Hierarchical Place Recognition for Topological Mapping

Emilio Garcia-Fidalgo and Alberto Ortiz, *Member, IEEE*

***Abstract*—In this paper, we propose a novel appearance-based approach for topological mapping based on a hierarchical decomposition of the environment. In our map, images with similar visual properties are grouped together in nodes, which are represented by means of an average global descriptor and an index of binary features based on a Bag-Of-Words online approach. Each image is represented by means of a global descriptor and a set of local features, and this information is used in a two level loop closure approach, where, first, global descriptors are employed to obtain the most likely nodes of the map and, then, binary image features are used to retrieve the most likely images inside these nodes. This hierarchical scheme enables us to reduce the search space when recognizing places maintaining a high accuracy when creating a map. Our approach is validated using several public datasets and compared against several state-of-the-art techniques. The accuracy and the sparsity of the generated maps are also discussed.**

***Index Terms*—Appearance-based localization, place recognition, topological mapping, loop closure, bag of binary words.**

I. Introduction

MAPPING is an essential problem in mobile robotics. The outcome of this process is a representation of the environment built using the information received through the sensors attached to the robot. This map is then used in other tasks such as localization, path-planning and obstacle avoidance and is of special interest for autonomous robotics, where the agents need to be able to operate without human interaction. Different kinds of devices, such as ultrasonic or laser sensors, have been used during years to construct these maps. However, in the last two decades, there has been a growing interest in visual solutions, because of the low cost of cameras and the richness of the sensor data provided.

As far as robotic mapping is concerned, two main paradigms are generally accepted: metric and topological mapping. Metric maps represent the world as accurately as possible, maintaining a lot of information about environment details, such as distances, measures or sizes, and they are usually referenced according to a global coordinate system. However, metric maps are more difficult to build and maintain, and are computationally demanding. Conversely, topological maps [1] represent the environment in an abstract manner by means of a graph, where nodes represent distinctive places in the environment and arcs model the relations between them. These maps are simple and compact, scale better and require much less space to be stored than metric maps.

Recent vision-based topological mapping approaches [2]–[9] are mainly based on loop closure detection, which entails the correct identification of previously visited places from sensor data. These solutions usually generate dense topological maps, where each input image is introduced as a new node in the map. In these cases, despite the use of different indexing techniques [10], this can be unfeasible for large environments. A hierarchical representation of the environment [11], where images which present a similar appearance are grouped together in nodes, can help in these cases, reducing the search space when finding similar places.

When using vision for mapping tasks, it is necessary to describe the acquired images and be able to compare the descriptions, being the quality of the map dependent on the description method. Most of the recent works that can be found in the literature make use of the Bag-of-Words (BoW) approach [10], where usually SIFT [12] or SURF [13] descriptors are quantized according to a reference visual dictionary, built from a training set in an offline step. However, binary features developed recently such as BRIEF [14], BRISK [15], ORB [16], FREAK [17] or LDB [18] can be also used in an online BoW scheme, avoiding the training step, as presented in our previous work [19]. Another group of solutions make use of global descriptors [20], [21], which are faster to compute but more sensitive to noise and illumination changes.

In this paper, we present a topological mapping framework, called Hierarchical Topological Mapping (HTMap), which is based on a novel hierarchical place recognition approach. In our approach we combine a global descriptor, which is used to obtain similar places to the current image, with binary local features, which are then used for achieving a more accurate place recognition. In more detail, in HTMap, images with similar visual properties are stored in *locations*. Each location is represented by means of a global descriptor, which summarizes the visual appearance of the images stored inside the node, and by an index of binary image features based on an online BoW scheme [19], which can be used to query the node using local image features.

As a main contribution of this paper, we introduce a robust hierarchical loop closure algorithm, which operates in a two-level approach. First, Pyramid Histogram of Oriented Gradients (PHOG) [21] global descriptors are calculated and employed for selecting the most similar locations to the current image, avoiding the need of searching in the whole map

Manuscript received XXX XX, 201X; revised XXX XX, 201X.

All authors are with the Department of Mathematics and Computer Science, University of the Balearic Islands, Palma de Mallorca, 07122 Spain (email: emilio.garcia@uib.es; alberto.ortiz@uib.es).

This work is partially supported by the ESF through grant FPI11-43123621R (Conselleria d'Educacio, Cultura i Universitats), by Govern de les Illes Balears project AAEE50/2015 and FEDER funding, by project EU FP7 INCASS (GA 605200) and by the Spanish project SUPERION (MINECO DPI2014-57746-C3-2-R).

and speeding up the retrieval process. Next, binary local features extracted from the current image are used to query the indices of the locations obtained at the previous step in order to find similar images inside the retrieved locations. The scores obtained at the two levels are combined as a likelihood function inside a Bayes filter to determine the most similar image to the current one. Two additional contributions are, on the one hand, a scalable method to construct topological maps which employs, as a key component, the above-mentioned hierarchical loop closure algorithm, and, on the other hand, to the best of our knowledge, the use for the first time of the PHOG descriptor for mapping and localization tasks. PHOG represents local image shape and its spatial layout, and was originally devised for image classification. Finally, we performed an extensive evaluation of our approach and a comparison with some state-of-the-art techniques, achieving better results in several public datasets.

The rest of the paper is organized as follows: Section II overviews fundamental works in the field, Section III introduces the image description techniques used in our approach and summarizes the corresponding indexing methods, Section IV explains the structure of the map generated by our approach, Section V details our topological mapping framework, Section VI reports on the results of the different experiments performed, and Section VII concludes the paper.

II. Related Work

Most of the vision-based topological mapping techniques developed during the last fifteen years can be mainly classified according to the description method used [22].

Global descriptors describe the image in a holistic manner, using the full image as input to the process. These descriptors generate a single description for the whole image and are usually fast to compute. However, they present less robustness to occlusion and illumination effects, what results into a lower discriminative power. Histograms provide a compact way of representing an image and have been used for topological mapping in different forms, e.g. colour histograms [23] or gradient orientation histograms [24]. Recently, several approaches based on the Gist descriptor [20] have emerged, specially applied to omnidirectional images [25]. Motivated by the success of Gist and the BRIEF binary feature descriptor [14], Sunderhauf and Protzel [26] adapted the former to be used as a global descriptor, introducing BRIEF-Gist. As a main drawback, BRIEF-Gist is not able to detect bidirectional loops. In this regard, Arroyo et al. [27], [28] introduced an algorithm called Able for Binary-appearance Loop-closure Evaluation applied to Panoramas (ABLE-P) which can deal with these cases. Another successful example of place recognition system using a global description approach is SeqSLAM [29], where instead of searching for a single previously seen image given the current frame, they performed the localization process recognizing coherent sequences of local consecutive images.

Several authors have used local features to perform topological mapping and localization tasks, specially since the release of the Lowe's Scale-Invariant Feature Transform (SIFT) algorithm [30]–[34]. Some researchers constructed hierarchical maps of the environment using local features in an offline step. That is the case of Zivkovic et al. [35], which presented a hierarchical representation that was later employed in [36], [37]. Unlike these works, the proposal presented here can deal with the construction of the map as new images arrive to the algorithm and relies on a full topological method instead of combining topological and metrical approaches. In a previous work [38] we proposed an appearance-based approach for visual mapping and localization based on local features.

The BoW algorithm [10], [39] is one of the most used techniques for appearance-based mapping and loop closure detection, since, in combination with an inverted index, it is a powerful tool to search for previous images. The idea is to quantize the detected local features (commonly SIFT or SURF) in the image according to a set of representative features, known as visual vocabulary, which can be generated offline or online. Probably the most well-known solution that generates the dictionary offline is Fast Appearance-Based Mapping (FAB-MAP) [3], [4], where the probabilities of visual word co-occurrences were approximated by means of a Chow-Liu tree. Later, an improved version called FAB-MAP 2.0 [5], [6] was presented adapting the probabilistic model to be used with an inverted index architecture similar to typical image search engines. This scheme was assessed using a dataset of 1000 km composed by omnidirectional images and GPS coordinates to be used as ground truth. Given that binary descriptors have been developed recently and, hence, it is a very recent research area, only a few attempts to create dictionaries from binary features can be found [9], [40].

A main shortcoming of generating the visual dictionary offline is the need of a training phase, which can take hours. Furthermore, this dictionary may not be representative of the scenario if the robot operates in a different environment. An alternative is to build the codebook online in an incremental manner, while the robot discovers the environment. In this regard, the work of Angeli et al. [7], [8], which employs the approach of Filliat [41] for generating visual dictionaries dynamically, can be considered of high importance in the field. Other approaches that fall into this category are Real-Time Appearance-Based Mapping (RTAB-Map) [42] and Online Visual Vocabulary (OVV) [43]. Given the benefits of generating the visual dictionary online and the use of binary features, recently we introduced a method for indexing binary features to build a dictionary online, already validated for loop closure detection in [19]. This method is used in this work for indexing images according to the binary local features detected.

Only a few of the solutions presented so far focus on generating a sparse representation of the environment. Regarding sparse topological mapping and hierarchical loop closure, there are some works which are briefly reviewed in the following. Maohai et al. [44] presented a hierarchical localization approach based on omnidirectional vision where, in a first step, colour histograms allow to select a subset of the images stored in the map. Next, SIFT local features are used to obtain a more accurate localization inside this subset. Unlike this work, in our approach, the map is built online, monocular images are used instead of omnidirectional images and binary features are employed instead of classical real-valued descriptors, such as SIFT or SURF. Other place

recognition solutions which have appeared recently are based on the BoW framework, adapting FAB-MAP to work as a hierarchical approach [45] or maintaining an inverted file per group of images or *environments* [46]. In these works, the visual dictionary is built offline through a training step, which we would like to avoid. Moreover, the work by Mohan et al. [46] deals with different image sequences, from different environments, and intends to localize the robot first within one of these environments and next find the view in the chosen environment most similar to the query; because of this, the authors qualify their solution as hierarchical. However, their method does not try to discover a hierarchy in the input data, contrary to our solution, which besides finds this hierarchy in an online manner, while the camera/vehicle is navigating. The framework presented by Korrapati et al. [11], [47], where Hierarchical Inverted Files (HIFs) are defined for indexing images, served us as inspiration. However, they are based on omnidirectional images and the offline BoW framework, which makes the training step unavoidable. Recently, Chen et al. have introduced several bio-inspired hierarchical place recognition approaches [48], [49].

III. Image Description

In HTMap, images are described using global and local features. When finding similar places, global descriptors allow us to obtain, in a fast way, a subset of the nodes in the map whose stored images can be similar to the current one. Then, local feature descriptors are used to select the most similar images inside the retrieved nodes. An image received at time t is described as $I_t = \{G_t, F_t\}$, where G_t is the computed global descriptor and F_t is the set of local features found in the image. In this section, the techniques used for computing G_t and F_t are detailed.

A. Global Feature Description

As a global representation, we use the Pyramid of Histograms of Orientation Gradients (PHOG) global descriptor [21], which was originally developed for image classification. Despite the fact that it is less effective for an accurate place recognition, this simple descriptor can help when summarizing image information inside a node, as will be shown later. PHOG represents an image by its local shape and the spatial layout of this shape. The local shape is represented by a histogram of edge orientations (HOG) [50] computed from an image subregion quantized into K bins, where the contribution of each edge is weighted according to its magnitude. The spatial layout is represented by L levels, dividing the image into a sequence of increasingly finer spatial grids by repeatedly doubling the number of divisions in each direction. The final PHOG descriptor is created by concatenating all the HOG descriptors and normalizing them to sum to unity. An example of PHOG computation is illustrated in Fig. 1. In our implementation, we have experimented with these parameters in order to maximize the recall maintaining 100% of precision, as will be shown in section VI. We set $K = 60$ and $L = 3$, which generates a descriptor of 1260 components. These values have been used in all experiments. Formally, the global descriptor

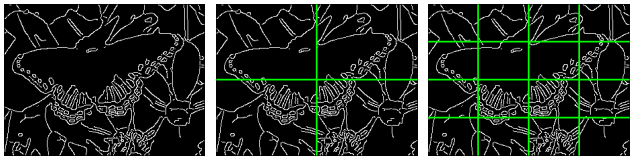


Figure 1. Example of PHOG descriptor computation: from left to right, grids for levels 0, 1 and 2; the final descriptor consists of a weighted concatenation of the histograms of oriented gradients for each grid cell.

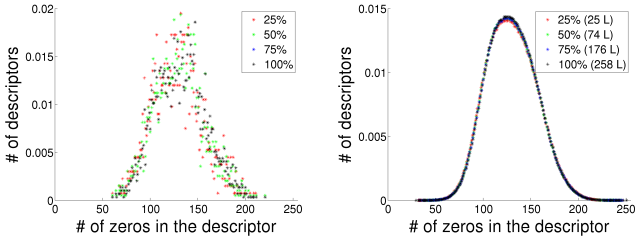


Figure 2. Histograms for the number of descriptors that contain a certain number of bits set to zero, for the St. Lucia dataset. The left histogram corresponds to location 18 once 25%, 50%, 75% and 100% of images have been processed. The right histogram is for all locations at the same percentages of the mapping process. (Bars have been removed to improve the visualization of the plots. In the right legend box, L stands for location.)

is defined as $G_t = \{g_0^t, g_1^t, \ldots, g_{1259}^t\}$. According to the original paper, the χ^2 distance exhibits a superior performance when comparing two of these descriptors. Given two PHOG descriptors, G_i and G_j, we define distance $d_g(G_i, G_j)$ as:

$$d_g(G_i, G_j) = \sum_{k=0}^{1259} \frac{(g_k^i - g_k^j)^2}{g_k^i + g_k^j}. \quad (1)$$

B. Local Feature Description

For each image, we also compute a collection of FAST corners [51] described each by an LDB binary descriptor [18]. This allows us to take advantage of their fast computation times and reduced storage needs, in front of classic approaches such as SIFT [12] or SURF [13]. LDB is a highly efficient, robust and distinctive binary descriptor, which performs in, basically, three steps. First, LDB captures the internal patterns of each image patch using a set of binary tests, comparing the average intensity and first-order gradients. Second, the structure is computed at different spatial granularities. Third, the algorithm selects a subset of the bits according to their distinctiveness to build the final binary descriptor.

The set of LDB descriptors of the n features found at image I_t is defined as $F_t = \{f_0^t, f_1^t, \ldots, f_{n-1}^t\}$. Their similarity is calculated in accordance to the Hamming distance $d_f(f_p^i, f_q^j)$ between two binary descriptors f_p^i and f_q^i from, respectively, sets F_i and F_j, i.e. $d_f(f_p^i, f_q^j) = \text{bitsum}(f_p^i \oplus f_q^j)$.

IV. Map Representation

Our map representation is based on the observation that the appearance of images taken from the same scene should look similar to one another. These images are grouped together in what we call *locations*. Hence, a location is a group of images of the environment that present some visual similarity. In order to manage the relationships between these locations, the environment is modeled by means of an undirected graph, whose nodes represent the locations in the map and edges represent

connectivities, i.e. traversability, between them. In formal terms, given $I = \{I_0, I_1, \ldots, I_{t-1}\}$ as the input sequence of images received up to time t, we define our topological map at t as $M_t = (\omega, \gamma)$, where ω is the set of existing locations and γ represents a set of edges which encodes the relationships between locations. The set of locations ω is defined as:

$$\omega = \{\ell_0, \ell_1, \ldots, \ell_{c-1}\}, \tag{2}$$

where ℓ_i represents the location i and c is the total number of locations. Particularly, the i-th location is defined as the tuple:

$$\ell_i = (\zeta_i, \rho_i, \beta_i), \tag{3}$$

where $\zeta_i = \{\iota_0, \iota_1, \ldots, \iota_{m-1}\}$ are the indices of the m images associated to the location, ρ_i is the representative of the location and β_i is an index of local features built from the images belonging to the location, which is used to retrieve images according to the local binary features found in the current image. When a new image I_t is added to the location ℓ_i, its index is added to ζ_i, and ρ_i and β_i are updated accordingly.

Given a query image, ρ_i is used to rapidly obtain a measure of similarity between I_t and the location, which is a key step in our hierarchical loop closure algorithm. This representative ρ_i is computed as the average PHOG descriptor of the images inside the location. Hence, it can be defined as $\rho_i = \{r_0^i, r_1^i, \ldots, r_{1259}^i\}$, where each component is computed as:

$$r_j^i = \frac{\sum_{k=0}^{m-1} g_j^{\iota_k}}{m}. \tag{4}$$

After retrieving the most similar locations, we employ local binary features to obtain similar images in these locations. In order to avoid image-to-image comparisons, we make use of the index of local features β_i. This index, inspired by image retrieval methods, is an efficient way to determine the similarity between the current image and the images stored in the location. Image retrieval methods developed recently are based on the BoW approach [10], where usually the dictionary is built offline. Furthermore, most of these BoW approaches are usually based on real-valued descriptors [12], [13] and is less common to find binary solutions [9]. To deal with the problems that these approaches present, in this work, we employ a method for computing a vocabulary of binary features that can be built online, avoiding thus a training phase. This method was formerly presented in [19] and so the interested reader is referred to the original paper for further details. A brief overview is provided next.

Our method is based on an incremental visual dictionary based on a modified version of Muja and Lowe's approach [52]. The dictionary is combined with an inverted index, which contains, for each word, a list of images where it was found. Since our approach relies on an incremental visual dictionary based on binary features, an updating policy for combining binary descriptors is needed. Averaging each component of the vector is an option for real-valued descriptors, but it is not for the binary case. We propose to use a bitwise AND operation. Formally, being B a binary descriptor:

$$B_{w_i}^t = B_{w_i}^{t-1} \wedge B_q, \tag{5}$$

where $B_{w_i}^{t-1}$ is the binary descriptor of the word w_i stored in the dictionary at time $t-1$, B_q is the query descriptor and $B_{w_i}^t$ is the merged descriptor for word w_i at time t. Given the way how we merge descriptors, and to determine whether this process does not end up into degenerated descriptors (e.g. almost all bits set to zero), we ran an experiment consisting in processing the longest sequence used in this work (see Table II) and then we analyse the final descriptors found in the indices. Note that this sequence produces more than 256-bit 10M descriptors, distributed among the binary indices of 258 locations. The results of the analysis can be found in Fig. 2, which shows histograms for the number of zeros in the different descriptors, i.e. bin k of the histogram accounts for the descriptors whose number of zeros are k. The plots show the histograms at a certain percentage of the mapping process for location 18 (left) and for all locations (right). As can be seen, the distribution of the number of zeros in the descriptors does not change relevantly along the processing of the sequence, despite the several merging operations performed between descriptors as loop closures are found.

The index is initially built using the descriptors of the first image of the location. When a new image needs to be added to the index, their descriptors are searched in the index. Given a query binary descriptor, we search for the two nearest neighbours traversing the tree from the root to the leafs and selecting at each level the node that minimizes the Hamming distance. Using these two neighbours, we apply the ratio test [12] in order to determine if both descriptors represent the same visual feature. If positive, the query descriptor and the visual word are merged using (5) and replaced in the dictionary. Otherwise, the query descriptor is considered a new feature and is added to the index as a new visual word. In both cases, the inverted index is updated accordingly, adding a reference to the current image in the list corresponding to the modified or added word. Given the features of a query image as input, the visual index returns an ordered list of images according to a scoring process.

Figure 3 illustrates an example of a map generated using our approach. Note that our hierarchical decomposition somehow favours long-term tasks. On the one hand, it speeds up the loop closure detection process by preventing an image search in the full map. On the other hand, locations can be serialized on disk and loaded on demand if the location is selected as a candidate, to save memory space. Only the representative of the location ρ_i needs to be kept in memory for it to be available for the first step of the loop closure detection algorithm.

V. Topological Mapping Framework

A. Algorithm Overview

HTMap builds a visual representation of the environment using a monocular camera, and localizes the robot within this map. Therefore, at time stamp $t-1$, there exists an *active* location $\ell_a \in \omega$, which can be defined as the current topological position of the robot within the map according to the received images up to time $t-1$. Given the next image I_t, our mapping algorithm tries to determine if there exist a similar location in the map or, otherwise, this image corresponds to an unexplored area of the environment.

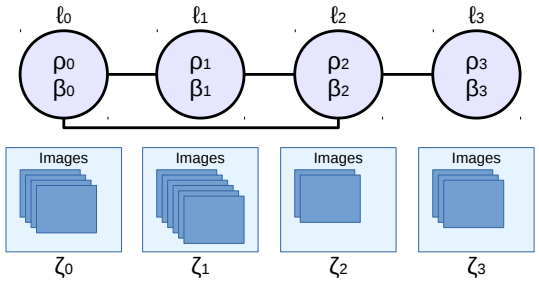


Figure 3. Example of a hierarchical map generated by our approach. The map comprises four locations ℓ_0, ℓ_1, ℓ_2 and ℓ_3 and a loop between ℓ_0 and ℓ_2. Besides the corresponding set of images ζ_i, each location i contains a representative ρ_i and an index of local features β_i.

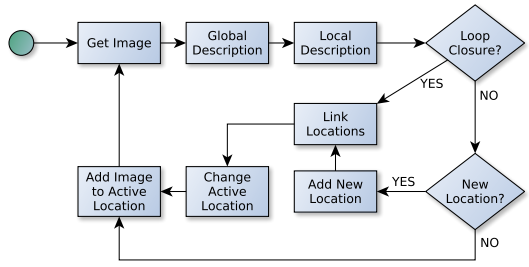


Figure 4. Overview of HTMap.

Figure 4 and Algorithm 1 illustrate our topological mapping approach. An initial location ℓ_0, marked as active, is created including only the first input image I_0. For each new image I_t, global G_t and local F_t descriptors are computed as explained in section III. These descriptors are then used in the loop closure step to determine if this image comes from an already known place. If positive and the retrieved location is different to the current active location ℓ_a, the locations are linked in the graph γ in order to register a relationship between these places and ℓ_a is updated to point to the loop location. Otherwise, a decision about whether this image belongs to ℓ_a or else is a new place must be made. If it can be considered as a new place, a location is added to the map, linked to the current location ℓ_a and marked as active. In the last step, I_t is associated to ℓ_a, and ρ_a and β_a are updated by means of, respectively, G_t and F_t.

The following sections detail the policy used to conclude whether an image belongs to the current active location or is a new location, as well as the loop closure detection algorithm.

B. New Location Policy

Once the current image has been found not to close a loop with any existing location, the dissimilarity between ℓ_a and I_t is evaluated in order to check if the image comes from the scene represented by the current location, i.e. if $d_g(\rho_a, G_t) < \tau_{nn}$, then I_t is associated to ℓ_a, being ρ_a the representative of the node and G_t the global descriptor of I_t. Threshold τ_{nn} plays a key role with regard to the sparsity of the map: the higher it is, the lower the number of nodes, but more images are associated to each location. We evaluate the quality and accuracy of the visual maps according to τ_{nn} in section VI.

Algorithm 1 Topological Mapping

procedure TOPOLOGICAL_MAPPING
  while there are images **do**
    I_t = get_image()
    G_t = global_description(I_t) ▷ PHOG extraction
    F_t = local_description(I_t) ▷ FAST and LDB
    if loop_closure(G_t, F_t, M_t) **then**
      ℓ_c = get_loop_closure_location()
      link(ℓ_a, ℓ_c, M_t)
      $\ell_a = \ell_c$ ▷ Updates the active location
    else
      if is_new_location(G_t, ℓ_a) **then**
        ℓ_n = create_new_location(M_t)
        link(ℓ_a, ℓ_n, M_t)
        $\ell_a = \ell_n$ ▷ Updates the active location
      else
        do_nothing() ▷ The image belongs to ℓ_a
      end if
    end if
    add_img_to_loc(I_t, ℓ_a) ▷ ρ_a and β_a are updated
  end while
end procedure

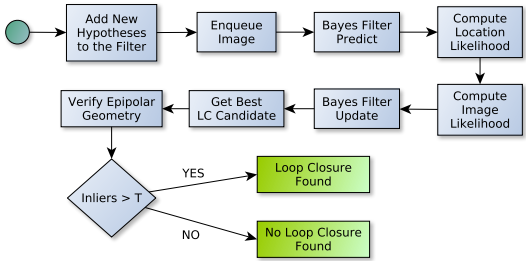


Figure 5. Hierarchical loop closure detection algorithm.

C. Hierarchical Loop Closure Detection

Our loop closure detection module is based on a discrete Bayes filter, which estimates the probability that the current image closes a loop with a previously seen image associated to an existing location of the map. As on other solutions, the Bayes filter allows us to deal with noisy measurements and

Algorithm 2 Hierarchical Loop Closure Detection

procedure LOOP_CLOSURE(G_t, F_t, M_t)
  add_hypotheses(t) ▷ Add valid hypotheses at time t
  enqueue_image(G_t, F_t)
  bayes_filter_predict()
  likelihood = compute_likelihood(G_t, F_t, M_t)
  bayes_filter_update(likelihood)
  c = get_best_candidate()
  ninliers = epipolarGeometry(F_t, F_c)
  if ninliers > τ_{ep} **then**
    return true ▷ Loop closure found
  else
    return false ▷ No loop closure found
  end if
end procedure

ensures temporal coherency between consecutive predictions, integrating past estimations over time.

Our approach is outlined in Fig. 5 and Algorithm 2. In order to avoid false loop closure detections with immediately previous images, the latter are not directly included as loop closure hypotheses as soon as they arrive. Instead of this, a buffer is used to store the most recent p images, delaying their publication as loop closure candidates. Consequently, the first step is to release the candidates that could be considered as possible loop closures at the current time step. After that, the current image is enqueued in the buffer and, then, the computation of the likelihood and the Bayes filter update steps are performed. An epipolarity analysis between the current image I_t and the image with the highest probability I_c is performed in order to validate if they can come from the same scene after a camera rotation and/or translation. Matchings that do not fulfill the epipolar constraint are discarded by means of RANSAC. If the number of inliers is above a threshold τ_{ep}, the loop closure hypothesis is accepted; otherwise, it is definitely rejected. Further details of the Bayes filter are detailed next.

1) Bayes filter derivation: The Bayes filter described below is based on our previous approach [38] after being adapted to be used within the hierarchical approach, since our map representation is very useful during the likelihood computation. Let L_i^t denote the event that image I_t closes a loop with image I_i, which is associated to some location ℓ_j at time t, where $i < t$. We also denote $O_t = \{G_t, F_t\}$ as the observation at time t, which comprises the global and local descriptions computed for the current image I_t. Using these definitions, we want to find the previous image $I_{\mathcal{C}}$ whose index $\mathcal{C}$ satisfies:

$$\mathcal{C} = \underset{i=0,\dots,t-p}{\arg\max} \left\{ P\left(L_i^t | O_{0:t}\right) \right\}, \tag{6}$$

where $P\left(L_i^t | O_{0:t}\right)$ is the full posterior probability at time t given all previous observations up to time t. As in [7] and as explained previously in this work, the most recent p images are not included as hypotheses in the computation of the posterior. This parameter p delays the publication of hypotheses and needs to be set according to the frame rate and the velocity of the camera. Separating the current observation from the previous ones, the posterior can be rewritten as:

$$P\left(L_i^t | O_{0:t}\right) = P\left(L_i^t | O_t, O_{0:t-1}\right), \tag{7}$$

and then, using conditional probability properties, we can isolate our final goal to obtain:

$$P\left(L_i^t | O_t, O_{0:t-1}\right) = \frac{P\left(O_t | L_i^t, O_{0:t-1}\right) P\left(L_i^t | O_{0:t-1}\right)}{P\left(O_t | O_{0:t-1}\right)}, \tag{8}$$

where $P\left(O_t | O_{0:t-1}\right)$ can be seen as a normalizing factor since its computation does not depend on L_i^t. Under this premise and the Markov assumption, the posterior is defined as:

$$P\left(L_i^t | O_{0:t}\right) = \eta P\left(O_t | L_i^t\right) P\left(L_i^t | O_{0:t-1}\right), \tag{9}$$

where η is the normalizing factor, $P\left(O_t | L_i^t\right)$ is an observation model and $P\left(L_i^t | O_{0:t-1}\right)$ is the prior, computed after a prediction step. The conditional probability $P\left(O_t | L_i^t\right)$ is considered as a likelihood function $\mathcal{L}\left(L_i^t | O_t\right)$, computed as explained in section V-C3. Notice that the Markov assumption, which has been successfully employed in other works, e.g. [7], is a simple approximation that allows us to avoid the computation of several high-order conditional dependencies between observations. Decomposing the right side of (9) using the Law of Total Probability, the full posterior can be written as:

$$P\left(L_i^t | O_{0:t}\right) = \eta P\left(O_t | L_i^t\right) \sum_{j=0}^{t-p} P\left(L_i^t | L_j^{t-1}\right) P\left(L_j^{t-1} | O_{0:t-1}\right), \tag{10}$$

where $P\left(L_j^{t-1} | O_{0:t-1}\right)$ is the posterior distribution computed at the previous time instant and $P\left(L_i^t | L_j^{t-1}\right)$ is the transition model. The observation model $P\left(O_t | L_i^t\right)$ is computed in two consecutive steps according to the observation pair O_t and using conditional probability properties:

$$P\left(O_t | L_i^t\right) = P\left(G_t, F_t | L_i^t\right) = P\left(G_t | L_i^t\right) P\left(F_t | L_i^t, G_t\right), \tag{11}$$

where $P\left(G_t | L_i^t\right)$ results from the similarity between G_t and the existing locations in the map, and $P\left(F_t | L_i^t, G_t\right)$ is computed searching similar images inside the retrieved locations.

2) Transition model: The loop closure probability at time t is predicted from the previous posterior according to an evolution model. The probability of loop closure with an image I_j at time $t-1$ is diffused over its neighbours following a discretized Gaussian-like function centered at j. In more detail, 90% of the total probability is distributed among j and exactly eight of its neighbours. The remaining 10% is shared uniformly across the rest of loop closure hypotheses according to $\frac{0.1}{\max\{0, t-p-9\}+1}$. This implies that there is always a small probability of jumping between hypotheses far away in time, improving the sensitivity of the filter when the robot revisits old places. Note that, unlike our previous version of the filter [38], the Gaussian comprises eight frames instead of four. This increases the sensitivity of the filter since the likelihood computation is limited to the images belonging to the locations which are similar enough to the current image I_t. Due to this reason, the use of a lower number of neighbours leads to a lower number of correct loop detections, while a higher number increases the execution time for this step. We empirically found 8 as a good compromise for our purposes.

3) Observation model: The current observation O_t is included in the filter once the prediction step has been performed. To this end, we make use of our hierarchical representation of the environment, which allows us to calculate the likelihood $\mathcal{L}\left(L_i^t | O_t\right)$ without the need of computing the similarity between I_t and all the previous images. This likelihood is calculated at two levels: first, global descriptors are used to obtain the locations most similar to the current image, what produces a similarity score for every location in the map. Then, local feature descriptors are searched only in the feature indices whose score is above a threshold, in order to obtain a similarity score regarding the images stored at those locations.

The goal of the first step is to obtain places in the map with a similar appearance to the current image I_t. To this end, the distance $d_g(\rho_i, G_t)$ between the global descriptor of the image G_t and the representative global feature of each location ρ_i, is computed. Next, these distances are converted into a similarity

Algorithm 3 Likelihood Computation

procedure COMPUTE_LIKELIHOOD(G_t, F_t, M_t)
  $d_g = []$
  for each location i in ω **do**
    $d_g[i]$ = compute_global_dist(ρ_i, G_t)
  end for
  d_{max} = get_max(d_g)
  d_{min} = get_min(d_g)
  $g = []$ ▷ Global scores
  $\omega^c = [\ell_a]$ ▷ Similar locations to the current image
  for each location i in ω **do**
    $g[i] = 1 - \frac{d_g[i] - d_{min}}{d_{max} - d_{min}}$
    if $g[i] > \tau_{llc}$ **then**
      $\omega^c = \omega^c \cup \ell_i$
    end if
  end for
  $s = []$ ▷ Combined scores
  for each location i in ω^c **do**
    for each image j in ℓ_i **do**
      l_j = sim_score(F_t, β_i)
      $s[j] = g[i] \cdot l_j$
    end for
  end for
  return s
end procedure

score g_i by means of equation (12):

$$g_i = 1 - \frac{d_g(\rho_i, G_t) - d_g^{min}}{d_g^{max} - d_g^{min}}\,, \tag{12}$$

where d_g^{min} and d_g^{max} are, respectively, the minimum and the maximum distances resulting for I_t. Despite that this normalization can lead to searching in more locations than the ones strictly necessary, the resulting images must still pass the second step of the algorithm. In order to select the most likely places given the current image, a set of locations ω', whose score is higher than a predefined threshold τ_{llc}, is defined as:

$$\omega' = \{\ell_i \in \omega \,|\, g_i > \tau_{llc}\}\,, \tag{13}$$

where ω was defined in Eq. 2. Note that, despite the number of locations increases as more images are processed, the time required to obtain location candidates is very low even with a high number of nodes, as will be seen in section VI. This is because the distance between PHOG descriptors can be calculated very fast. Therefore, we select locations with $g_i > \tau_{llc}$ irrespective of the number of candidates, which achieves better performance. The final set of candidate locations ω^c is obtained combining ω' with the currently active location ℓ_a:

$$\omega^c = \omega' \cup \ell_a\,. \tag{14}$$

The intuition behind the inclusion of the currently active location is that, if a loop was detected at the previous time with an image associated to ℓ_a, it is possible that the current image also closes a loop with an image in ℓ_a, due to the visual similarity between consecutive images. Hence, we update the likelihood of the images associated to ℓ_a irrespective of the score g_i resulting for the images belonging to ℓ_a.

Table I
DEFAULT PARAMETERS FOR HTMAP EXECUTION

Grid size	4×4	Nearest neighbour ratio	0.8
Max. keypoints per image	1000	τ_{nn}	0.15
FAST threshold	10	τ_{llc}	0.65
LDB descriptor length	32 bytes		

In a second step, image similarities are computed in order to find the most likely images in the selected locations. To this end, local binary descriptors of the current image are searched in the feature indices of the retrieved locations ω^c, and a similarity score $\wp_j$ is computed for every image j associated to each candidate location i:

$$\wp_j = \text{sim\_score}\left(F_t, \beta_i\right)\,, \forall I_j \in \ell_i, \forall \ell_i \in \omega^c\,, \tag{15}$$

which is a score based on the Term Frequency Inverse Document Frequency (TF-IDF) weighting factor [53]. Being $I_{0:k}$ the set of the images added to the index β_i up to the current time t, the TF-IDF value $\varrho_{w_r}^t$ computed given the word w_r and the image I_t is defined as:

$$\varrho_{w_r}^t = \frac{n_{w_r}^t}{N_t} \times \log \frac{k-1}{n_{w_r}}\,, \tag{16}$$

where $n_{w_r}^t$ is the number of occurrences of the word w_r in image I_t, N_t is the total number of features found in image I_t, $k-1$ coincides with the cardinal of set $I_{0:k}$, and n_{w_r} is the total number of images in $I_{0:k}$ containing the word w_r. This value is accumulated to the corresponding score according to:

$$\wp_j = \wp_j + \varrho_{w_r}^t\,, \tag{17}$$

being j the index of the image extracted from the inverted index. The computation of the scores finishes when all descriptors in F_t have been processed. Next, a combined similarity score s_j^i for image j is calculated and stored at location i:

$$s_j^i = g_i \cdot \wp_j\,. \tag{18}$$

A likelihood is then calculated according to the following rule (similarly to [7]):

$$\mathcal{L}\left(L_i^t | O_t\right) = \begin{cases} \dfrac{s_j^i - s_\sigma}{s_\mu} & \text{if } s_j^i \geq s_\mu + s_\sigma \\ 1 & \text{otherwise} \end{cases}\,, \tag{19}$$

being, respectively, s_μ and s_σ the mean and the standard deviation of the set of scores. Notice that, by means of (19), given the current observation O_t, only the most likely images update their posterior. The likelihood computation is formally stated in Algorithm 3. To finish, after incorporating the observation into our filter, the full posterior is normalized in order to obtain the probability $P\left(O_t | L_i^t\right)$.

VI. EXPERIMENTAL RESULTS

In this section, we evaluate HTMap from different points of view. An Intel Core i3 (2.27Ghz) / 8 GB RAM computer was used in all experiments. HTMap was configured by default using the parameters indicated in Table I.

Table II
DATASETS USED TO VALIDATE HTMAP

Name	# Imgs	Size (px)	Rate (hz)	Dist (km)
City Center [4]	1237	1280×480	0.5	2.0
New College [4]	1073	1280×480	0.5	1.9
KITTI 00 [54]	4541	1241×376	10	3.7
KITTI 05 [54]	2761	1226×370	10	2.2
KITTI 06 [54]	1101	1226×370	10	1.2
St. Lucia [55]	21815	640×480	15	17.6

A. Datasets

The evaluation has been performed using several community datasets, which correspond to different operating conditions. The City Center and the New College datasets, originally obtained for the evaluation of FAB-MAP [4], consist of, respectively, 1237 and 1073 pairs of images of size 640×480 taken by two cameras (one pointing left and one pointing right), mounted on a robot while it travels through the environment. Since our approach has been developed to be used with monocular cameras, we merged left and right frames resulting into images of size 1280×480. The City Center dataset was recorded to validate the ability of a system for matching images in the presence of scene changes, while the interest of the New College dataset is its high perceptual aliasing conditions. We also use several sequences from the KITTI suite [54]. The KITTI odometry benchmark consist of 22 outdoor sequences, with more than 40k images covering 39.2 km. Among these 22 sequences, there are 12 that contain loop closures. We employ sequences 00, 05 and 06 as a representative set of the benchmark. Finally, we validate HTMap against the St. Lucia 19-08-09 08:45 sequence [55], which comprises 17.6 km and 21815 images in a highly dynamic environment. The details of each dataset are summarized in Table II.

B. Loop Closure Detection

We first evaluate the performance of our hierarchical loop closure approach for recognizing previously seen places. In order to obtain performance measures, each dataset is provided with a ground truth, which indicates, for each image in the sequence, which other images can be considered as a loop. Since originally the KITTI sequences did not include a specific ground truth for loop closure detection, we use the ones provided by Arroyo et al. [28]. Each dataset is also provided with either a pose file or a file with GPS measurements, which are used to plot the different paths and hence visualize navigation and mapping results. For the St. Lucia dataset, this file was also used to generate the ground truth, using 10 meters as the threshold for placing two images in the same location.

The assessment against the ground truths is performed counting, for each sequence, the number of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN), where positive is meant for detection of loop closure. Then, precision-recall metrics are calculated:

- *Precision* is the ratio between real loop closures and the total amount of loop closures detected $(TP/(TP+FP))$.
- *Recall* is the ratio between real loop closures and the total amount of loop closures existing in the sequence $(TP/(TP+FN))$.

In our validation tests, we are particularly interested in the maximum recall that can be achieved for a precision of 100%, which implies no false positive detections in any case. The reason is that, in mapping tasks, a false positive can induce the algorithm to produce inconsistent maps and, therefore, avoiding these false positives becomes essential.

Since HTMap is based on a hierarchical combination of both BoW and global schemes, we want to verify its performance in comparison with two solutions of each of these paradigms executed alone. Therefore, we compare with FAB-MAP 2.0 [6] and SeqSLAM [29], which can be considered as state-of-the-art approaches of, respectively, each paradigm. The former has been evaluated using the binaries provided by the authors, while for the latter we have used OpenSeqSLAM [56].

Regarding FAB-MAP, it was configured with the default parameters and we ran it against all datasets using the outdoor vocabulary provided by the authors. Its output is a matrix, the n-th row of which corresponds to the probability distribution over previously seen places due to the n-th image. In this matrix, the main diagonal corresponds to the probability that the image comes from a new place. Since we do not take into account this case and we want to avoid the false detection of loop closures with recent frames, we rectify this matrix by removing the most recent probabilities for each row and normalizing the resulting distribution. A loop closure is detected if the probability is above a threshold τ_{FM}.

OpenSeqSLAM has been also configured with the default parameters, except the temporal length of the image sequences (d_s) which is, according to the authors, the most influential parameter of the algorithm. Longer sequence lengths usually perform better in terms of precision-recall, but, in some datasets, they can result into the opposite behaviour because of the absence of sequences of that length, specially in environments with frequent turns. Since we want to increase the performance of each approach, this parameter was experimentally set to 30, what maximized the recall in all datasets.

The precision–recall curves for each dataset are shown in Fig 6. In all cases, the curves result from modifying τ_{ep} in HTMap, τ_{FM} in FAB-MAP 2.0 and the loop closure acceptance threshold in SeqSLAM. For an easier understanding of the curves, best results at 100% of precision are also shown in Table III. As can be observed, the area under the curve (AUC) of HTMap is higher than the corresponding curves for the other solutions, outperforming them in all datasets. According to our experiments, SeqSLAM is usually able to obtain higher recall at 100% of precision than FAB-MAP 2.0, except for New College, where results are very similar. The performance of our approach is specially high for KITTI 00 and KITTI 06, where a recall above 90% results for a 100% of precision. The maximum recalls for the other datasets are 79.68% (City Center), 73.60% (New College), 75.88% (KITTI 05) and 70.11% (St. Lucia), which are very high in comparison with the other solutions. We believe that this increase in performance is due to our divide-and-conquer approach, which decreases the dispersion of loop closures between candidates

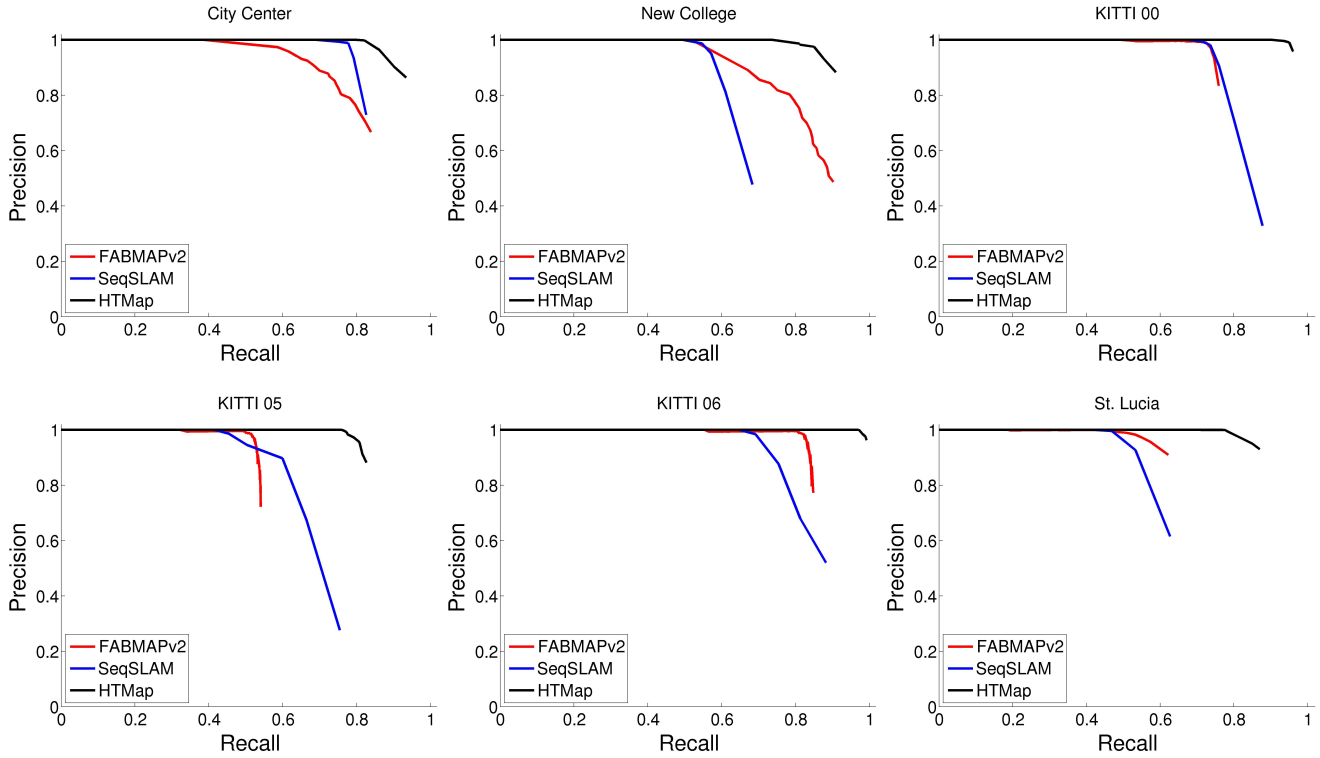
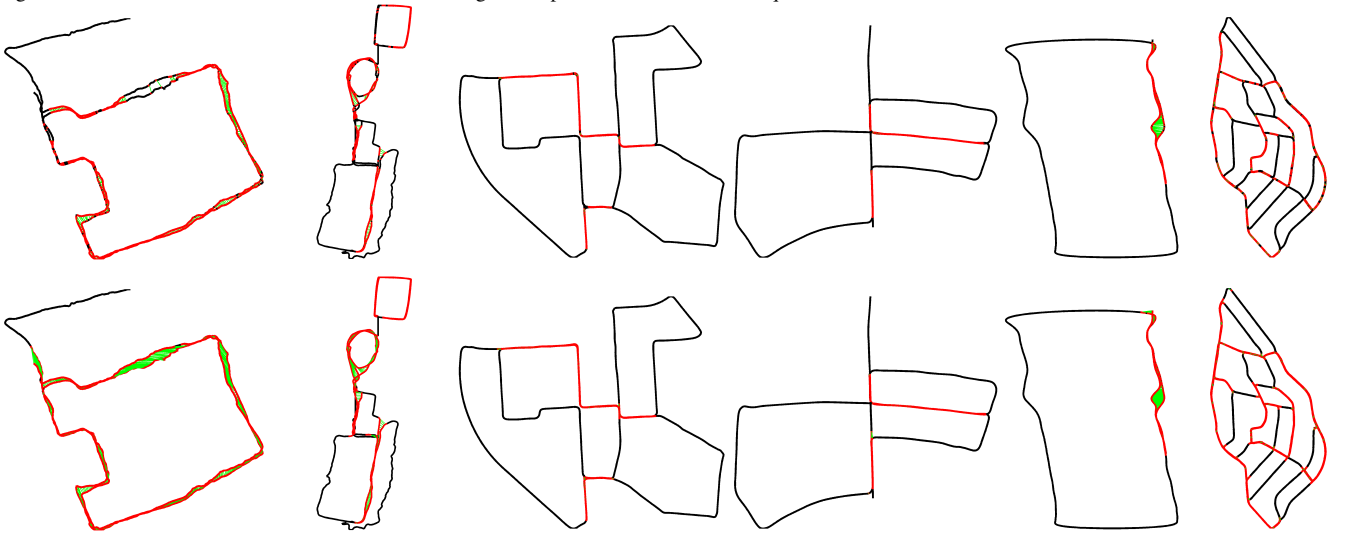


Figure 6. Precision-recall curves for each dataset using HTMap, FAB-MAP 2.0 and SeqSLAM.



(a) City Center (b) New College (c) KITTI 00 (d) KITTI 05 (e) KITTI 06 (f) St. Lucia

Figure 7. Appearance-based loop closure results for each dataset. Image positions are plotted as black dots. Wherever an image closes a loop with another image, both are labelled with a red dot and linked with a green line. These green lines highlight the presence of false positives, if any. Top row shows the results of HTMap, while the bottom row shows the ideal path that should be obtained if all the loops present in the dataset were correctly detected.

by means of a previous step for selecting similar areas of the environment. Notice that these results also show the usefulness of PHOG for this kind of tasks.

Table III also shows the value of τ_{ep} required to achieve the respective maximum precision-recall. Given these values, we want to analyse the effect of setting this parameter to a fixed value in terms of recall. To do that, we process the datasets for $\tau_{ep} = 140$, which ensures 100% of precision in all datasets, and then we evaluate the resulting recall. As can be observed, the recall decreases a bit in most datasets, since the number of positives resulting from the algorithm also decreases. Despite this, HTMap is able to achieve, on average, 77.80% of recall.

After the results presented in Table III, the top row of Fig. 7 shows the loops detected by HTMap at 100% of precision for each dataset. No false positives were detected in any case. In the figure, the positions available for each image of the dataset are used to spatially plot the images using black dots. When a loop closure is detected, images representing this loop are labelled in red and are linked with a green line. The bottom row of the figure shows the corresponding ground truths, i.e.

Table III
RESULTS AT 100% OF PRECISION FOR FAB-MAP 2.0, SEQSLAM AND HTMAP. RECALLS FOR $\tau_{ep} = 140$ ARE ALSO SHOWN.

	FABMAPv2		SeqSLAM		HTMap			
Dataset	Pr	Re	Pr	Re	Pr	Re	τ_{ep}	Re(140)
City Center	100	38.50	100	68.98	**100**	**79.68**	75	74.25
New College	100	51.91	100	49.39	**100**	**73.60**	125	67.80
KITTI 00	100	49.21	100	67.04	**100**	**90.24**	80	88.85
KITTI 05	100	32.15	100	41.37	**100**	**75.88**	**140**	75.88
KITTI 06	100	55.34	100	64.68	**100**	**97.03**	125	94.76
St. Lucia	100	18.54	100	41.56	**100**	**70.11**	75	65.25
						Average Recall ($\tau_{ep} = 140$)		**77.80**

the ideal path that should be obtained in case the approach detects all the loops of the dataset. Note that most part of the existing loops were detected by HTMap, specially in the KITTI and St. Lucia datasets.

C. Topological Mapping

The same sequences used in the previous experiments were employed to validate HTMap as for mapping. To this end, the topological maps obtained at 100% of precision are shown in Fig 8. For each dataset, a random color has been assigned to each map location and thus all images associated to the same node are labelled using the same color. We want to check whether images are tagged with the same color when revisiting a place. As can be seen, this is true in most cases, and it is specially evident in the City Center dataset, where a large loop is found. In this case, despite the map contains few locations, HTMap does not get confused and images corresponding to already visited places are assigned to the correct node. The total number of locations for each dataset resulted to be 18 (City Center), 86 (New College), 32 (KITTI 00), 19 (KITTI 05), 20 (KITTI 06) and 258 (St. Lucia).

D. Sparsity

The number of locations of a map with regard to the number of images is defined as its sparsity [11]. The lower the number of locations, the higher the sparsity, since more images are assigned to the same place. It is obvious that accurate loop closure detection relies on an adequate sparsity level.

In the first level of our loop closure approach, more location candidates are taken into account, what can induce false loop detections. Moreover, a larger sparsity makes the binary index of each location contain more visual words, sharing the scores among more loop closure candidate images. A good mapping technique must balance the number of locations and the ability of detecting previously seen places. In this section, we want to evaluate the effect of sparsity in HTMap performance.

The parameter with a major influence on the sparsity of the map, in HTMap, is τ_{nn}: the higher the value of τ_{nn}, the lower the number of locations (see section V-B). Since we are interested in avoiding false positives, we fix the parameter τ_{ep} to 150, which, as shown in Table III, is enough to ensure 100% of precision in all datasets. Then, we execute our approach varying τ_{nn} and the number of locations and recalls are observed. This experiment has been performed over the City Center and the New College datasets, since they have approximately the same length and have been taken at the same frame rate. This is to ensure that the number of generated locations is independent of the length of the sequence. Note that using a low value of τ_{ep} can lead to a higher recall, but in this case we are only interested in the relation existing between the number of locations and the recall produced by HTMap under these conditions. The results are shown in Fig. 9.

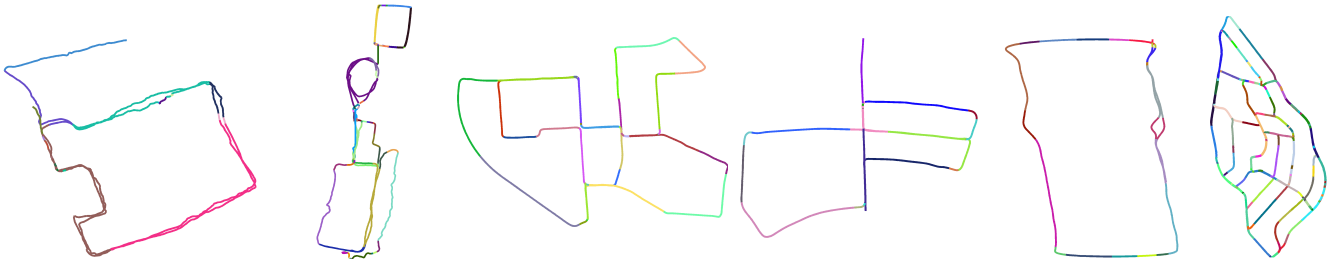
As can be seen, a high number of locations in HTMap does not imply better performance: from approximately 100 locations, the recall starts to decrease for both datasets. A number of locations between 10 and 80 are enough to guarantee the best recall values, being the recall more or less stable in this interval (≈ 0.70 for City Center and ≈ 0.65 for New College). This is an interesting point, taking into account that having more locations in the map could imply higher computational load during the first step of our loop closure approach, where the most likely locations are retrieved. Note that minimum sparsity is equivalent to perform loop closure detection at exclusively the image level. This proves that our grouping approach helps during the detection of previously seen places and validates the ability of the PHOG global descriptor to summarize the visual information that characterizes a place.

Table IV
AVERAGE COMPUTATIONAL TIMES (MS) OF HTMAP IN ALL DATASETS.

		CC	NC	K00	K05	K06	StL
DESC	PHOG	12.6	11.9	10.5	10.4	9.5	7.6
	FAST	39.2	38.4	30.5	30.4	28.5	19.4
	LDB	2.3	2.8	1.9	1.8	1.5	1.4
LC	LLC	2.2	2.4	2.5	2.8	2.9	1.6
	ILC	32.2	16.2	75.6	40.2	20.1	179.6
	PRED	0.01	0.01	0.02	0.02	0.01	0.1
	UPD	0.1	0.1	0.5	0.3	0.1	2.6
	EA	3.6	5.1	3.6	3.6	3.8	3.5
	Total	**92.2**	**76.9**	**125.1**	**89.5**	**66.4**	**215.8**
FABMAPv2		293.0	244.2	237.4	233.8	188.2	264.4
SeqSLAM		128.5	123.1	180.7	131.5	74.7	291.1

E. Computational Times

In this section, we evaluate the performance of HTMap in terms of computational time. To this end, we executed our approach over the datasets using the parameters that gave us the maximum recall in the previous experiments and, then, the average execution time of the different parts of the algorithm were measured. The results are shown in Fig. 10, where the average time per image is shown in groups of 100 images for enhancing the visualization. The times are also summarized in Table IV, where *PHOG* is the time needed to perform the global description of the image, *FAST* includes the keypoint detection and the selection of the best 1000 corners, and *LDB* is the time needed to compute the binary description of these corners. Regarding the loop closure detection times, *LLC* refers to the time required to perform the first step of our hierarchical loop closure algorithm, *ILC* involves the computation of the second step of HTMap, *PRED* is the time needed to make a prediction and *UPD* is the time needed to update the filter using the computed likelihood. Finally, *EA* includes matching features and the computation of the fundamental



(a) City Center (b) New College (c) KITTI 00 (d) KITTI 05 (e) KITTI 06 (f) St. Lucia

Figure 8. Topological maps generated at 100% of precision for each dataset. Images belonging to the same location are labelled with the same color. The total number of generated locations are, respectively: 18, 86, 32, 19, 20 and 258.

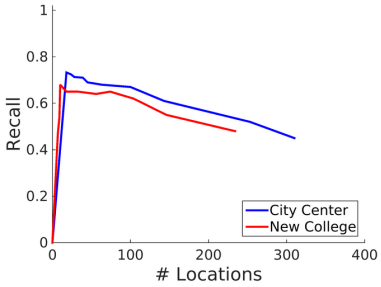


Figure 9. Sparsity analysis. Recall of HTMap according to the number of locations at 100% of precision and τ_{ep} set to 150 for City Center and New College datasets.

matrix. As can be observed, times are substantially short. Regarding image description, the computation of PHOG is even faster than detecting features using FAST and describing them using LDB. The total average time needed for describing an image is much faster than using SIFT or SURF [6]. The highest execution time corresponds to the image likelihood computation. This time increases with the number of nodes, since more locations must be searched. This effect could be reduced increasing τ_{llc} for selecting less location candidates or even limiting the number of locations to a fixed number, which would reduce the recall, but could be enough depending on the environment. In this case, we preferred to select all locations with a score higher than τ_{llc} to maximize the recall. Nonetheless, the maximum average time measured is 179.6 ms, which can be considered as a reasonable result given the number of images processed. The computation of the location likelihood is very fast despite the increment of the number of nodes in the map. Unlike FAB-MAP 2.0, where the likelihood computation time is roughly constant due to the use of a static visual dictionary, our approach makes use of an online visual dictionary, which can increase the likelihood computation time, but, as previously explained, provides other advantages such as the avoidance of the training stage and a vocabulary specifically adapted to the operating environment. According to our experiments, HTMap can process an image in 215.8 ms on average in the largest dataset.

In order to compare HTMap with FAB-MAP 2.0 and SeqSLAM in terms of computational times, we ran the binaries provided by the authors on the same machine as HTMap, adapting the code to measure the average processing time per image. From the shown in Table IV, we can observe that HTMap outperforms the other solutions also in terms of average computational time. Note that, while HTMap and

Table V
COMPARISON BETWEEN SINGLE AND HIERARCHICAL APPROACHES

	Single			Hierarchical		
Dataset	Pr	Re	Time (ms)	Pr	Re	Time (ms)
City Center	100	88.24	94.1	100	79.68	92.2
New College	100	53.15	83.4	100	73.60	76.9
KITTI 00	100	78.73	242.3	100	90.24	125.1
KITTI 05	100	62.58	145.14	100	75.88	89.5
KITTI 06	100	84.76	78.15	100	97.03	66.4
St. Lucia	100	64.32	465.1	100	70.11	215.8

SeqSLAM times increase with the number of images, FAB-MAP 2.0 presents a more stable behavior. The reason is that the number of visual words in the dictionary of FAB-MAP 2.0 is predefined, and so no insertion/removal overhead takes place during navigation. However, unlike HTMap, this implies a training phase. A previously-built visual dictionary can be employed to avoid the learning stage, but this usually results into a reduction of performance in terms of precision-recall. According to our results, SeqSLAM is faster than FAB-MAP 2.0 but not than HTMap.

To further validate the potential savings in computational time that our PHOG-based hierarchical approach offers, we have performed a last experiment comparing HTMap with a solution that indexes all the images using a single online BoW scheme [19], i.e. without using the PHOG global descriptor. The results are shown in Table V, where the maximum precision and recall values for each approach and its corresponding average time per image are indicated for all datasets. As can be seen, HTMap outperforms the single approach in most part of the datasets in terms of precision-recall. This is mainly imputable to the dispersion of the visual words that occurs in the index when a single approach is used. The most important observation has to do with the average time: the higher is the sequence, the higher is the improvement on the computational time performance. In effect, the average processing time in the City Center, New College and KITTI 06 datasets is similar in both approaches. However, as the number of processed images increases, the hierarchical approach is faster than the single one. This is specially evident in the KITTI 00 and the St. Lucia datasets. In the first case, HTMap needs 125.1 ms in front of 242.3 ms, which implies an increase in performance of 1.9x regarding the average execution time. In the second case, HTMap requires 215.8 ms per image while the single approach requires 465.1 ms per image, resulting in a 2.2x performance improvement.

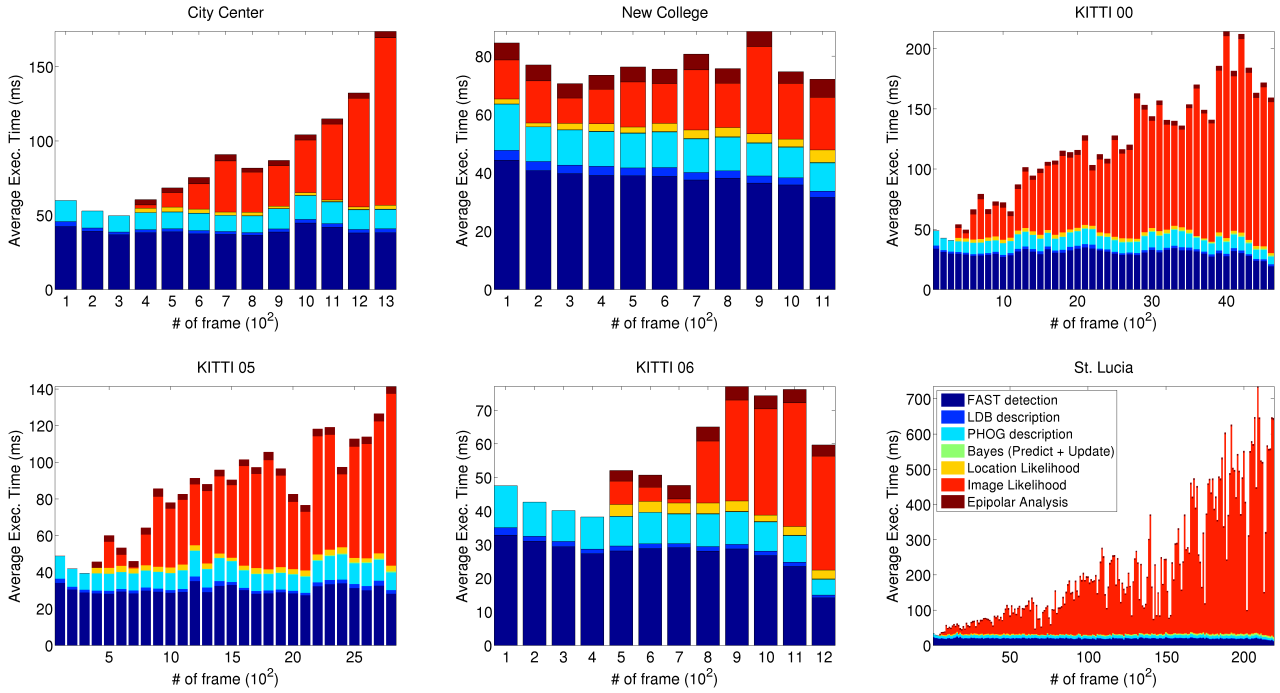


Figure 10. Computational times of HTMap for all datasets. Bars represent average execution times over intervals of 100 consecutive images.

VII. Conclusions

In this paper we have introduced a new topological mapping approach, HTMap, based on a hierarchical place recognition technique. Instead of generating a dense map, where all the images correspond to a node in the final topology, our approach builds a hierarchical decomposition of the environment, where images with similar properties are grouped together to form locations. Each location is represented by means of an average PHOG global descriptor and an index of binary features, which is based on a BoW scheme that can be built online. This map representation has proved to be very useful for reducing the search space when searching for loop closures. Then, as a key component of our topological mapping technique, we have also presented a hierarchical loop closure detection method. First, given the current image, PHOG global descriptors are used to obtain the most likely places in the map. After that, local binary features are used to obtain the most likely images belonging to the previously retrieved places. These scores are combined as a likelihood in a discrete Bayes filter. We have verified the utility of the PHOG descriptor in place recognition tasks, showing that it can be very helpful for summarizing the visual information that a place presents. HTMap has been shown to compare favourably with SeqSLAM and FAB-MAP 2.0 under the conditions imposed by several well-known public, long-distance and dynamic datasets, also comprising viewpoint changes and perceptual aliasing.

Referring to future research, despite the number of locations is not a critical factor in HTMap, we will consider the inclusion of some indexing method, such as kd-trees, for improving even more the retrieval of the most likely locations at the first step of the loop closure algorithm. We will also consider the incorporation of other global descriptors, given the good results obtained with PHOG.

References

[1] E. Remolina and B. Kuipers, "Towards a General Theory of Topological Maps," *Artificial Intelligence*, vol. 152, no. 1, pp. 47–104, 2004.

[2] F. Fraundorfer, C. Engels, and D. Nister, "Topological Mapping, Localization and Navigation Using Image Collections," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2007, pp. 3872–3877.

[3] M. Cummins and P. Newman, "Probabilistic Appearance Based Navigation and Loop Closing," in *IEEE Int. Conf. Robot. Autom.*, 2007, pp. 2042–2048.

[4] ——, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *Int. J. Rob. Res.*, vol. 27, no. 6, pp. 647–665, 2008.

[5] ——, "Highly Scalable Appearance-Only SLAM - FAB-MAP 2.0." in *Robot.: Sci. Syst.*, vol. 5, 2009, p. 17.

[6] ——, "Appearance-Only SLAM at Large Scale with FAB-MAP 2.0," *Int. J. Rob. Res.*, vol. 30, no. 9, pp. 1100–1123, 2011.

[7] 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.

[8] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, "Incremental Vision-Based Topological SLAM," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2008, pp. 1031–1036.

[9] 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.

[10] 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.

[11] H. Korrapati and Y. Mezouar, "Vision-Based Sparse Topological Mapping," *Rob. Auton. Syst.*, vol. 62, no. 9, p. 1259–1270, 2014.

[12] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[13] 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.

[14] 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.

[15] S. Leutenegger, M. Chli, and R. Siegwart, "BRISK: Binary Robust Invariant Scalable Keypoints," in *IEEE Int. Conf. Comput. Vision*, 2011, pp. 2548–2555.

[16] 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.

[17] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK : Fast Retina Keypoint," in *IEEE Conf. Comput. Vision Pattern Recog.*, 2012, pp. 510–517.
[18] 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.
[19] 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.
[20] A. Oliva and A. Torralba, "Modeling the Shape of the Scene : A Holistic Representation of the Spatial Envelope," *Int. J. Comput. Vision*, vol. 42, no. 3, pp. 145–175, 2001.
[21] A. Bosch, A. Zisserman, and X. Munoz, "Representing Shape with a Spatial Pyramid Kernel," *Image Processing*, vol. 5, no. 2, pp. 401–408, 2007.
[22] E. Garcia-Fidalgo and A. Ortiz, "Vision-based Topological Mapping and Localization Methods: A Survey," *Rob. Auton. Syst.*, vol. 64, pp. 1–20, 2015.
[23] I. Ulrich and I. Nourbakhsh, "Appearance-Based Place Recognition for Topological Localization," in *IEEE Int. Conf. Robot. Autom.*, vol. 2, 2000, pp. 1023–1029.
[24] J. Kosecka, L. Zhou, P. Barber, and Z. Duric, "Qualitative Image Based Localization in Indoors Environments," in *IEEE Conf. Comput. Vision Pattern Recog.*, vol. 2, 2003, pp. II–3–II–8.
[25] G. Singh and J. Kosecka, "Visual Loop Closing using Gist Descriptors in Manhattan World," in *Workshop on Omnidirectional Robot Vision (ICRA)*, 2010, pp. 4042–4047.
[26] N. Sunderhauf and P. Protzel, "BRIEF-Gist - Closing the Loop by Simple Means," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2011, pp. 1234–1241.
[27] 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.
[28] 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.
[29] 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.
[30] J. Kosecka and X. Yang, "Location Recognition and Global Localization Based on Scale-Invariant Keypoints," in *Work. Stat. Learn. Comput. Vision*, 2004.
[31] J. Kosecka and F. Li, "Vision Based Topological Markov Localization," in *IEEE Int. Conf. Robot. Autom.*, vol. 2, 2004, pp. 1481–1486.
[32] C. Valgren, T. Duckett, and A. Lilienthal, "Incremental Spectral Clustering and Its Application to Topological Mapping," in *IEEE Int. Conf. Robot. Autom.*, 2007, pp. 10–14.
[33] A. Kawewong, S. Tangruamsub, and O. Hasegawa, "Position-Invariant Robust Features for Long-Term Recognition of Dynamic Outdoor Scenes," *IEICE T. Inf. Syst.*, vol. E93-D, no. 9, pp. 2587–2601, 2010.
[34] A. Kawewong, N. Tongprasit, S. Tangruamsub, and O. Hasegawa, "Online and Incremental Appearance-based SLAM in Highly Dynamic Environments," *Int. J. Rob. Res.*, vol. 30, no. 1, pp. 33–55, 2011.
[35] Z. Zivkovic, B. Bakker, and B. Krose, "Hierarchical Map Building Using Visual Landmarks and Geometric Constraints," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2005, pp. 2480–2485.
[36] O. Booij, B. Terwijn, Z. Zivkovic, and B. Krose, "Navigation Using an Appearance Based Topological Map," in *IEEE Int. Conf. Robot. Autom.*, 2007, pp. 3927–3932.
[37] O. Booij, Z. Zivkovic, and B. Krose, "Efficient Data Association for View Based SLAM using Connected Dominating Sets," *Rob. Auton. Syst.*, vol. 57, no. 12, pp. 1225–1234, 2009.
[38] E. Garcia-Fidalgo and A. Ortiz, "Vision-Based Topological Mapping and Localization by means of Local Invariant Features and Map Refinement," *Robotica*, vol. 33, no. 7, pp. 1446–1470, 2015.
[39] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," in *IEEE Conf. Comput. Vision Pattern Recog.*, vol. 2, 2006, pp. 2161–2168.
[40] 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.
[41] D. Filliat, "A Visual Bag of Words Method for Interactive Qualitative Localization and Mapping," in *IEEE Int. Conf. Robot. Autom.*, 2007, pp. 3921–3926.
[42] 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.
[43] 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.
[44] L. Maohai, S. Lining, H. Qingcheng, C. Zesu, and P. Songhao, "Robust Omnidirectional Vision based Mobile Robot Hierarchical Localization and Autonomous Navigation," *Inf. Tech. J.*, vol. 10, no. 1, pp. 29–39, 2011.
[45] K. MacTavish and T. D. Barfoot, "Towards Hierarchical Place Recognition for Long-Term Autonomy," in *Work. Vis. Place Recogn. in Changing Envir.*, 2014, pp. 1–6.
[46] M. Mohan, D. Gálvez-López, C. Monteleoni, and G. Sibley, "Environment Selection And Hierarchical Place Recognition," in *IEEE Int. Conf. Robot. Autom.*, 2015, pp. 5487–5494.
[47] H. Korrapati, J. Courbon, Y. Mezouar, and P. Martinet, "Image Sequence Partitioning for Outdoor Mapping," in *IEEE Int. Conf. Robot. Autom.*, 2012, pp. 1650–1655.
[48] Z. Chen, A. Jacobson, U. M. Erdem, M. E. Hasselmo, and M. Milford, "Multi-Scale Bio-Inspired Place Recognition," in *IEEE Int. Conf. Robot. Autom.*, 2014, pp. 1895–1901.
[49] Z. Chen, S. Lowry, A. Jacobson, M. E. Hasselmo, and M. Milford, "Bio-Inspired Homogeneous Multi-Scale Place Recognition," *Neural Networks*, vol. 72, pp. 48–61, 2015.
[50] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *IEEE Conf. Comput. Vision Pattern Recog.*, 2005, pp. 886–893.
[51] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," in *Eur. Conf. Comput. Vision*, 2006, pp. 430–443.
[52] M. Muja and D. G. Lowe, "Fast Matching of Binary Features," in *Conf. Comput. and Rob. Vision*, 2012, pp. 404–410.
[53] K. Sparck Jones, "A Statistical Interpretation of Term Specificity and its Application in Retrieval," *J. Doc.*, vol. 28, pp. 11–21, 1972.
[54] 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.
[55] A. Glover, W. Maddern, M. Milford, and G. Wyeth, "FAB-MAP + RatSLAM: Appearance-based SLAM for Multiple Times of Day," in *IEEE Int. Conf. Robot. Autom.*, 2010, pp. 3507–3512.
[56] N. Sünderhauf, P. Neubert, and P. Protzel, "Are We There Yet? Challenging SeqSLAM on a 3000 km Journey Across All Four Seasons," *Work. Long-Term Auton.*, 2013.

Emilio Garcia-Fidalgo is a postdoctoral researcher with the Systems, Robotics and Vision (SRV) group, University of the Balearic Islands (Spain). He received the B.Sc. (2007), M.Sc. (2011) and Ph.D. (2016) degrees in Computer Science from this university. His research activity is mainly focused on topological mapping, appearance-based SLAM, visual loop closure detection and UAVs.

Alberto Ortiz is Associate Professor at the Department of Mathematics and Computer Science of the University of the Balearic Islands (UIB). He holds B.Sc. and Ph.D. degrees in Computer Science. He is author and co-author of more than 140 publications related with computer vision and mobile robotics. His current research interests are localization and mapping, visual guidance of mobile robots including obstacle detection and avoidance, control architectures for mobile robots and machine learning.