

iBoW-LCD: An Appearance-based Loop Closure Detection Approach using Incremental Bags of Binary Words*

Emilio Garcia-Fidalgo and Alberto Ortiz

Abstract—In this paper, we introduce iBoW-LCD, a novel appearance-based loop closure detection method. The presented approach makes use of an incremental Bag-of-Words (BoW) scheme based on binary descriptors to retrieve previously seen similar images, avoiding any vocabulary training stage usually required by classic BoW models. In addition, to detect loop closures, iBoW-LCD builds on the concept of *dynamic islands*, a simple but effective mechanism to group similar images close in time, which reduces the computational times typically associated to Bayesian frameworks. Our approach is validated using several indoor and outdoor public datasets, taken under different environmental conditions, achieving a high accuracy and outperforming other state-of-the-art solutions.

I. INTRODUCTION

One of the most important aspects of Simultaneous Localization and Mapping (SLAM) [1] is to correctly manage the perceived information from the environment. Irrespective of the kind of sensor involved, there always intervene unavoidable noise sources that produce inaccurate measurements, leading to inconsistent representations when only raw sensor data is considered. For this reason, SLAM algorithms usually rely on *loop closure detection* mechanisms, which entail the correct identification of previously visited places. A robust loop closure detection scheme leads to additional constraints for the map generation process, resulting into more consistent representations. Although a variety of sensors have been used for loop closure detection, in the last decades, a high number of visual solutions have emerged, specially motivated by the low cost of cameras, the increase in computing power and the richness of the sensor data provided. Using a camera as the main source of information to undertake the association problem is generically known as *appearance-based* loop closure detection [2]–[9].

The performance of an appearance-based loop closure detection algorithm is highly influenced by the method used to describe the input images and the ability to retrieve previous images similar to the current one. Regarding image description, recent binary descriptors, such as BRIEF [10], ORB [11], LDB [12] or AKAZE [13], are progressively replacing the classical real-valued descriptors like SIFT [14] or SURF [15], given their reduced storage needs and computational times. As for the next issue, image indexing, the Bag of Words (BoW) model [16], [17] has proven to be an effective solution, specially when used in combination with an inverted index. In this model, the set of detected local features is quantized according

to a set of representative features called *visual words*, which conform a *visual vocabulary*, from which a histogram of visual word occurrences can be derived as the image descriptor. This visual vocabulary is typically generated off line. As a main limitation, these approaches need a training phase, which, depending on the number of descriptors (required to be high for an adequate performance) and the clustering technique used, can take a long time. Furthermore, this visual vocabulary is intended to be useful for generic scenarios, perhaps with a different appearance with regard to the training set, what can lead to additional false detections [18]. In these cases, the vocabulary can be regenerated using images taken from the current environment, at the expense of a priori knowledge and more computation time. An alternative to cope with these issues is to build the dictionary in an incremental manner.

This paper proposes a novel and effective method for visual loop closure detection oriented to view/place recognition called iBoW-LCD (*Incremental Bag-of-Words Loop Closure Detection*). Our approach adopts an incremental Bag of Binary Words strategy, which is able to build a visual vocabulary in an on-line manner, avoiding the drawbacks of off-line approaches. This scheme, used in combination with an inverted index, is employed to efficiently retrieve previously seen images. A robust loop closure detection method is proposed next. It extends and enhances the concept of *island* [5] in order to avoid images taken from the same place to compete among them as loop closure candidates. iBoW-LCD is validated using different public indoor and outdoor datasets and compares favourably against several state-of-the-art solutions, outperforming them in several ways.

Regarding the BoW strategy, our previous works [6], [19]–[21] adopted a purely incremental approach, where the visual words were never forgotten (removed) but increased in number as new images were processed. In this work, we consider not only adding but also deleting words as they are not deemed useful by an optimized version of the incremental BoW approach. This results in similar performance (as shown by the experimental results) with significantly less visual words. Regarding loop closure, our previous works were mainly based on Bayes filtering, which usually exhibits increasing processing times as more images are considered by the filter. In this work, we replace that filter by a simpler but effective mechanism to visually close loops on the basis of the novel concept of *dynamic island*, obtaining similar performance but reducing processing times. Unlike in [6], the method presented in this paper is not based on the FLANN implementation of the Muja’s algorithm [22] and it has been developed from scratch.

The rest of the paper is organized as follows: Section II overviews most important works in the field; Section III introduces the new incremental Bag of Binary Words scheme;

This work was supported by the EU H2020 project ROBINS (GA 779776).

All authors are with the Department of Mathematics and Computer Science, University of the Balearic Islands, 07122 Palma, Spain. {emilio.garcia, alberto.ortiz}@uib.es.

* Copyright © IEEE 2018 All rights reserved. IEEE Robotics and Automation Letters (2018). Digital Object Identifier (DOI): 10.1109/LRA.2018.2849609.

Section IV presents the loop closure detection approach; Section V reports on the results obtained; and, finally, Section VI concludes the paper and discusses topics for future research.

II. RELATED WORK

Most appearance-based loop closure detection solutions developed during the last years can be mainly classified according to the method used to describe the input images [23]. In this respect, some authors have opted for using a holistic approach to compute a global descriptor of the image. This kind of descriptors are usually fast to compute, but less tolerant to illumination and view-point changes, what reduces their discriminative capabilities. In order to alleviate this effect, loop closure techniques based on global descriptors tend to match sequences instead of single images [24]–[26]. This has been proven to be more robust against appearance changes, but renouncing to other desirable properties to detect loops, such as rotation invariance.

In this line, CNN-based solutions [27]–[30] have recently emerged as effective against environmental changes. As a pioneering work, Sünderhauf et al. [27] evaluated the utility of ConvNets for place recognition. In [28], they combined an object proposal technique with CNN features to match places over extreme appearance changes. Arroyo et al. [29] proposed a method where they fused the information from different convolutional layers to perform topological localization. In a recent work, Arandjelovic et al. [30] introduced a CNN architecture mainly based on a layer inspired in the VLAD image representation for weakly supervised place recognition. Despite the good performance shown by this kind of solutions, they are still disconnected from real SLAM and loop closure detection problems, as stated in [9].

The BoW model [16], [17] is, by far, the most used technique for appearance-based loop closure, given its demonstrated efficiency for retrieving previous similar images. Solutions based on this scheme can be mainly classified as off-line and on-line, depending on the nature of the vocabulary building process. Key works that fall into the off-line category are the FAB-MAP algorithm [2] and its extension FAB-MAP 2.0 [3], where a Chow-Liu tree was used to approximate the probabilities of visual word co-occurrences. Gálvez-López and Tardós [5] trained a visual vocabulary based on BRIEF [10], promoting the use of binary descriptors for place recognition tasks. Using this vocabulary as a basis, they introduced a loop closure detection method based on the concept of *islands* to group similar images close in time. The authors prevent images with a similar appearance to compete among them as loop closure candidates splitting the image sequence into fixed-size intervals. The algorithm establishes a relationship between the query image and each island according to a global score, computed as the sum of the individual scores of each image belonging to the island. In this work, we extend this idea by adapting the generation of the islands to the operating environment, allowing islands of different and dynamic sizes, as will be shown later. Mur-Artal and Tardós [31] enhanced their original algorithm [5] by using ORB [11], more robust against scale and rotation changes. A more recent work [9]

proposes an extension of the BoW model that groups visual words with similar optical flow when observed along two consecutive frames. These groups are called Structure-Aware and Viewpoint-Invariant High-Order Visual-Words (SVHVs). They naturally include the environment structure into the image description. All the methods surveyed so far require a training stage. In this work, we want to address the problem from a different point of view, by building the dictionary in an on-line manner.

Several on-line BoW attempts can be found in the literature [4], [6]–[8], [18], [32]. Among them, the work by Angeli et al. [4], which proposes a loop closure method based on an incremental BoW scheme [33] and a Bayesian filtering framework, can be considered of high importance in the field. Other on-line approaches involve RTAB-Map [32] and OVV [18], although these approaches are based on real-valued descriptors. Recently, an incremental BoW scheme based on binary descriptors called IBuILD [7] was introduced. This work describes a method to construct a visual dictionary in an on-line manner, aiming at loop closure detection. However, as stated by the authors, their approach does not employ an indexing scheme for an efficient search of words, which affects the scalability of the algorithm. The approach proposed in our paper features a hierarchical and incremental structure for such purpose, reducing the complexity during the BoW assignment process. In a more recent solution, Zhang et al. [8] proposed a technique for learning a visual word from a pair of matched features along two consecutive frames. The learned descriptor has perspective invariance to camera motion. This technique is finally integrated into the IBuILD algorithm, which, as mentioned before, lacks of a hierarchical structure to efficiently search for visual words.

III. INCREMENTAL BOW FOR IMAGE INDEXING

In order to manage an increasing number of visual words, an efficient indexing scheme is required, since a linear search becomes infeasible. Normally, this problem is solved using hierarchical structures such as kd-trees [34] or hierarchical k-means trees [22], but these methods are not suitable for binary descriptors because they expect that the descriptor components can be continuously averaged. Instead, hashing techniques [35], [36] can be used for matching binary descriptors. In this respect, Muja and Lowe introduced in [37] a novel method that achieves better performance than hashing approaches. Furthermore, their method involves a hierarchical tree which is a perfect structure for adding and deleting descriptors, as it is required in our case. In this work, we extend this method to be used as an incremental visual dictionary, as explained in the following sections.

A. Overview of Muja's Approach

Muja and Lowe introduced an effective hierarchical structure [37] to index and match binary features, which requires less storage space and scales better than other hashing methods. This structure consists in a tree where non-leaf nodes contain cluster centres and leaf nodes store visual descriptors to be matched. The visual words of the incremental vocabulary

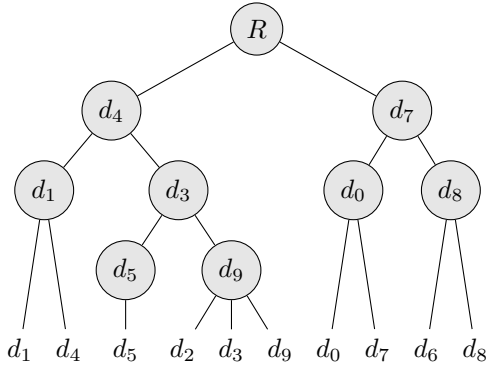


Fig. 1. A simple example of a hierarchical tree built by means of the Muja’s approach [37] to index 10 visual descriptors (d_0, \dots, d_9) using as parameters $K = 2$ and $S = 3$. Labels in non-leaf nodes (grey circles) indicate the descriptor selected randomly as the cluster centre during the building process.

are hence stored in the leaf nodes. To build one of these trees, the algorithm randomly selects, from the initial set of points, K descriptors as cluster centres. Next, each remaining input descriptor is assigned to its closest cluster centre according to their Hamming distance. This process is repeated recursively until the number of descriptors within a cluster is below a certain threshold S . The authors also demonstrated that building several trees T_i and using them in parallel during the search led to higher performance. An example of a tree built using this process is shown in Fig. 1.

In order to search descriptors in parallel using several trees, the algorithm starts with a single traverse of each tree from the root until reaching a leaf node, selecting at each step the node closest to the query descriptor and adding the unexplored nodes to a priority queue. When a leaf node is reached, all the points within this node are linearly searched. After exploring each tree once, the search continues from the closest node stored in the priority queue. The process finishes when a certain amount of descriptors has been examined (64 in our experiments).

B. Visual Vocabulary Update

Muja’s scheme was originally devised to index a static set of descriptors. Given that in our approach we handle an incremental visual dictionary, several modifications over the original approach have been introduced. First of all, binary descriptors are matched and combined during navigation to update the visual words of the vocabulary by means of a merging policy. As in our previous works [6], [20], we make use of a bitwise AND operation, which experimentally provides us better recognition performance than a bitwise OR. Formally stated:

$$B_{w_i}^t = B_{w_i}^{t-1} \wedge B_q, \quad (1)$$

where $B_{w_i}^{t-1}$ is the binary descriptor associated to the visual word w_i at time $t - 1$, B_q is a query descriptor and $B_{w_i}^t$ is the descriptor associated to the visual word w_i after the fusion of descriptors. In [20], we report on several experiments that demonstrate that this policy does not end up into degenerated descriptors (e.g. almost all bits set to zero).

Secondly, descriptors without a match in the index are incorporated into the dictionary as new visual words. To this

Algorithm 1 Adding a descriptor as a new visual word

Input: T : Hierarchical tree, B : Binary descriptor

- 1: $node \leftarrow \text{searchDescriptor}(T, B)$
- 2: **if** $\text{numDescriptors}(node) + 1 < S$ **then**
- 3: $\text{appendDescriptorToNode}(node, B)$
- 4: **else**
- 5: $D \leftarrow \text{getDescriptors}(node)$
- 6: $D = D \cup B$
- 7: $\text{buildNodeRecursively}(node, D)$

Algorithm 2 Deleting a visual word

Input: T : Hierarchical tree, B : Binary descriptor

- 1: $node \leftarrow \text{getNodeOfDescriptor}(T, B)$
- 2: $\text{deleteDescriptor}(node, B)$
- 3: **if** $\text{numDescriptors}(node) > 0$ **then**
- 4: **if** $B == \text{getClusterCentre}(node)$ **then**
- 5: $\text{selectNewClusterCentre}(node)$
- 6: **else**
- 7: $node_r \leftarrow \text{getRootNode}(node)$
- 8: $\text{deleteChildNode}(node_r, node)$
- 9: $\text{deleteNodesRecursively}(node_r)$

end, each descriptor is searched from the root until reaching a leaf node. Next, we assess if adding the corresponding new descriptor to the selected leaf node exceeds the maximum leaf size S . If that is the case, the node is recursively rebuilt adding the query descriptor to the original descriptor set. Otherwise, the descriptor is simply appended to the leaf node. Algorithm 1 illustrates this process.

Third, we maintain an inverted index. It stores, for each visual word, a list of images where it was found. Initially, the visual dictionary is created with the binary descriptors of the first image as a set of visual words. When an input image is processed, its extracted descriptors are matched against the visual words of the index applying the ratio test [14]. Matched descriptors are merged with their corresponding visual words using Eq. 1. Unmatched descriptors are added as temporary visual words to the vocabulary. In order to reduce the complexity of the index, these temporal visual words survive only if, after several consecutive frames P_f , they have been observed (e.g. matched) at least a certain number of times P_o . The inverted index is updated accordingly. The main purpose of this policy is to determine visual words that are most likely to be observed again if the agent returns to a previous location.

Lastly, we have provided the vocabulary with a mechanism to delete visual words to support the update policy outlined above. After deleting a descriptor from the dictionary, the node where it was appended and its ancestors are recursively revised to assess if they still contain children nodes. A node without any children node is no longer required, and therefore, deleted. If the deleted descriptor coincides with the cluster centre, a new centre is randomly selected. The process is summarized in Alg. 2.

C. Retrieval of Similar Images

The approach introduced in this section is used to efficiently retrieve previous images which are similar to the current

image. The inverted index allows us to efficiently compare a query image only with those images that share some visual words with it. A similarity score $s(I_t, I_k)$ is initialized to 0 for all possible k previously seen frames. Being z_t the set of binary descriptors extracted from the current image I_t , we search each descriptor of z_t in the dictionary to find the closest visual word. Next, we obtain, from the inverted index, the list of images where this visual word has appeared, and add a weight to the score s corresponding to each of the retrieved images. This weight is related to the term frequency-inverse document frequency (tf-idf [16], [20]) scoring, which reflects the importance of a visual word with regard to the visual vocabulary and the current image. After processing all descriptors in z_t , the ordered list of scores s is returned as the images most similar to I_t . The source code of this image indexing method is available to the community¹ as *OBIndex2* (Online Binary Index 2).

IV. LOOP CLOSURE DETECTION

This section details iBoW-LCD, a novel loop closure detection approach which makes use of the aforementioned *OBIndex2*. The source code is also available on line².

A. Searching for Previous Images

Given an image I_t at time stamp t , the process starts querying the image index, as explained in Sec. III-C. A buffer is used to store the most recent p images, and hence delay their publication as loop closure candidates. As a result of the search, the list of the j most similar images $C_t = \{I_{s_1}, \dots, I_{s_j}\}$, ordered by their associated scores $s(I_t, I_k)$, is obtained. The range of these scores highly depends on the distribution of the visual words and varies between even consecutive and similar images. Therefore, they are normalized, using a min-max technique, as follows:

$$\tilde{s}(I_t, I_k) = \frac{s(I_t, I_k) - s(I_t, I_{s_1})}{s(I_t, I_{s_j}) - s(I_t, I_{s_1})}, \quad (2)$$

where $s(I_t, I_{s_1})$ and $s(I_t, I_{s_j})$ are, respectively, the minimum and maximum scores obtained from the image search. This normalization step maps the scores to the range $[0, 1]$. Next, we discard those images whose normalized score \tilde{s} is below a predefined threshold τ_{im} , generating the final ordered list of image matches $\tilde{C}_t \subseteq C_t$. Note that this threshold determines the number of candidates. Setting τ_{im} to a low value results quite convenient since, in this way, there are still enough candidates but the worst choices can be discarded.

B. Dynamic Islands Computation

In previous works [6], [20], we relied on discrete Bayes filters to detect loop closures. As it is well known, these techniques lead to increasing computational times as more images are processed, specially due to the cost of calculating the transition model. To overcome this problem, iBoW-LCD introduces the concept of *dynamic island*, as an extension

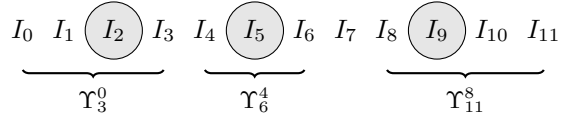


Fig. 2. A simple example comprising 3 islands (Υ_3^0 , Υ_6^4 , Υ_{11}^8) of different sizes, resulting from a sequence of 12 images (I_0, \dots, I_{11}). Note that I_7 does not belong to any island. Grey circles denote the island's representative, which is the image with the highest score \tilde{s} and hence the island's origin.

of the idea of *island* [5] that locally adapts the size of the group of images. The innovation against the original concept of island is twofold: on the one hand, iBoW-LCD does not compute islands using all the previous images but makes use of a filtered set of similar images \tilde{C}_t resulting from the previous step; on the other hand, the size of the islands is not fixed but depends on the similarities between neighbouring images and the camera velocity, what adapts the resulting islands to the image stream.

In this work, an *island* is defined as a group of similar images whose timestamps lie between two different instants. This criterion allows us to group images close in time and avoid them from competing to each other as loop candidates. We denote Υ_n^m as the island which groups the images whose timestamps are between m and n . Additionally, there always exists a representative image for each island, which corresponds to the image with the largest score \tilde{s} within the range $[m, n]$. To manage the set of islands, images in the list \tilde{C}_t are considered sequentially: for each image $I_c \in \tilde{C}_t$, we assess if its timestamp lies within the time interval of an existing island Υ_n^m ; if this is the case, the image is associated to Υ_n^m and the time interval of Υ_n^m is updated to accommodate I_c and the b previous and posterior frames; otherwise, an island is created, with a predefined initial size $2b + 1$ around time c , and I_c is associated to the new island. After processing all the images in \tilde{C}_t , the limits of the resulting islands are revised and truncated, if needed, in order to obtain a disjoint set, avoiding time overlaps among islands. Figure 2 illustrates the concept through a simple example. For each island, a global score G is computed as:

$$G(\Upsilon_n^m) = \frac{\sum_{i=m}^n \tilde{s}(I_t, I_i)}{m - n + 1}, \quad (3)$$

which is the average of normalized scores of the images associated to the island. Finally, the resulting list of islands Γ_t is sorted according to their global score G . The full process of building islands is summarized in Alg. 3.

C. Island Selection

At this step, iBoW-LCD selects the best matching island $\Upsilon^*(t) \in \Gamma_t$. To this end, it recalls the best island at the previous timestamp $t-1$, $\Upsilon^*(t-1)$, and checks whether any of the islands $\Upsilon_n^m \in \Gamma_t$ overlaps with $\Upsilon^*(t-1)$. The overlapping islands are named *priority islands* inspired by the observation that consecutive images should close loops with areas of the environment where previous images closed a loop, if any. If priority islands are found, the one with the highest global score G is selected for the next step. Otherwise, iBoW-LCD chooses

¹<http://github.com/emiliofidalgo/obindex2>

²<http://github.com/emiliofidalgo/ibow-lcd>

Algorithm 3 Building dynamic islands

Input: \tilde{C} : Ordered list of similar images**Output:** Γ_t : Ordered list of islands at time t .

```
1:  $\Gamma_t \leftarrow \square$ 
2: for each image  $I_c$  in  $\tilde{C}$  do
3:   found  $\leftarrow$  false
4:   for each island  $\Upsilon_n^m$  in  $\Gamma_t$  do
5:     if  $m < c < n$  and not found then
6:       associateToIsland( $I_c, \Upsilon_n^m$ )
7:       changeIslandSize( $\Upsilon_n^m, c, b$ )
8:       found  $\leftarrow$  true
9:   if not found then
10:     $\Upsilon_{c+b}^{c-b} \leftarrow$  createNewIsland( $I_c, b$ )
11:     $\Gamma_t = \Gamma_t \cup \Upsilon_{c+b}^{c-b}$ 
12:  $\Gamma_t \leftarrow$  obtainDisjointIslands( $\Gamma_t$ )
13: for each island  $\Upsilon_n^m$  in  $\Gamma_t$  do
14:    $G(\Upsilon_n^m) \leftarrow$  computeIslandScore( $\Upsilon_n^m$ )
15: sort( $\Gamma_t$ )
```

the first island from the current set Γ_t , i.e. the one with the largest score G , which turns out to be the island most similar to the current image.

D. Loop Closure Decision

This stage of iBoW-LCD chooses first the representative of the selected island as the final loop closure candidate I_f . An epipolarity analysis is performed next for I_t and I_f to validate whether they can come from the same scene after a camera rotation and/or translation. To this end, we compute a set of putative matchings between I_t and I_f using the ratio test [14] and find, using RANSAC, the inliers resulting from imposing the fundamental matrix model to the set of feature matchings. The loop hypothesis is accepted only if the number of inliers is high enough.

Note that, instead of this geometrical check, a temporal coherency technique could be applied here, like in [5], [7], to reduce the computational requirements of the approach. However, this last method tends to reduce the recall values. This option could be considered when using iBoW-LCD in a real SLAM system, where achieving high recall values are not essential and several correct loops are enough to ensure coherent maps.

iBoW-LCD also tracks the number of consecutive loops occurred at time t in order to avoid the computation of the fundamental matrix on every image and speed up the process: the algorithm accepts a loop, without performing the epipolarity analysis, if a priority island is found and the number of consecutive loops at time t is higher than a threshold τ_c .

V. EXPERIMENTAL RESULTS

This section evaluates the proposed approach and compares it against several state-of-the-art solutions. An Intel Core i7-6500U (2.5Ghz) / 12 GB RAM computer was used in all experiments. OBIndex2 made use of four cores to perform a search in four trees at the same time, while iBoW-LCD employed only a single core.

TABLE I
PARAMETERS VALUES AND SECTION WHERE THEY ARE DEFINED.

K (Sec. III-A)	16	P_f (Sec. III-B)	2
S (Sec. III-A)	150	P_o (Sec. III-B)	2
T_i (Sec. III-A)	4	τ_{im} (Sec. IV-A)	0.3
p (Sec. IV-A)	50	b (Sec. IV-B)	5
Features per image	1000	τ_c (Sec. IV-D)	20

A. Methodology

As usual in these cases, the evaluation is performed in terms of precision-recall metrics. Along with the curves, we are particularly interested in the maximum recall that can be achieved at 100% precision, what implies no false positive detections missing a minimum number of loops. Avoiding these false positives becomes essential when the algorithm is employed in a full SLAM system, given that they can produce inconsistencies in the resulting maps. The following public datasets, taken under different visual conditions, have been considered for the evaluation: City Centre [2] (CC), New College [2] (NC), Lip6 Indoor [4] (L6I), Lip6 Outdoor [4] (L6O), KITTI 00 [38] (K00) and KITTI 06 [38] (K06). For benchmarking purposes, we use the ground truth provided by the original authors of each method except for the KITTI sequences, where the files provided by [26] are employed as a reference. This last ground truth was created manually by the authors, labelling as long stops the time intervals where the vehicle was not in motion.

B. Algorithm Configuration

In order to find a suitable set of parameters for iBoW-LCD, we initially executed the algorithm against the City Centre dataset several times, modifying the parameters until obtaining the best possible recall at 100% precision. The resulting parameters, which are enumerated in Table I, have then been used for the remaining experiments. Furthermore, a collection of ORB interest points [11] have been detected and described for each image. Note, however, that our algorithm is descriptor-agnostic and that any other binary descriptor could be used instead.

C. General Performance

As a measure of general performance of iBoW-LCD, we have computed for all datasets including CC the precision-recall curves shown in Fig. 3, resulting from modifying the threshold on the number of inliers required to accept a loop (Sec. IV-D). As can be seen, iBoW-LCD works reasonably well in all cases, achieving high recall rates while keeping precision at 100%. From the curves, it can be observed that the approach exhibits a very stable behaviour especially for the L6O and K06 datasets, where precision decreases minimally as recall values increase. This behaviour repeats for all datasets, even under viewpoint and illumination changes. We hypothesize that deleting unstable descriptors, as done by iBoW-LCD, favours keeping more stable visual words in the dictionary, improving the general system tolerance as for the aforementioned appearance changes.

Next, we chose the largest dataset considered in this work (K00) to analyse the evolution of the visual dictionary size

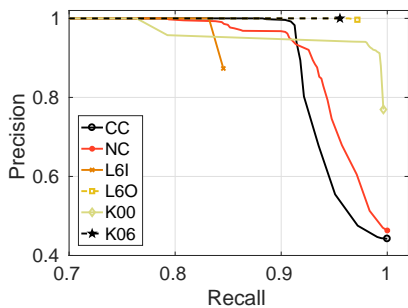


Fig. 3. Precision-recall (P-R) curves. ($P = 1$ for $R < 0.75$ in all cases.)

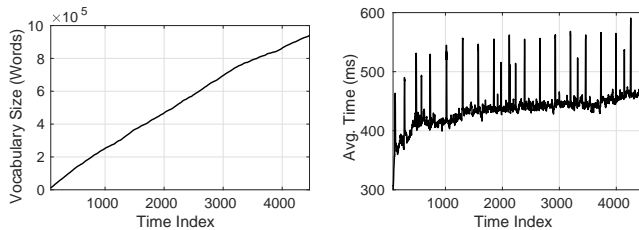


Fig. 4. Performance metrics computed over the K00 dataset. (left) Vocabulary size with regard to the number of images processed. (right) Average response time per image with regard to the number of images processed. Peaks are mainly due to the rearrangement of the visual words.

and the corresponding average response time per image, which includes all the stages of the algorithm (visual word handling, image query, islands computation and loop closure decision). The results obtained can be found in Fig. 4. In spite of the fact that the number of visual words grows as more images are processed, the average response time remains more stable, exhibiting a moderate increment. Note that this growth is highly related to the trajectory performed by the vehicle: the more similar areas are revisited, the less new visual words are added, since more visual words are matched against the visual dictionary. On average, the time required to process a frame of the K00 dataset is 432.38 ms. Times could be slightly higher than the ones presented by some off-line solutions, especially due to the delay required to manage visual words. To alleviate this, several improvements could be incorporated, such as reducing the number of features per image or applying a temporal consistency check instead of performing an epipolarity analysis. In this work, we have prioritized high recall values against computational times.

D. Comparison with Other Solutions

This section compares iBoW-LCD against other state-of-the-art solutions. First of all, given that the proposed approach is an evolution of one of our previous works [6], we want to check whether the modifications proposed here represent a real improvement in terms of response time and recall. In this regard, Table II summarizes the final vocabulary size (VS), the maximum recall at 100% of precision (R) and the average response time per image (T) obtained for each approach and dataset. As can be observed, the impact in terms of recall is minimum and, in general, quite similar. However, iBoW-LCD is able to process an image in less time using a more reduced set of visual words in contrast to our previous solution. We believe that this fact is mainly due to the new visual word managing process and the simplification of the loop closure

TABLE II
COMPARISON WITH OUR PREVIOUS APPROACH.

	Previous [6]			iBoW-LCD		
	VS	R (%)	T (ms)	VS	R (%)	T (ms)
CC	1.6M	88.24	503.65	95K	88.25	368.41
NC	1.3M	53.15	489.24	98K	79.40	352.08
L6I	30K	79.09	24.93	4K	83.18	19.17
L6O	826K	97.51	304.01	121K	85.24	249.45
K00	4.7M	78.73	546.21	958K	76.50	432.38
K06	1.1M	84.76	480.49	212K	95.53	395.16

TABLE III
COMPARISON OF MAXIMUM RECALL AT 100% PRECISION.

	CC	NC	L6I	L6O	K00	K06
Cummins [3]	38.50	51.91	n.a.	n.a.	49.21	55.34
Angeli [4]	n.a.	n.a.	36.86	23.59	n.a.	n.a.
Milford [24]	68.98	49.39	n.a.	n.a.	67.04	64.68
Khan [7]	38.92	n.a.	n.a.	n.a.	n.a.	n.a.
Gálvez-López [5]	30.61	55.92	n.a.	n.a.	n.a.	n.a.
Mur-Artal [31]	43.03	70.29	n.a.	n.a.	n.a.	n.a.
Bampis [9]	52.36	74.60	42.32	49.55	n.a.	n.a.
Zhang [8]	41.18	59.20	n.a.	n.a.	n.a.	n.a.
Stumm [39]	38.00	39.00	n.a.	n.a.	n.a.	n.a.
Cieslewski [40]	n.a.	n.a.	n.a.	n.a.	≈60.00	n.a.
iBoW-LCD	88.25	79.40	83.18	85.24	76.50	95.53

scheme. Notice the high reduction of the final vocabulary size in comparison with our previous approach. The variability in the obtained recall values can be attributed to the high dependence of the method on the distribution of the visual words. As shown in Table II, deleting visual words does not always imply higher recall values, but always reduces computational times and the size of the final visual vocabulary.

Table III compares the maximum recall achieved by our approach at 100% precision in contrast to other state-of-the-art solutions. The results reported are taken from the original papers, except the ones corresponding to Cummins [3] and Milford [24] which were obtained by ourselves in a previous work [20]. The term *n.a.* means that the corresponding information is not available from any source. From this table, one can observe that the increase in processing times is compensated in terms of accuracy, since iBoW-LCD achieves a higher recall in all datasets considered. This enhancement is specially evident in the CC dataset, where it is usually difficult to attain high recall at 100% of precision.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we have introduced iBoW-LCD, an appearance-based loop closure detection algorithm, which mainly relies on an incremental Bag of Binary Words scheme to retrieve previous similar images. This incremental visual dictionary builds on a hierarchical structure to efficiently search, insert and delete new visual words on line, avoiding the main drawbacks that off-line approaches present. Next, iBoW-LCD makes use of a novel concept to group similar images close in time called *dynamic island*, which naturally exploits the nature of image sequences to detect loop closures. The proposed method has been validated using several public datasets, obtaining competitive results in comparison with other state-of-the-art solutions.

Referring to future work, we will consider to extend the methods developed in this paper to a hierarchical loop closure scheme, given the good results obtained in this matter in one of our previous publications [20]. We will investigate other appearance-based methods to group images. To further favour the long-term operation of the method, a mechanism to mitigate the growth of the dictionary (e.g. based on response time) could be useful. Additionally, we also plan to enhance the response time of iBoW-LCD parallelizing some of their stages. Finally, we want to incorporate our solution into a complete SLAM / 3D reconstruction framework.

REFERENCES

- [1] H. Durrant-Whyte and T. Bailey, "Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms," *IEEE Robot. Autom. Mag.*, vol. 2, pp. 99–110, 2006.
- [2] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *Int. J. Rob. Res.*, vol. 27, no. 6, pp. 647–665, 2008.
- [3] —, "Appearance-Only SLAM at Large Scale with FAB-MAP 2.0," *Int. J. Rob. Res.*, vol. 30, no. 9, pp. 1100–1123, 2011.
- [4] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer, "A Fast and Incremental Method for Loop-Closure Detection Using Bags of Visual Words," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 1027–1037, 2008.
- [5] D. Galvez-Lopez and J. Tardos, "Bags of Binary Words for Fast Place Recognition in Image Sequences," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1188–1197, 2012.
- [6] E. Garcia-Fidalgo and A. Ortiz, "On the Use of Binary Feature Descriptors for Loop Closure Detection," in *IEEE Emerg. Tech. Fact. Autom.*, 2014, pp. 1–8.
- [7] S. Khan and D. Wollherr, "iBuILD: Incremental Bag of Binary Words for Appearance-Based Loop Closure Detection," in *IEEE Int. Conf. Robot. Autom.*, 2015, pp. 5441–5447.
- [8] G. Zhang, M. J. Lilly, and P. A. Vela, "Learning Binary Features Online from Motion Dynamics for Incremental Loop-Closure Detection and Place Recognition," in *IEEE Int. Conf. Robot. Autom.*, 2016, pp. 765–772.
- [9] L. Bampis, A. Amanatiadis, and A. Gasteratos, "High Order Visual Words for Structure-Aware and Viewpoint-Invariant Loop Closure Detection," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 4268–4275.
- [10] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," in *Eur. Conf. Comput. Vision*, ser. Lecture Notes in Computer Science, vol. 6314, 2010, pp. 778–792.
- [11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An Efficient Alternative to SIFT or SURF," in *IEEE Int. Conf. Comput. Vision*, vol. 95, 2011, pp. 2564–2571.
- [12] X. Yang and K.-T. Cheng, "Local Difference Binary for Ultrafast and Distinctive Feature Description," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 1, pp. 188–94, 2014.
- [13] P. F. Alcantarilla and T. Solutions, "Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1281–1298, 2011.
- [14] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Key-points," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [15] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," in *Eur. Conf. Comput. Vision*, ser. Lecture Notes in Computer Science, vol. 3951, 2006, pp. 404–417.
- [16] J. Sivic and A. Zisserman, "Video Google: A Text Retrieval Approach to Object Matching in Videos," in *IEEE Int. Conf. Comput. Vision*, 2003, pp. 1470–1477.
- [17] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," in *IEEE Conf. Comput. Vision Pattern Recog.*, vol. 2, 2006, pp. 2161–2168.
- [18] T. Nicosevici and R. Garcia, "Automatic Visual Bag-of-Words for Online Robot Navigation and Mapping," *IEEE Trans. Robot.*, vol. 28, no. 4, pp. 886–898, 2012.
- [19] E. Garcia-Fidalgo, A. Ortiz, F. Bonnin-Pascual, and J. P. Company, "Fast Image Mosaicing using Incremental Bags of Binary Words," in *IEEE Int. Conf. Robot. Autom.*, 2016, pp. 1174–1180.
- [20] E. Garcia-Fidalgo and A. Ortiz, "Hierarchical Place Recognition for Topological Mapping," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1061–1074, 2017.
- [21] —, *Methods for Appearance-based Loop Closure Detection: Applications to Topological Mapping and Image Mosaicing*, ser. Springer Tracts in Advanced Robotics. Springer, 2018, vol. 122.
- [22] M. Muja and D. G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," *Int. Conf. Comput. Vis. Theory and Apps.*, vol. 2, no. 331–340, p. 2, 2009.
- [23] E. Garcia-Fidalgo and A. Ortiz, "Vision-based Topological Mapping and Localization Methods: A Survey," *Rob. Auton. Syst.*, vol. 64, pp. 1–20, 2015.
- [24] M. Milford and G. Wyeth, "SeqSLAM: Visual Route-Based Navigation for Sunny Summer Days and Stormy Winter Nights," in *IEEE Int. Conf. Robot. Autom.*, 2012, pp. 1643–1649.
- [25] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, J. Yebes, and S. Gamez, "Bidirectional Loop Closure Detection on Panoramas for Visual Navigation," in *IEEE Intell. Veh. Symp.*, 2014, pp. 1378–1383.
- [26] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, J. J. Yebes, and S. Bronte, "Fast and Effective Visual Place Recognition using Binary Codes and Disparity Information," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 3089–3094.
- [27] N. Sündnerhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford, "On the Performance of ConvNet Features for Place Recognition," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 4297–4304.
- [28] N. Sündnerhauf, S. Shirazi, A. Jacobson, F. Dayoub, E. Pepperell, B. Upcroft, and M. Milford, "Place Recognition with Convnet Landmarks: Viewpoint-Robust, Condition-Robust, Training-Free," in *Robot.: Sci. Syst.*, 2015.
- [29] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, and E. Romera, "Fusion and Binarization of CNN Features for Robust Topological Localization Across Seasons," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 4656–4663.
- [30] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN Architecture for Weakly Supervised Place Recognition," in *IEEE Conf. Comput. Vision Pattern Recog.*, 2016, pp. 5297–5307.
- [31] R. Mur-Artal and J. D. Tardos, "Fast Relocalisation and Loop Closing in Keyframe-Based SLAM," in *IEEE Int. Conf. Robot. Autom.*, 2014, pp. 846–853.
- [32] M. Labbe and F. Michaud, "Appearance-Based Loop Closure Detection for Online Large-Scale and Long-Term Operation," *IEEE Trans. Robot.*, vol. 29, no. 3, pp. 734–745, 2013.
- [33] D. Filliat, "A Visual Bag of Words Method for Interactive Qualitative Localization and Mapping," in *IEEE Int. Conf. Robot. Autom.*, 2007, pp. 3921–3926.
- [34] C. Silpa-Anan and R. Hartley, "Optimised KD-Trees for Fast Image Descriptor Matching," in *IEEE Conf. Comput. Vision Pattern Recog.*, 2008, pp. 1–8.
- [35] A. Gionis, P. Indyk, and R. Motwani, "Similarity Search in High Dimensions via Hashing," in *International Conference on Very Large Data Bases*, 1999, pp. 518–529.
- [36] R. Salakhutdinov and G. Hinton, "Semantic Hashing," *Int. J. of Appr. Reas.*, vol. 50, no. 7, 2009.
- [37] M. Muja and D. G. Lowe, "Fast Matching of Binary Features," in *Conf. Comput. and Rob. Vision*, 2012, pp. 404–410.
- [38] A. Geiger, P. Lenz, and R. Urtasun, "Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *IEEE Conf. Comput. Vision Pattern Recog.*, 2012, pp. 3354–3361.
- [39] E. S. Stumm, C. Mei, and S. Lacroix, "Building Location Models for Visual Place Recognition," *Int. J. Rob. Res.*, vol. 35, no. 4, pp. 334–356, 2016.
- [40] T. Cieslewski and D. Scaramuzza, "Efficient Decentralized Visual Place Recognition From Full-Image Descriptors," in *IEEE Int. Symp. on Multi-Robot and Multi-Agent Systems*, 2017, pp. 78–82.