-SLAM2	ORB-SLAM3
fr2/desk	Detections	**23**	28	297
	Loop Closure	**2122**	2104	3067
	True Positives	**23**	28	51
	Avg. Precision	**100%**	100%	17%
	Avg. Recall	**1.1%**	1.4%	1.7%
fr3/long_ office_ household	Detections	**6**	37	348
	Loop Closure	**2429**	1322	2529
	True Positives	**6**	30	43
	Avg. Precision	**100%**	82%	13%
	Avg. Recall	**0.3%**	2.4%	1.7 %

The covisibility subgraphs on the true loop closure frames are shown in Figs. 4 and 5. The matched map objects and edges are plotted in green, while the mismatched ones are in red. The two loop closure candidate frames pass the

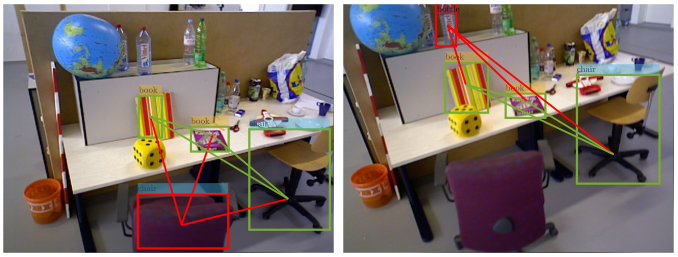

Fig. 5. Covisibility subgraphs on two true loop closure frames (left and right) in fr3/long_office_household sequence. Green and red plots represent matched and mismatched map objects and edges, respectively.

comparison tests for vertices and edges; thus, our method determined them correctly as true loop closures.

2) Ablation Study: Our method has many checks and relies on multiple thresholds. To demonstrate the contribution of each component to the performance and the roles of the thresholds, we conducted an ablation study with the results shown in Table III. In the top row, we report the precision and recall of the proposed initial loop candidates. From the top row to the bottom row, we gradually add additional checks as represented by the corresponding thresholds. We can see that with each additional check, our method is able to identify and reject more false loop closures and increase the precision. Thus each check makes its unique contribution. With all checks conducted, our method is able to reject all false loop closures and reach 100% precision for the given data set.

TABLE III
ABLATION STUDY RESULTS

Thresholds	fr2/desk		fr3/long_office_household	
	precision	recall	precision	recall
None	11.8%	4.3%	14.2%	5.6%
$+\tau_n$	11.9%	4.2%	14.9%	5.3%
$+\tau_{as}$	28.9%	3.6%	53%	3.4%
$+\tau_e$	29.3%	3.6%	54.5%	3.3%
$+\tau_s$, $+\tau_{epsilon}$	100%	1.1%	100%	0.3%

3) Drift Reduction: We represent a drift as the root mean square positional error between the estimated trajectory by SLAM and the ground truth trajectory. Benefiting from the coarse-to-fine approach to compute the SIM(3) transformation, the performance of our method in drift reduction is either on par with or better than the baseline methods. Table IV shows the five-time average of the trajectory errors tested on the two TUM sequences. The entry "ORB-VO" refers to the result of ORB-SLAM2 with loop closing turned off. Our method obtains the same level of drift reduction in the fr3/long_office_household sequence and outperforms ORB-SLAM2 and ORB-SLAM3 in the fr2/desk sequence.

Figs. 6 and 7 show the time-error plots of applying our (SmSLAM+LCD) method vs. applying ORB-SLAM2 and ORB-SLAM 3 in the two sequences. The sudden drop in the plot indicates when the loop closure happens. Because our method enforces a stricter loop closure check, the loop closure occurs later than ORB-SLAM2 and ORB-SLAM3. For drift reduction, our method shows superior performance.

TABLE IV
AVERAGE ERROR ON TUM SEQUENCES

	fr2/desk	fr3/long_office _household
ORB-VO	6.0cm	7.0cm
ORB-SLAM2	2.0cm	1.6cm
ORB-SLAM3	1.5cm	1.6cm
SmSLAM+LCD (this paper)	**1.4cm**	**1.6cm**

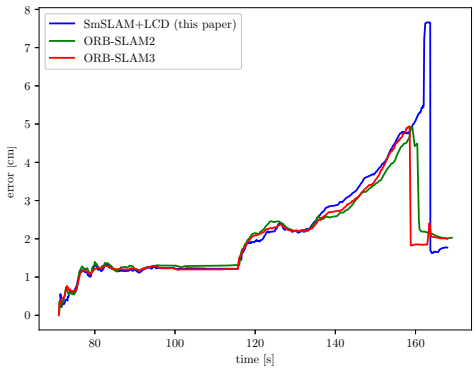

Fig. 6. Time-error plots of the fr2/desk sequence.

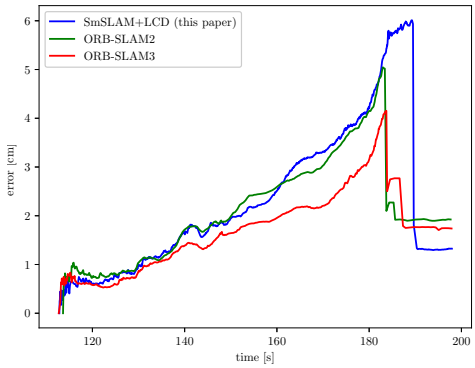

Fig. 7. Time-error plots of the fr3/long_office_household sequence.

B. Results from Virtual Dataset

The virtual environment is built using Unreal Engine 4 (UE4). UnrealCV [29] is used as the interface between the virtual environment and our system. This environment provides an ideal platform for testing our SmSLAM+LCD method, particularly its ability to distinguish very similar scenes. Our virtual world features a two-story apartment building, modified from the Archviz Interior Rendering sample project built by Epic Games. The second-floor apartment is designed to be similar to the first-floor apartment, though with slightly different furniture and decorations.

We first tested the loop closure capability in this dataset. A trajectory is designed to traverse the first floor and form a loop. Five-time consecutive tests show that in terms of reducing drifts, our method is able to obtain better performance, benefit from the coarse-to-fine approach to compute the SIM(3) transformation. The average trajectory error with the ORB-VO method is 2.3cm. After the loop is successfully closed, the average drift is reduced to 1.7cm by our method, comparing to 2.1cm by ORB-SLAM2 and 2.1cm by ORB-SLAM3. The

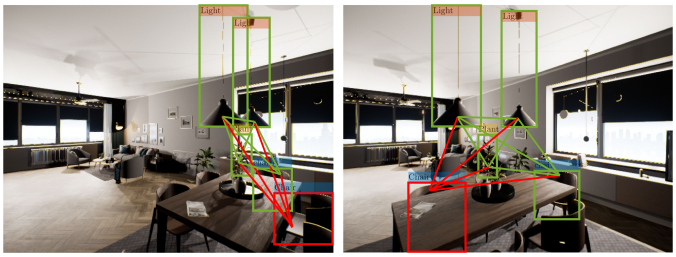

Fig. 8. Covisibility subgraphs on true loop closure frames in the virtual dataset. Green and red plots represent matched and mismatched map objects and edges, respectively.

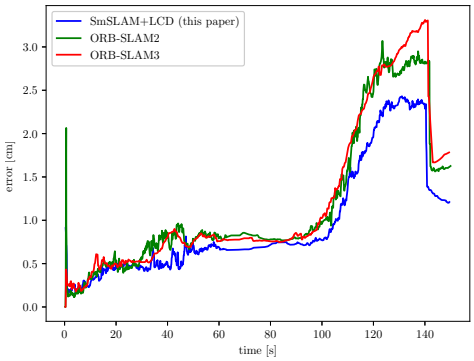

Fig. 9. Time-error plots of the virtual dataset.

covisibility subgraphs used to detect the true loop closure are shown in Fig. 8. The two covisibility subgraphs are in agreement except for the two chairs marked in red and the edges connected to them. Hence, our method correctly concludes that the two frames form a true loop closure. Fig. 9 shows the time-error plots of our method vs. ORB-SLAM2 and ORB-SLAM3. It is clear that our method achieves more significant drift reduction and sooner than ORB-SLAM2 and ORB-SLAM3.

Another trajectory is designed to traverse the first and the second floors to test the ability to distinguish similar places. Both ORB-SLAM2 and ORB-SLAM3 report a false loop closure between the first-floor apartment image and the second-floor apartment image, as shown in Fig. 10. The false loop closures on the designed trajectory are shown in Fig. 11, denoted by green and red lines, respectively. In contrast, our method can successfully distinguish the two apartments by comparing the object covisibility subgraphs presented on the two images. An incorrect match occurred between the two couches, represented by green bounding boxes in Fig. 10, but all other edges and vertices of the two covisibility subgraphs are not matched. Hence our method correctly concludes that the two images were taken at two different places.

C. Results from Real-world Dataset

Finally, we tested our algorithm in a real-world dataset. The dataset is constructed by a human operator carrying a ZED camera and traversing two adjacent offices. Posters with the same content are placed on the walls of the two offices to create two similar scenes. ORB-SLAM2 and ORB-SLAM3 report false loop closure when tested in this dataset.

Fig. 10. The false loop closure reported by ORB-SLAM2 and ORB-SLAM3. The left and right frames correspond to the first and second floor apartments. Covisibility subgraphs are also plotted on the two frames. Green and red plots represent matched and unmatched map objects and edges, respectively.

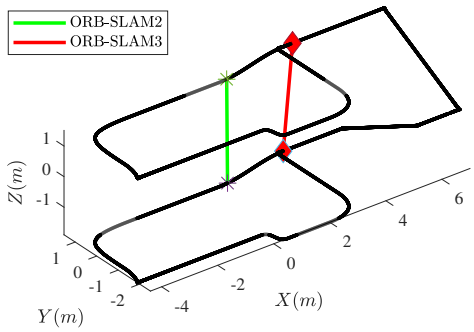

Fig. 11. False loop closure visualization on the designed trajectory.

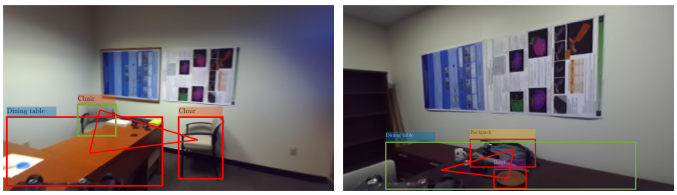

Fig. 12. The false loop closure reported by ORB-SLAM2 and ORB-SLAM3 but not by our method. The left and right frames correspond to the first and second offices, respectively. Covisibility subgraphs are also plotted on the two frames. Green and red plots represent matched and unmatched map objects and edges, respectively.

In contrast, our algorithm can successfully distinguish the two offices apart by comparing the covisibility subgraphs presented on loop candidate frames, as shown in Fig. 12. The chair in the first office is incorrectly matched with the dining table in the second office by our method, possibly because the detection of the dining table contains portions of a similar chair. Nevertheless, our method successfully distinguished other map objects of the two covisibility subgraphs and ruled out this false loop closure.

The estimated sensor poses of our method, ORB-SLAM2, and ORB-SLAM3 in the real-world dataset are plotted in Fig. 13. Black lines in the image mark the rough layout of the real-world environment. From Fig. 13, we can see that because ORB-SLAM2 and ORB-SLAM3 performed a false-positive loop closure, the estimated trajectories are severely distorted and cut through walls. In contrast, our method successfully rejected the false loop closure and maintained more accurate tracking.

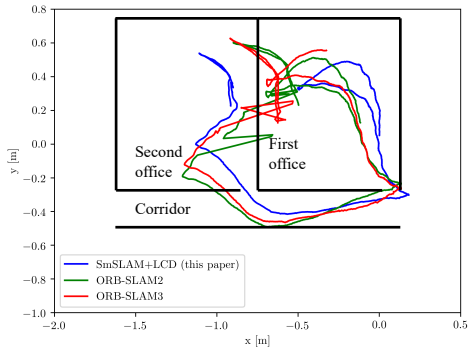

Fig. 13. Estimated sensor poses in the real-world dataset.

VI. CONCLUSIONS

In this paper, we have introduced SmSLAM+LCD, a semantic loop closure method well integrated into a semantic SLAM system of our prior work [17]. Our method leverages high-level semantic and low-level geometric information from the SLAM system for loop detection and drift correction. In particular, SmSLAM+LCD builds an object covisibility graph upon the mapped 3D semantic objects and constantly updates it with the latest observation. When performing loop detection, our method checks the loop candidates proposed based on low-level geometric features by comparing the object covisibility subgraphs associated with the loop candidate frames. To correct the accumulated drift, our method includes a coarse-to-fine approach to compute the SIM(3) transformation between loop closure frames.

We tested SmSLAM+LCD using the TUM RGB-D dataset and our own virtual and real-world datasets. The experimental results show that SmSLAM+LCD can achieve more accurate drift correction after detecting a loop closure than ORB-SLAM2 and ORB-SLAM3 and distinguish similar scenes to avoid false-positive loop closures reported by ORB-SLAM2 and ORB-SLAM3.

In this paper, we assume that every object contributes equally to the detection of a true loop closure or the rejection of a false loop closure. This could be modified to treat static and moving objects differently, which we plan to do next. We will also investigate using weights in the covisibility graphs to improve loop detection in dynamic environments.

REFERENCES

[1] 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.

[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] 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.

[5] S. Yang and S. Scherer, "Cubeslam: Monocular 3-d object slam," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 925–938, 2019.

[6] 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.

[7] C.-Y. Yang, Y.-C. Zhang, Y.-H. Chen, and C.-W. Huang, "Toward semantic loop closure in simultaneous localization and mapping systems," in *Current Developments in Lens Design and Optical Engineering XIX*, vol. 10745. International Society for Optics and Photonics, 2018, p. 107450I.

[8] M. Hu, S. Li, J. Wu, J. Guo, H. Li, and X. Kang, "Loop closure detection for visual slam fusing semantic information," in *2019 Chinese Control Conference (CCC)*. IEEE, 2019, pp. 4136–4141.

[9] N. Merrill and G. Huang, "Calc2. 0: Combining appearance, semantic and geometric information for robust and efficient visual loop closure," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4554–4561.

[10] H. Chen, G. Zhang, and Y. Ye, "Semantic loop closure detection with instance-level inconsistency removal in dynamic industrial scenes," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 3, pp. 2030–2040, 2020.

[11] 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.

[12] A. Mousavian, J. Košecká, and J.-M. Lien, "Semantically guided location recognition for outdoors scenes," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 4882–4889.

[13] Y. Wang, Y. Qiu, P. Cheng, and X. Duan, "Robust loop closure detection integrating visual–spatial–semantic information via topological graphs and cnn features," *Remote Sensing*, vol. 12, no. 23, p. 3890, 2020.

[14] S. Cascianelli, G. Costante, E. Bellocchio, P. Valigi, M. L. Fravolini, and T. A. Ciarfuglia, "Robust visual semi-semantic loop closure detection by a covisibility graph and cnn features," *Robotics and Autonomous Systems*, vol. 92, pp. 53–65, 2017.

[15] Y. Liu, Y. Petillot, D. Lane, and S. Wang, "Global localization with object-level semantics and topology," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4909–4915.

[16] J. Li, K. Koreitem, D. Meger, and G. Dudek, "View-invariant loop closure with oriented semantic landmarks," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7943–7949.

[17] Z. Qian, K. Patath, J. Fu, and J. Xiao, "Semantic slam with autonomous object-level data association," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 11 203–11 209.

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

[19] A. Bhattacharyya, "On a measure of divergence between two statistical populations defined by their probability distributions," *Bull. Calcutta Math. Soc.*, vol. 35, pp. 99–109, 1943.

[20] J. R. Kennedy Jr, "Solving unweighted and weighted bipartite matching problems in theory and practice," Ph.D. dissertation, Stanford University, Stanford, USA, 1995.

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

[22] J. Banks and P. Corke, "Quantitative evaluation of matching methods and validity measures for stereo vision," *The International Journal of Robotics Research*, vol. 20, no. 7, pp. 512–532, 2001.

[23] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[24] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Josa a*, vol. 4, no. 4, pp. 629–642, 1987.

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

[26] S. Feizi, G. Quon, M. Recamonde-Mendoza, M. Medard, M. Kellis, and A. Jadbabaie, "Spectral alignment of graphs," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 3, pp. 1182–1197, 2019.

[27] E. Stumm, C. Mei, S. Lacroix, and M. Chli, "Location graphs for visual place recognition," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 5475–5480.

[28] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, 2021.

[29] W. Qiu, F. Zhong, Y. Zhang, S. Qiao, Z. Xiao, T. S. Kim, and Y. Wang, "Unrealcv: Virtual worlds for computer vision," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 1221–1224.