2021 IEEE Intelligent Transportation Systems Conference (ITSC)
Indianapolis, USA. September 19-21, 2021

Hierarchical Loop Closure Detection for Long-term Visual SLAM with Semantic-Geometric Descriptors

Gaurav Singh, *Member, IEEE*, Meiqing Wu, *Member, IEEE*, Siew-Kei Lam, *Senior Member, IEEE*, and Do Van Minh

2021 IEEE International Intelligent Transportation Systems Conference (ITSC) | 978-1-7281-9142-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ITSC48978.2021.9564866

Abstract— **Modern visual Simultaneous Localization and Mapping (SLAM) systems rely on loop closure detection methods for correcting drifts in maps and poses. Existing loop closure detection methods mainly employ conventional feature descriptors to create vocabulary for describing places using bag-of-words (BOW). Such methods do not perform well in long-term SLAM applications as the scene content may change over time due to the presence of dynamic objects, even though the locations are revisited with the same viewpoint. This work enhances the loop closure detection capability of long-term visual SLAM by reducing the number of false matches through the use of location semantics. We extend a semantic visual SLAM framework to build compact global semantic-geometric location descriptors and local semantic vocabulary trees, by leveraging on the already available features and semantics. The local semantic vocabulary trees support incremental vocabulary learning, which is well-suited for long-term SLAM scenarios where the scenes encountered are not known beforehand. A novel hierarchical place recognition method that leverages the global and local location semantics is proposed to enable fast and accurate loop closure detection. The proposed method outperforms recent state-of-the-art methods (i.e., FABMAP2, SeqSLAM, iBOW-LCD, and HTMap) on all datasets considered (i.e., KITTI, Synthia, and CBD), with highest loop closure detection accuracy and lowest query time.**

I. INTRODUCTION

Loop closure detection in modern visual SLAM systems is used to correct drifts by finding the correspondences of the current place with previously visited places. It is also used during relocalization to recover the camera pose in the event of localization failure. Most of the existing loop closure detection methods use traditional visual bag-of-words (BOW) for place recognition. The BOW requires a training stage where a set of visual features (from the training images) are clustered to generate a vocabulary, i.e., set of quantized visual features called visual words [4]. During the loop closure detection, images are represented as histograms of occurrences of the visual words. However, the traditional feature-based BOW methods are ineffective in long-term SLAM applications where the scenes change over time. Furthermore, pre-trained BOW methods use static vocabularies which are not well-suited for long-term place recognition (as scenes may differ from the training images).

This paper improves the speed and accuracy of loop closure detection under dynamic scene conditions and varying viewpoints. Specifically, we leverage on the semantic information that are readily available in existing semantic visual SLAM systems [1, 14] to build location semantics for scene identification. We develop a viewpoint-invariant global semantic-geometric location descriptor which encodes the distances between various semantic blobs in a scene using a normalized histogram. This histogram provides a coarse description of the places, which can be quickly matched to retrieve a set of global locations (group of places). Each of the global locations maintains local semantic class-wise BOW vocabulary to enable precise image matching of the current place and the visited places. The semantic class-wise BOW vocabulary is learnt online through class-based grouping of tree nodes, and the local trees under a class node are learned using incremental BOW approach [5] without a pre-training stage.

A hierarchical loop closure detection method that utilizes the global and local location semantics is proposed. The hierarchical approach leads to lower query time by grouping similar (semantically and structurally) places, which are globally represented as semantic-geometric descriptors. Once a set of global locations is matched with the the current location, the corresponding class-wise BOW vocabulary, which incorporates higher descriptiveness of the scenes, is used to refine the place recognition. In addition, selective weighting of words based on the semantic class of the local-tree is used to discard dynamic objects in the matching process. This reduces the number of mismatches, leading to higher loop closure detection accuracy.

Our work highlights the importance of distinguishing between grossly dissimilar looking places based on global semantic and geometric cues, similar to the way humans perceive locations. Detail local descriptors are then used to refine place recognition within the global locations. The class-wise BOW vocabulary increases the robustness of scene identification in the presence of moving objects by providing semantic class-based weighting of words in vocabulary trees. The proposed method can be integrated into existing semantic visual SLAM systems for more accurate loop closure detection without incurring much computation cost. We provide extensive experimental results on well-known datasets to show that the proposed technique outperforms existing state-of-the-art loop closure detection methods in challenging environments.

Autonomous agents (robots, vehicles) perform navigation by localizing themselves with respect to a metric map of world and is commonly addressed as a joint problem known as SLAM. Due to sensor noise and other factors (limited sensor range, abrupt camera motion, scene dynamics etc.), the generated map and localization usually suffers from high inaccuracy or even localization failures [14]. Therefore, SLAM employs loop closure detection to find correspondences of the current scene with previously visited scenes and use them to rectify the pose or relocate the agent. Moreover, place recognition (crucial function of loop closure detection) plays an important role in distributed SLAM, where maps built by different agents need to be fused to create a globally consistent map.

We will only discuss loop closure detection in the context of visual SLAM (i.e., using camera sensor) due to its inherent advantages [3]. The primary challenges faced by loop closure detection methods are viewpoint invariance, query time complexity, and scalability [11]. Various methods have been proposed to deal with these challenges and majority of them falls into the category of appearance-based methods. The place appearance is associated with place descriptors. In topological representation, the agent's locations (using associated images) are partitioned into distinct locations based on their descriptors. If the place description contains only topological information, then place recognition just provides most likely locations, whereas if place description contains metric information, then precise location can be identified.

Among the appearance methods, the BOW model is the most popular [11, 4] due to its efficiency in representing places in the form of compact visual words and tree-based indexing. BOW is an image indexing model inspired by text-based document analysis. In this model, a vocabulary (dictionary) is maintained which contains distinct visual words. The local feature descriptors from the query image are quantized into set of representative visual words and the histogram of visual words is used as the image descriptor [4]. Each word also maintains a term frequency inverse document frequency (TF-IDF) score [11], which is the product of term frequency (TF) and inverse document frequency (IDF). The TF measures the frequency of appearance of a word across images and IDF measures how common is the word across all images. DBOW [4] and FAB-MAP are the best performing methods and are most commonly used in visual SLAM. DBOW [4] uses FAST features and binary BRIEF descriptors to increase speed. DBOW2 [4], which uses robust scale and rotation invariant ORB descriptor, has demonstrated improved accuracy and is also one of the fastest. FAB-MAP [10] and FAB-MAP 2.0 [2] also use an inverted index with a BOW model to perform visual place recognition. Chow-Liu tree is used for approximation of co-occurrence probabilities of visual words.

In the BOW model based methods [4], the visual vocabulary is traditionally trained offline due to the time consuming clustering of descriptors. This is one of the main limitation of these methods as the vocabulary is trained only for specific scenarios and require a separate training stage. As such, they do not perform well on unseen scenarios. To resolve this problem, [5] developed incremental BOW based loop closure detection (iBOW-LCD). In this method, vocabulary is learned incrementally on the incoming scenes. iBOW-LCD adapts to the current scene by updating the vocabulary tree via adding and deleting visual words. To keep the computational complexity low, it uses islanding to group images which are temporally close.

Methods such as [6] have demonstrated that place recognition speed can be improved by hierarchically grouping the places. HTMap [6] extracts PHOG as global feature descriptors and LDB as local feature descriptor from images, and then uses the global features for grouping similar places. It uses local features for final image matching. HTMap uses Bayes filter to combine the descriptor matching scores of the two levels. However, the transition model calculation becomes computationally expensive with increasing number of images. Another approach utilizes sequence of images to perform place recognition [12]. The main assumption of this method is that the camera passes through previously visited paths, which can be leveraged upon to improve place recognition in visually challenging environments.

Learning deep descriptors, such as LocNet [9], PointNetVLAD [15] for place recognition is another field of work. However, these methods require long training and inference time, which is not feasible for real-time visual SLAM running on resource-constrained embedded platforms. Furthermore, although there are several existing methods that exploit semantics for improving SLAM front-end, there is little work that exploits semantics for the SLAM back-end to perform loop closure detection.

Unlike existing works, our proposed method incorporates a hybrid topological-metric approach for describing places by grouping places with similar semantic-geometric structures, and using feature-based BOW to refine the representation of a place. In the latter, we use the training methodology for a basic vocabulary tree similar to iBOW-LCD. However unlike iBOW-LCD, our loop closure detection method creates separate semantic class-based nodes at the top level of the trees to distinguish between BOW descriptors of different classes and facilitates intra-class matching of descriptors.

The main contributions of this paper are as follows:

- A hierarchical loop closure detection method that finds semantically similar places, and refines place recognition using locally learned visual words. This leads to higher accuracy and lower query time, which is well-suited for long-term SLAM operations.
- A viewpoint-invariant global semantic-geometric descriptor that groups locations with similar semantic-geometric structures. A new location is created dynamically when the appearance of the environment, determined from the semantic descriptor, differs notably from past locations.
- Each location maintains its own vocabulary of local scenes that is learnt online. We propose local semantic vocabulary trees that allow simple class-wise discrimination of words and dynamic removal of moving objects.
- The proposed method outperforms other recent state-of-the-art methods (i.e., FABMAP2, SeqSLAM, iBOW-LCD, and HTMap) on all datasets considered (i.e., KITTI, Synthia, and CBD), with highest loop closure detection accuracy and lowest query time.

II. Proposed Method

For fast loop closure detection, the visited places need to be described in a compact form. At the same time, it must be descriptive enough to distinguish between dissimilar places. We propose a hierarchical description of places using location semantics to reduce the search space for place recognition by enabling large number of unrelated locations to be filtered without sacrificing the accuracy. The local semantic class-wise BOW vocabulary of the potential locations are then used to achieve precise place recognition. This overcomes the limitations of existing approaches that rely solely on feature-based BOW for queries, as the depth of the vocabulary tree increases with the number of places visited.

We store the historical information of the visited places in the form of global locations, $L = \{l_1, l_2, ...\}$. Each global location, $l_j \in L$, is a group of semantically similar images (or places) and is described by a single representative mean descriptor, $d_j^{repr\_mean}$. The $d_j^{repr\_mean}$ is the average descriptor of the group members (i.e. images in the group). We propose semantic-geometric descriptors, d_m^{GSG}, to represent an image I_m globally, and thereby utilizing the similarity in 3D semantic and geometrical structure of the scene. A location l_j also has a corresponding vocabulary tree ($Class\_Tree_j$) to refine place recognition within a location. The vocabulary tree ($Class\_Tree_j$) is built using local feature descriptors (we use ORB descriptors) through semantic-class-wise branching. The semantic-class-wise branching is used for improving the distinctness within BOW description and to separate weighting of words belonging to dynamic classes.

The proposed loop closure detection process is shown in Fig. 1. ORB feature descriptors (D_i^{Local}) and semantic map (S^i) are extracted from each incoming image (I^i). The semantic-geometric descriptor (d_i^{GSG}) is extracted from the semantic map (S^i). The d_i^{GSG} is then used to search for the matching location among the set of locations $L = \{l_1, l_2, ...\}$. If matching location is not found, a new location l_n is created by initializing its representative mean descriptor, $d_n^{repr\_mean}$ and initial mean descriptor, $d_n^{repr\_mean}$ as d_i^{GSG}. The corresponding $Class\_Tree_n$ is also initialized. Otherwise, if matching location $best\_loc$ is found, image matching is performed within the $best\_loc$ using $Class\_Tree_{best\_loc}$ and the location is updated using the new member image I^i.

Semantic-geometric descriptor extraction is discussed in Section II-A, while the search, creation and update of locations are discussed in Section II-B. Section II-C and Section II-D describes the Class-tree and image matching process respectively.

A. Global Semantic-Geometric Descriptor

We define a location as a group of similar semantic places (images). As semantic information is already available in most semantic visual SLAM systems, we utilize it to group places. However, the semantic entities alone cannot provide enough distinctiveness for place recognition. We hypothesize that the relationship among different semantic entities in the scene can provide sufficient distinct information about the place. While similar relationship has been modelled in the past using random walk descriptors (feature-based) to encode the neighbourhood information, they incur high computational complexity due to graph creation, random walk descriptor generation from graph, graph-matching. and continuous graph merging for each input image. We propose to formulate the neighbourhood information for all pairs of semantic class entities. The relationship between the semantic entities can be further strengthened with structural information of the scene. We propose to model the structural information using distances between the semantic entities. This leads to lower computational complexity compared to using distances between feature (key-points) pairs. The semantic entity pairs and the distances between them are used to generate histogram descriptors. Our analysis of these descriptors reveals that images with similar semantic-geometric structures have similar (low distance) normalized histogram (distribution). The complete process of generating these descriptors is explained next.

First, from the pixel-wise semantic segmentation of the current image, we extract connected regions of each class (also known as blobs). The blobs are smoothed using morphological operations (i.e. dilation and erosion) to remove unwanted noise (holes, disconnected edges, and invalid labels). This ensures clean boundaries between semantic segments. Each blob is then represented by its centroid in 3D camera coordinate. To re-project the blob centroid from 2D image coordinate into 3D camera coordinate, depth information is utilized (also available in most SLAM systems). Each blob is then represented as its semantic label s_i and 3D centroids (X_i), as shown in Fig. 2 (blob extraction step). Note that there can be multiple instances of the same semantic class, for example class 'A' has two instances in Fig. 2. The semantic relationships are then encoded as pairs of semantic classes present in the current scene. 3D distances between blobs accounts for the semantic-geometry present in the current scene. These distances are then uniformly quantized into 'K' bins (e.g. 0-10m, 10-20m, and $\geq$ 20m in Fig. 2 for bin range, R=10 and K=3) for each type of semantic pair (AA, AB, and so on, in Fig. 2). Then for all these semantic pair bins, histogram is generated by counting the number of edges present in a particular bin and semantic pair. The bins are then concatenated and normalized with respect to the number of edges present, to obtain the descriptor. As such, the descriptor encodes of distribution of semantic-geometries, i.e. how the pairs of semantic entities are distributed in a scene. If two different scenes contain same semantic classes but significantly different edge distance between entities, then their descriptors (i.e. histogram/distributions) are also different. However, if two scenes have similar semantic pairs and their corresponding distances are also in the same bins, then they can be grouped together. The number of bins determines the distinctness of the descriptor. Larger 'K' value will lead to sparser and distinct descriptors, and therefore, more number of locations are generated. On the other hand, smaller 'K' will lead fewer number of locations, because the

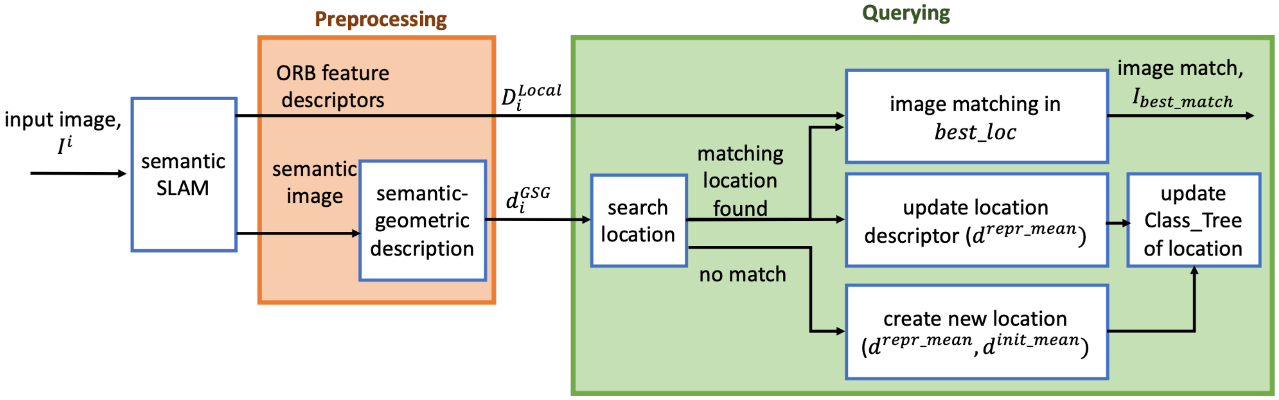
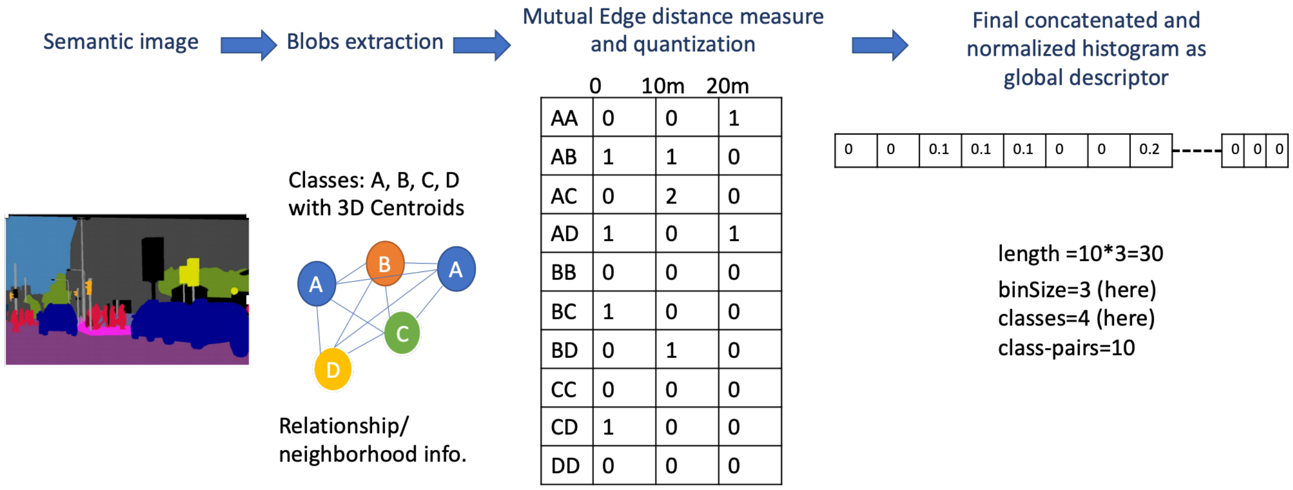


Fig. 1: Proposed loop closure detection, which consists of prepossessing and querying modules. First, location is searched using semantic-geometric descriptors, d_i^{GSG}, and then image is searched within the best matched location using ORB descriptor D_i^{Local}.



	0	10m	20m
AA	0	0	1
AB	1	1	0
AC	0	2	0
AD	1	0	1
BB	0	0	0
BC	1	0	0
BD	0	1	0
CC	0	0	0
CD	1	0	0
DD	0	0	0

Fig. 2: A dummy example of extracting global semantic-geometric descriptor from semantic image.

range of distances within a bin is large. In our current work, 'K' is chosen empirically as discussed in the experiments section.

B. Global Locations: Creation, Search and Update

We describe a location as a group of images which have similar semantic-geometric descriptors. This similarity can be measured in vector space of the descriptor, but for simplicity we use Euclidean distance measure for our experiments. Since each location must be generated on-the-fly (i.e., for each incoming image), using traditional clustering methods that processes on a set of samples will incur large delay and storage requirements. Furthermore, by using such clustering, the clusters can undergo significant changes in cluster centres, and hence the data-points (images) may need to be dynamically moved to other clusters. In our case, we require each image to be associated with a fix cluster to maintain a consistent hierarchical database of vocabulary trees. Otherwise, if images shift from their clusters, the associated local vocabularies need to be updated frequently which will increase the system complexity. Hence, we propose to use a simple but effective mean-shift based grouping as described below. The method is shown in Fig. 3 and Algorithm 1.

1) Creation and Update: A location l_j is represented by two mean descriptors i.e., initial mean descriptor vector $d_j^{init\_mean}$ and representative mean descriptor vector $d_j^{repr\_mean}$. A new location $l_k = \{d_k^{repr\_mean}, d_k^{init\_mean}\}$ is created when a new image I_i with descriptor d_i^{GSG} arrives that does not match with any of the existing locations (or there are no existing locations). The location descriptors are initialized as $d^{repr\_mean} = d^{init\_mean} = d_i^{GSG}$. The initial mean descriptor is updated only until first 'M=5' member images are added and is then fixed thereafter. The representative mean descriptor of a best matching location (l_{best_loc}) is updated throughout its lifetime as new image (I_i) is added to the location; which is the mean of all member

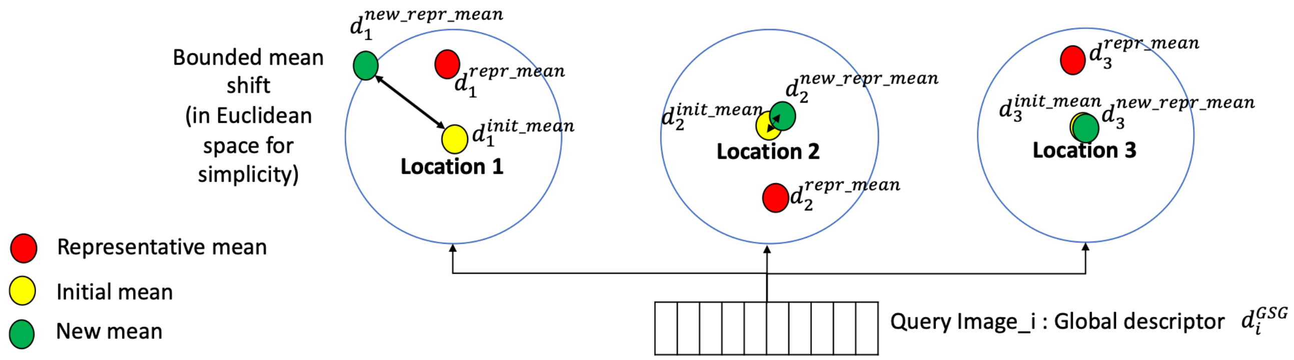


Fig. 3: Location matching and creation.

Algorithm 1 Search location: find matching location for d_i^{GSG}

```
initialize $\Delta_{min} = \infty$
for all $l_j \in L$ do
    $\Delta_{i,j} = ||d_i^{GSG} - d_j^{repr\_mean}||$
    if $\Delta_{i,j} < \tau_{max\_dist}$ then
        $d_j^{new\_repr\_mean} = (d_j^{repr\_mean} * N_j + d_i^{GSG})/(N_j + 1)$
        $\Delta(d_j^{new\_repr\_mean}, d_j^{init\_mean}) = ||d_i^{new\_repr\_mean} - d_j^{repr\_mean}||$
        if $\Delta(d_j^{new\_repr\_mean}, d_j^{init\_mean}) < \tau_{max\_shift}$ and $\Delta(d_j^{new\_repr\_mean}, d_j^{init\_mean}) < \Delta_{min}$ then
            $best\_loc = j$
            $\Delta_{min} = \Delta(d_j^{new\_repr\_mean}, d_j^{init\_mean})$
        end if
    end if
end for
```

Algorithm 2 Create or update location: if no location matches then create new location, otherwise update matched location.

```
if $best\_loc \neq \emptyset$ then
    $d_{best\_loc}^{repr\_mean} = d_{best\_loc}^{new\_repr\_mean}$
    $N_{best\_loc} = N_{best\_loc} + 1$
    if $N_{best\_loc} < 5$ then
        $d_{best\_loc}^{init\_mean} = d_{best\_loc}^{repr\_mean}$
    end if
else
    $d^{repr\_mean} = d^{init\_mean} = d_i^{GSG}$
    $l_k = \{d^{repr\_mean}, d^{init\_mean}\}$
    $L = L \cup l_k$
    $best\_loc = k$
    $N_{best\_loc} = N_{best\_loc} + 1$
end if
```

descriptors, i.e.,

$$d_{best\_loc}^{repr\_mean} = (d_{best\_loc}^{repr\_mean} * N_{best\_loc} + d_i^{GSG})/(N_{best\_loc} + 1) \tag{1}$$

where N_{best_loc} is the number of members in the l_{best_loc} before the update and d_i^{GSG} is the descriptor of I_i. The idea of having two (i.e., initial and representative) mean descriptors is to allow bounded shift in the cluster (location) centre (i.e., representative mean) as shown in Fig. 3. This ensures that the location or cluster centre cannot deviate far from its initial formation. This avoids the formation of large clusters with high intra-variance. The location search process to find the best location (*best_loc*) is described as follows and listed in Algorithm 1.

2) Search: For each new image I_i, the distance ($\Delta_{i,j}$) of its descriptor d_i^{GSG} is calculated with each existing location $l_j \in L$ as:

$$\Delta_{i,j} = ||d_i^{GSG} - d_j^{repr\_mean}|| \tag{2}$$

If the distance $\Delta_{i,j}$ is less than predefined threshold τ_{max_dist}, then the shift in the representative mean descriptor with respect to the initial mean descriptor is calculated.

$$d_j^{new\_repr\_mean} = (d_j^{repr\_mean} * N_j + d_i^{GSG})/(N_j + 1) \tag{3}$$

$$\Delta(d_j^{new\_repr\_mean}, d_j^{init\_mean}) = ||d_i^{new\_repr\_mean} - d_j^{repr\_mean}|| \tag{4}$$

The lowest $\Delta(d_j^{new\_repr\_mean}, d_j^{init\_mean})$ within allowable shift τ_{max_shift} is chosen as the *best_loc*. If there is no matching location, a new location is created. Comparisons with only one descriptor per location is required to match an incoming new image to a global location, which requires very low computation. Before adding a new image to this location, the shift in the representative mean descriptor with respect to the initial mean descriptor is compared (Fig. 3). If this mean-shift is greater than a threshold, the corresponding location is considered unmatched. By performing mean-shift based grouping of images, distinct clusters are maintained without shifting the cluster centre too much. This also results in computational savings compared to traditional clustering.

C. Local Class-tree

Each location maintains its own local BOW trees which are updated as more images are added to the location. We use the OBINDEX library [7] to create and update the incremental BOW vocabulary [5]. Moreover, as semantics provide class labels of the features in the image, we utilize this information as extra layer of nodes at the top-level of the hierarchical trees to introduce class-wise weighting of the words, and to query only the features from the class that is present in the scene, while disregarding the occluded classes. The query image consists of N feature descriptors (ORB) and each descriptor is searched only in its respective class tree, as shown in Fig. 4. Finally, image match is retrieved using TF-IDF score to obtain the maximum weighted match. In this work, we empirically assign significantly lower weights to sky class and dynamic classes (person, car, etc.), hence minimizing the impact of dynamic objects in place recognition.

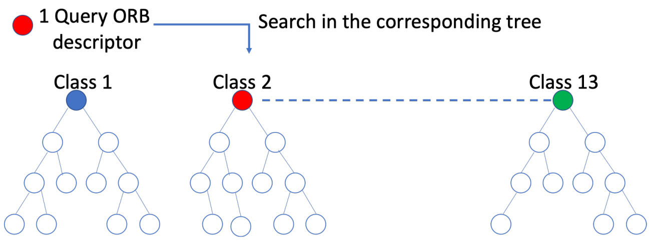


Fig. 4: Class-wise location trees.

D. Image Matching

Algorithm 3 Image Matching: search D_i^{Local} in corresponding class-trees of the best location and update trees

```
for all $j \in D_i^{Local}$ do
    search $D_{i,j}^{Local}$ in $Class\_Tree_{best\_loc}$ to get representative $Word_j$
    $Words = Words \cup Word_j$
end for
$Image\_Matches = TF\_IDF\_Voting(Words)$
$I_{best\_match} = Islanding(Image\_Matches)$
geometric verification of $I_{best\_match}$
update $BOW\_CLASS\_TREE_{best\_loc}$ and $TF\_IDF$
```

Once the best matching location has been retrieved as discussed in Section II-B and the current image has been added to the best location, the best location match is then used to query the local descriptors D_i^{Local}, as given in Algorithm 3. If there are no location matches, a new location is created. For the matched location, the corresponding local descriptors from D_i^{Local} list are searched in the corresponding BOW class-tree (Section II-C). The dynamic classes are not given zero weight in this work, therefore we do not search descriptors belonging to these classes, but other weighting strategy also can be employed. The probable image matches are weighted and retrieved using TF-IDF inverse index of that location. The matches are then grouped into islands similar to [5]. We use islanding method instead of discrete Bayes filter to reduce the computation time. The best image match found using the islands approach is then geometrically verified using the inliers count. And finally the $BOW\_CLASS\_TREE_{best\_loc}$ is updated using incremental learning [7] with class-specific tree branching.

TABLE I: Parameter settings used in proposed approach.

Parameters	Value
Number of ORB descriptors, N	1000
bin size, K	8
bin range, R	8
max distance to location, τ_{max_dist}	4
maximum mean shift, τ_{max_shift}	2

III. Experiments

In this section we evaluate the proposed loop detection approach against state-of-the-art methods in terms of performance and runtime.

A. Experimental Setup

1) Datasets: In this work, we try to overcome the challenges faced by existing loop closure detection methods (place recognition in general). These challenges include scalability, speed, dynamic objects, and viewpoint variations, which are prevalent in autonomous driving systems. Therefore, we select one of the most popular public datasets used for localization i.e., KITTI [8]. KITTI provides 11 video sequences, out of which 5 sequences (00, 02, 05, 06 and 07) contains loop. KITTI contains moderate number of dynamic objects including car and pedestrians. For proof of concept, we also include another public dataset, i.e., synthetic Synthia [13], where sequence 04 contains loop. To test the robustness of the proposed method against large number of dynamic objects, we also include CBD dataset [16]. The proposed method requires depth values to estimate the 3D coordinates, and hence we use datasets that provide depth information.

2) State-of-the-art Baselines: We compare the proposed method with state-of-the-art loop closure detection methods, FABMAP 2.0, SeqSLAM, iBOW-LCD and HTMap. These methods encapsulate variety of methods that are currently used in visual SLAM system. FABMAP 2.0 is the latest version of FABMAP [2], that is based on pre-trained BOW vocabulary. SeqSLAM [12] uses sequence of images to improve the loop detection accuracy. iBOW-LCD learns the vocabulary incrementally and therefore doesn't require any training stage. Similarly, HTMap uses incremental learning. However, HTMap employs a hierarchical approach to reduce the search space for place recognition and maintain high accuracy when creating a map. There have been several other works focusing on various aspects of loop detection and place recognition, but in this work we primarily focus on visual SLAM systems that require real-time operation. Apart from the speed requirements, we also tackle the challenges posed by dynamic objects in the scene on loop closure detection.

3) Implementation Details: All the experiments are performed on Intel® Xeon(R) CPU E5-1630 v4 @ 3.70GHz × 8. The baselines are run using their default settings. Similar to iBOW-LCD, the proposed method runs on single core and OBIndex [7] uses multiple cores. The main parameter settings used in our method are given in Table I.

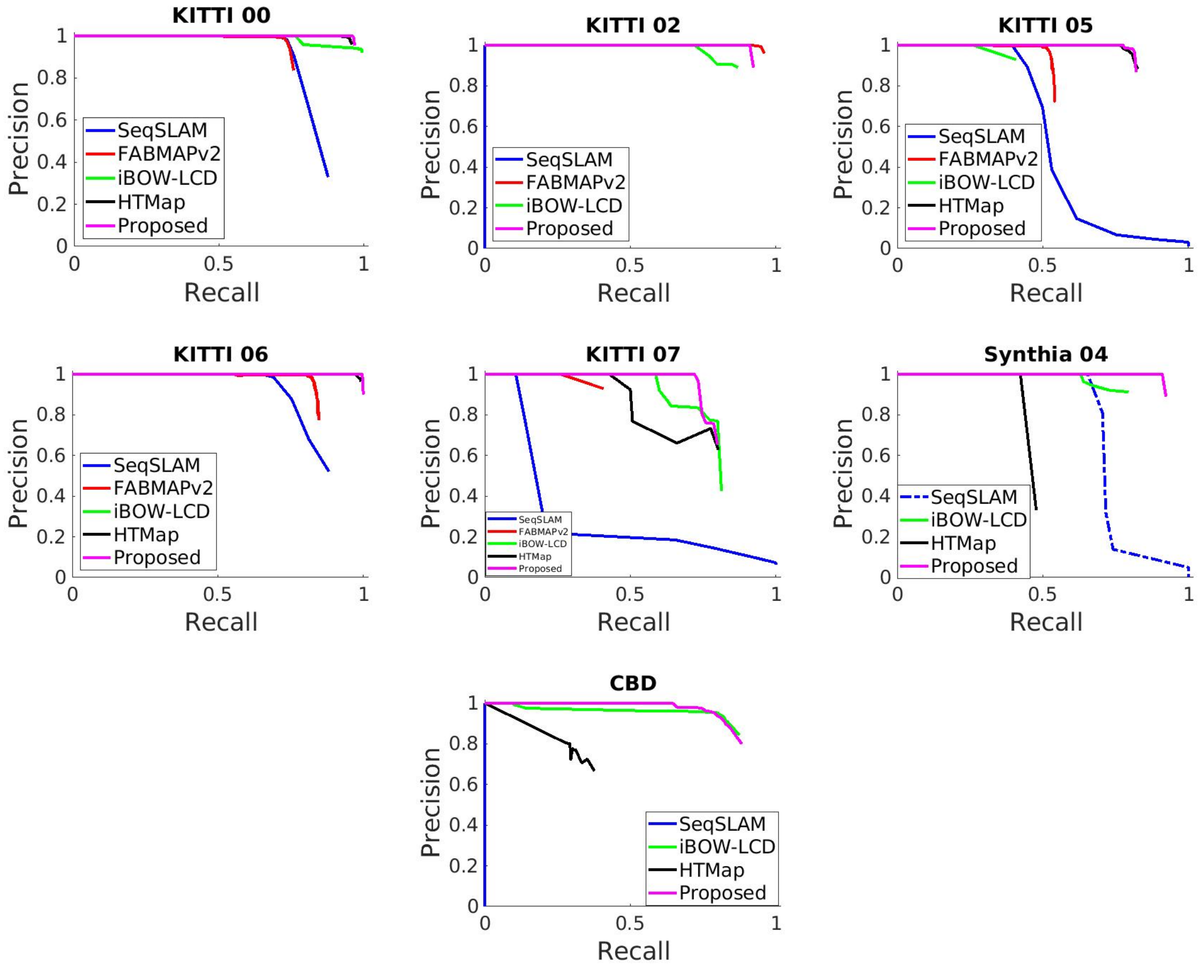


Fig. 5: Precision-recall curves for FABMAP2, SeqSLAM, iBOW-LCD, HTMap, and proposed method for various datasets. Only methods with valid loop detection results are shown.

TABLE II: Recall rates at 100% precision are given, missing value 'x' indicates the method did not achieve 100% precision. The proposed method is indicated as 'Ours'.

Sequence	FAB-MAP2	seqSLAM	iBOW-LCD	HTMap	Ours
KITTI 00	49.21	67.04	76.36	90.62	95.64
KITTI 02	90.95	x	71.96	x	91.20
KITTI 05	32.15	35.93	25.91	75.88	77.58
KITTI 06	55.34	64.68	92.19	97.03	99.63
KITTI 07	23.64	10.67	58.67	42.67	72.00
SYNTHIA 04	x	65.38	62.82	42.31	91.03
CBD stereo	x	x	9.83	x	64.47

B. Performance Evaluation

The loop detection method accuracy is defined as the maximum recall at 100% precision. The accuracy comparison of the proposed method with FAB-MAP2, SeqSLAM, iBOW-LCD, and HTMap is given in Table II. All the experimental results are obtained by running the open source code of these methods in their default configuration provided by the authors. The 'x' in the table implies that the particular method's precision on the specific sequence cannot achieve 100% precision and therefore, has inferior performance. The best accuracy among all methods for each sequence is also highlighted. It can be observed that the proposed method obtains highest accuracy among all methods for all sequences. HTMap is the second best performing method on KITTI 00, KITTI 05, and KITTI 06 sequences after the proposed method. On the highly dynamic sequence CBD, the proposed method outperforms all other methods by achieving 64.47 recall at 100% precision. In this sequence, several parts of the image is occluded by dynamic objects during the loop closure detection, which makes it very challenging to detect the matching places. The precision-recall (PR) curves are shown in Fig. 5.

TABLE III: Runtime comparison of loop closure detection of proposed method with FABMAP2, SeqSLAM, HTMap and iBOW-LCD.

Method	Avg. total time millisec	Avg. query time millisec
FAB-MAP2	226.04	-
SeqSLAM	160.5	-
HTMap	106.8 (feat. extr. +query)	87
iBOW-LCD	203 (feat.extr. + query)	180
Proposed	71.7 (feat.extr. + query) + 69.7 (semantic extraction)	45

C. Runtime Analysis

In this subsection, we discuss the runtime of the proposed method to query images to find matching location. The runtime of the proposed method is compared with the baselines in Table III. It can be observed that SeqSLAM is faster than FABMAP2, but HTMap is still faster requiring 106.8ms for feature extraction and query process. On the other hand, the proposed method takes only 71.7 ms for feature extraction and querying. If we take the runtime of semantic segmentation into account (i.e., 69.7ms with EdgeNet), then the total time increases to 141.4ms. However, we argue that in modern semantic visual SLAM systems, the feature extraction and semantic image are already available, therefore only the query time of loop closure detection is relevant in our evaluation. Hence, we also compare the query time of HTMap and iBOW-LCD with the proposed method. Compared to HTMap (87ms) and iBOW-LCD (180ms), the proposed method only takes 45ms for the querying process. This improvement in querying speed is due to the proposed hierarchical approach that reduces the search space at the location level and the computation reduction at the class-tree. The proposed method therefore increases the scalability of the system through the use of coarse locations that are grouped based on their semantic-geometric structures to reduce the query search space. The Class-tree further reduces the search time through class-based branching with significantly lesser descriptor comparisons.

IV. Conclusion

In this work, we propose to use global semantic-geometric descriptors for high-level place categorization which is inspired by how humans perceive places. As the metric information is necessary for precise localization, we also use local feature description of the places. The proposed hierarchical loop closure detection uses the high level semantic-geometry information to narrow down the search space and remove false matches, hence reducing the query time. The semantic locations are created and maintained dynamically. The semantic Class-wise tree BOW is used to obtain higher descriptiveness within each class of features. This leads to improvement in loop closure detection accuracy. The dynamic outliers due to the presence of moving objects are handled by the proposed weighting strategy for class-trees. The effectiveness of this method is demonstrated in datasets with highly dynamic sequences. Moreover, the proposed method does not require a pre-training stage for vocabulary compared to FABMAP and SeqSLAM. This enables long-term operations (words get updated and removed) where the agents will likely encounter many unseen places. Finally, the proposed loop closure detection method does not pose any notable overhead in modern semantic visual SLAM systems as it uses the features and semantic information that are readily available.

Acknowledgment

This research was conducted in part at Singtel Cognitive and Artificial Intelligence Lab for Enterprises (SCALE@NTU), which is a collaboration between Singapore Telecommunications Limited (Singtel) and Nanyang Technological University (NTU) that is supported by A*STAR under its Industry Alignment Fund (LOA Award number: I1701E0013).

References

[1] S. L. Bowman et al. "Probabilistic data association for semantic SLAM". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 1722–1729. DOI: 10.1109/ICRA.2017.7989203.

[2] Mark Cummins and Paul Newman. "Appearance-only SLAM at large scale with FAB-MAP 2.0". In: *The International Journal of Robotics Research* 30.9 (2011), pp. 1100–1123. DOI: 10.1177/0278364910385483.

[3] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha. "Visual simultaneous localization and mapping: a survey". In: *Artificial intelligence review* 43.1 (2015), pp. 55–81.

[4] Dorian Gálvez-López and J. D. Tardós. "Bags of Binary Words for Fast Place Recognition in Image Sequences". In: *IEEE Transactions on Robotics* 28.5 (Oct. 2012), pp. 1188–1197. ISSN: 1552-3098. DOI: 10.1109/TRO.2012.2197158.

[5] E. Garcia-Fidalgo and A. Ortiz. "iBoW-LCD: An Appearance-Based Loop-Closure Detection Approach Using Incremental Bags of Binary Words". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3051–3057. DOI: 10.1109/LRA.2018.2849609.

[6] Emilio Garcia-Fidalgo and Alberto Ortiz. "Hierarchical Place Recognition for Topological Mapping". In: *IEEE Transactions on Robotics* 33.5 (Oct. 2017), pp. 1061–1074. DOI: 10.1109/TRO.2017.2704598.

[7] Emilio Garcia-Fidalgo and Alberto Ortiz. "On the use of binary feature descriptors for loop closure detection". In: *Emerging Technology and Factory Automation (ETFA), 2014 IEEE*. Sept. 2014, pp. 1–8. DOI: 10.1109/ETFA.2014.7005121.

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[9] Spyros Gidaris and Nikos Komodakis. "LocNet: Improving Localization Accuracy for Object Detection". In: *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*. 2016.

[10] A. J. Glover et al. "FAB-MAP + RatSLAM: Appearance-based SLAM for multiple times of day". In: *2010 IEEE International Conference on Robotics and Automation*. 2010, pp. 3507–3512. DOI: 10.1109/ROBOT.2010.5509547.

[11] S. Lowry et al. "Visual Place Recognition: A Survey". In: *IEEE Transactions on Robotics* 32.1 (2016), pp. 1–19. DOI: 10.1109/TRO.2015.2496823.

[12] M. J. Milford and G. F. Wyeth. "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights". In: *2012 IEEE International Conference on Robotics and Automation*. 2012, pp. 1643–1649. DOI: 10.1109/ICRA.2012.6224623.

[13] G. Ros et al. "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 3234–3243. DOI: 10.1109/CVPR.2016.352.

[14] Gaurav Singh, Meiqing Wu, and Siew-Kei Lam. "Fusing Semantics and Motion State Detection for Robust Visual SLAM". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Mar. 2020.

[15] Mikaela Angelina Uy and Gim Hee Lee. "PointNetVLAD: Deep Point Cloud Based Retrieval for Large-Scale Place Recognition". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[16] G. Zhang, X. Yan, and Y. Ye. "Loop Closure Detection Via Maximization of Mutual Information". In: *IEEE Access* 7 (2019), pp. 124217–124232. DOI: 10.1109/ACCESS.2019.2937967.