Vision-Based Topological Mapping and Localization by means of Local Invariant Features and Map Refinement[*]

Emilio Garcia-Fidalgo and Alberto Ortiz
Department of Mathematics and Computer Science,
University of the Balearic Islands, Spain
{emilio.garcia, alberto.ortiz}@uib.es

February 14, 2014

Abstract

We propose an appearance-based approach for topological visual mapping and localization using local invariant features. To optimize running times, matchings between the current image and previously visited places are determined using an index based on a set of randomized kd-trees. We use a discrete Bayes filter for predicting loop candidates, whose observation model is a novel approach based on an efficient matching scheme between features. In order to avoid redundant information in the resulting maps, we also present a map refinement framework, which takes into account the visual information stored in the map for refining the final topology of the environment. These refined maps save storage space and improve the execution times of localizations tasks. The approach is validated using image sequences from several environments and compared with the state-of-the-art FAB-MAP 2.0 algorithm.

[*]This work is supported by the European Social Fund through the grant FPI11-43123621R (Conselleria d'Educacio, Cultura i Universitats, Govern de les Illes Balears) and by FP7 project INCASS (GA 605200).

1 Introduction

Mapping and localization are essential problems in mobile robotics. In order to solve them, several approaches propose to perform both tasks at the same time, creating an incremental map of an unknown environment while localizing the robot within this map. These techniques are known as Simultaneous Localization and Mapping (SLAM) [1]. In SLAM, loop closure detection is a key challenge to overcome. It entails the correct detection of previously seen places from sensor data. This allows generating consistent maps and reducing their uncertainty.

Ultrasounds and laser sensors have been used for years for SLAM and loop closure detection. Nevertheless, in the last decades there has been a significant increase in the number of visual solutions because of the low cost of cameras, the richness of the sensor data provided and the availability of cheap powerful computers. This naturally guides us to an appearance-based SLAM, where the environment is represented in a topological way by means of a graph. Each node of this graph represents a distinctive visual location visited by the robot, while the edges indicate connectivities between locations. Using this representation, the loop closure problem can be solved by direct image comparison, what avoids the need of maintainance and estimation of the position of the feature landmarks found within the environment.

In the Bag-of-Words (BoW) [2] approach, local invariant features obtained from an image are quantized into a vector according to a visual vocabulary, typically built during a training phase through a clustering algorithm, e.g. k-means. This representation is one of the most used techniques in appearance-based SLAM. However, this method presents some drawbacks. On the one hand, it is affected by the perceptual aliasing effect [3], i.e. two different places are perceived as the same because of the similarity between their representations. The quantization process performed during the clustering step can contribute to emphasize this effect even more. On the other hand, the training phase is typically performed off-line and can take a long time.

Topological maps obtained from visual information tend to contain spurious paths and nodes [4, 5]. This is because of image noise, partial invariance of image descriptors to viewpoint, scale and/or illumination changes, or due to the mapping algorithm itself. The final map obtained can be very large and can contain more nodes than are actually required to represent the environment, resulting in an increment of the storage needs and the computational requirements. We propose to refine the map as it is being built, instead of investing efforts in improving the mapping algorithm.

To cope with the aforementioned issues, this paper presents a complete visual mapping and localization framework based on raw local invariant features and a map refinement strategy. Our framework was assessed using multiple indoor and outdoor datasets captured under different weather conditions and illumination changes. As main contributions, we present a Bayesian framework for visual loop closure detection which uses constellations of local invariant features as image descriptors. It comprises a novel observation model which allows us to succeed in challenging loop closure situations such as camera rotations, occlusions and changes in illumination. Using this algorithm as a key component, we also propose a topological mapping and localization framework, which uses a map refinement strategy to remove the redundant paths that may appear during the graph building process. The strategy is presented in this work, by means of a novel map refinement theory based on the visual information gathered from the environment. This refinement is done online each time a loop closure is detected.

This work extends our previous solution [6] with the contributions enumerated above and as well as with a more extensive evaluation of the approach. A comparison with the well-known FAB-MAP 2.0 algorithm is also included.

The rest of the paper is organized as follows: Section 2 enumerates fundamental works related to loop closure detection and visual localization and mapping, Section 3 explains the basics of our algorithm, including how images are described and matched, our Bayesian loop closure algorithm using visual features and our novel map refinement framework, Section 4 reports experimental results obtained from different datasets, and Section 5 concludes the paper.

2 Related Work

A high number of appearance-based localization and mapping solutions were proposed during the last decade. Although many approaches assume the availability of omnidirectional images [7, 8, 9, 10, 11, 12, 13], many others make use of monocular configurations [14, 15, 16, 17, 18, 19]. Our approach belongs to this latter class.

Referring to the description of the image, the BoW approach, initially used by Sivic and Zisserman [2] for searching images in video sequences, has become quite popular. Descriptors extracted from a set of training images are clustered using the k-means algorithm, generating a visual vocabulary. When a query input frame is received, its descriptors are quantized using these visual words. As a result, the image is described by a list of integers specifying the number of occurrences of each visual word in the image. Using inverted files, a scoring process based on Term Frequency Inverse Document Frequency (TF-IDF) weighting is performed in order to select similar previous images. The main drawback of this approach was the high computational cost, since a linear search was performed in order to find the nearest reference descriptor in the visual dictionary. Nister and Stewenius [20] improved this indexing scheme proposing a vocabulary tree based on a hierarchical k-means approach. This hierarchical quantization allows to use a larger vocabulary size leading to a better recognition performance. Using this approach, Fraundorfer et al. [18] presented a highly scalable vision-based localization and mapping method using image collections. They used local geometric information to navigate within the topological map. Since the BoW approach does not take into account the spatial arrangement of the detected features, recently Johns and Yang [19] showed that quantizing local features in both feature and image space enhances the recognition performance at different times of the day.

Probably the most popular solution based on the BoW approach is Fast Appearance-Based Mapping (FAB-MAP) [14], where a Chow-Liu tree approximates the co-occurrences between the visual words in the vocabulary. This approximation permits the authors to compute efficiently an observation likelihood which is used in a Bayes filter for predicting loop closure candidates. Initially, this likelihood was computed for each image candidate in the filter, resulting into computational problems. To speed up this process, their work was improved in [21], introducing a probabilistic bail-out test based on the use of concentration inequalities for rapidly identifying promising loop closure hypotheses, and in [22], adapting the probabilistic model to be used with an inverted index architecture similar to image typical search engines. Recently, Glover et al. [23] employed FAB-MAP as a loop closure component to create a robust appearance-based SLAM system obtaining interesting results in outdoor environments.

Some authors proposed that a visual vocabulary can be built online while the robot is navigating, avoiding the offline training phase. Angeli et al. [15, 16], using the online visual dictionary proposed by Filliat [24], extended the BoW paradigm to incremental conditions and relied on Bayesian filtering to estimate the probability of loop closure. Their work was expanded constructing a complete topological SLAM system [4] and including robot odometry information [25]. Nicosevici and Garcia [26] presented an online visual vocabulary building method based on agglomerative clustering. They used this algorithm for mapping underwater environments. Since these methods avoid the execution of the offline training phase, they are limited to smaller vocabulary sizes.

Despite its well-known general performance, the BoW paradigm is more affected by perceptual aliasing [3]. For this reason, our loop closure detection algorithm follows an approach similar to [15, 16], but uses local invariant features for image description and matching.

Other approaches make use of global descriptors, such as Gist [27]. Singh and Kosecka [12] computed Gist descriptors from omnidirectional images of urban environments for detecting loop closures. They presented a novel image matching strategy for panoramas, but Bayes filtering is not considered in this work. Liu and Zhang [28] applied Principal Component Analysis (PCA) to Gist descriptors in order to compute the likelihood in a particle filter which is used for detecting loop closures. Siagian and Itti [29] presented a biologically-inspired system to scene classification using Gist as image representation. Recently, other global descriptors were used for mapping and localization, e.g. BRIEF-Gist [30], Weighted Gradient Orientation Histogram (WGOH) [31], Weighted Grid Integral Invariant (WGII) [32], Orientation Adjacency Coherence Histogram (OACH) [33] or Whole Image SURF (WI-SURF) [34]. The main drawback of these descriptors is that they are not descriptive enough, and thus they are more sensitive to noise, which leads to a larger number of incorrect detections. For this reason, they are usually employed for place categorization, where the goal is to know the type of place where it is.

Rather than BoW or global descriptors, some authors used local invariant features for visual localization and mapping as well as for loop closure detection. Kawewong et al. presented Position-Invariant Robust Features (PIRFs) [11], a method for generating averaged features from Scale-Invariant Feature Transform (SIFT) keypoints [35] that can be matched along several consecutive frames. These features are then used as input in an incremental appearance-based SLAM algorithm based on a majority voting scheme. Valgren et al. [36] used Incremental Spectral Clustering (ISC) to cluster images and create a topological map of the environment. Bacca et al. [37] proposed an innovative feature management approach for topological map-building and localization, which is based on a human memory model and implements concepts such as Short-Term Memory (STM) and Long-Term memory (LTM). Using Feature Stability Histograms (FSH), their method can deal with temporary occlusions and changes in illumination caused by dynamic environments. Booij et al. [5] showed a navigation system based on a topological map which used the epipolar geometry to obtain a robust heading estimation. Recently, Zhang [3] presented a method for selecting a subset of SIFT features extracted from an image. These features are used for matching consecutive images. A location is represented by a set of features that can be matched consecutively along several images. The problem of this approach is that the number of features to manage increases while new images are added, and a linear search for matching becomes intractable. In order to overcome this issue, in this work, we index features using a set of randomized kd-trees.

Even though most solutions are based on either topological or metric maps, some authors tried to make hybrid solutions. Zivkovic et al. [7] presented an algorithm for automatically generating hierarchical maps from images. A low-level map is built using SIFT keypoints. Then they cluster nodes to construct a high-level representation. In [13], a hierarchical localization method for omnidirectional images is proposed. Vertical lines are matched using pyramidal matching in order to obtain the most similar image to the current one in a predefined visual memory. The 1D radial trifocal tensor is employed to obtain a metric localization. Blanco et al. [38] presented a solution called Hybrid Metric-Topological SLAM (HTM-SLAM), where they propose probability distributions over both metric and topological maps. These approaches try to maximize the advantages and minimize the problems of each kind of map alone and combine them in a different mapping technique.

3 Algorithm Overview

The main goal of our approach is to construct a clean visual representation of the robot environment using a single monocular camera while localizing the robot within this map. Since in a real scenario storing all images taken by the camera is impossible, we need to reduce the number of images to manage without losing visually distinct locations found in the robot environment. The elements of this subset of images are called *keyframes* [39]. Our approach is also based on the keyframe concept, as it is outlined in Fig. 1 and Alg. 1. In our map, each node represents a keyframe image, and each keyframe is represented by its corresponding SIFT [35] features. In order to select these keyframes, we discard: (a) images similar to the current location of the robot (keyframe), since they do not provide distinct visual information about the environment and therefore are redundant; and (b) robot camera turns, because they are noisy and can introduce errors in the mapping and localization processes. For the first case, SIFT features of the current image are matched applying the ratio test [35] to the features of the current location keyframe. If the number of matched features is higher than a threshold, the image is considered similar to the current location. The same matching step is applied between the current image and the last received image in the sequence: if it is not possible to match a certain number of features, the image is classified as a turn. In these two cases, the image is discarded. Otherwise, it is considered useful and needs to be processed in order to determine whether it is a loop closure or a new keyframe to be added to the map. This keyframe selection policy is shown graphically in Fig. 2.

Our loop closure approach makes use of a discrete Bayes filter. This filter is updated with every image irrespective of whether the image has been discarded or not. If a loop closure is not found, the current image is considered as a new keyframe and is added to the map as a new node. Otherwise, we create a link between the current location of the robot and the loop closure candidate and, then, a map refinement process runs, in order to determine if redundant paths exist. As a consequence, a set of superfluous nodes may be detected. If this is the case, they are removed from the map, and the robot position within the map is updated accordingly.

In order to avoid false loop closure detections between the current image and its neighbours in the sequence, new keyframes are not inserted directly as loop closure hypotheses in the filter.

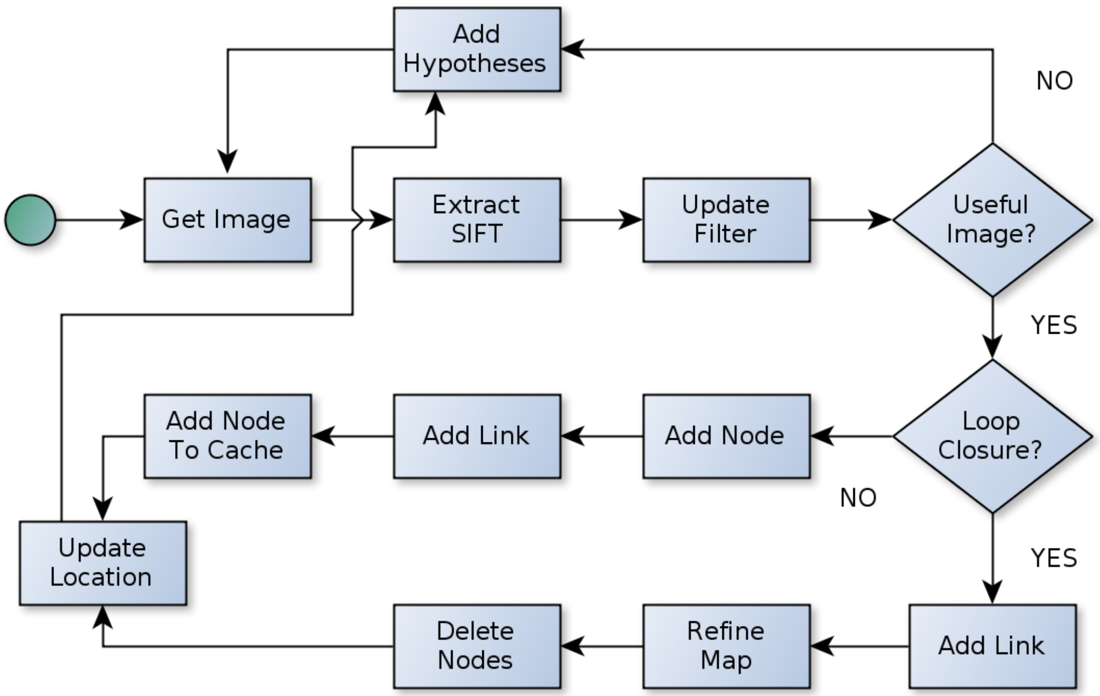


Figure 1: Overall algorithm diagram. See text for details.

They are instead stored in a temporarily cache list and pushed into the filter once a certain number of images have been considered.

The image description and matching process, the loop closure detection algorithm and the map refinement strategy are detailed in the following sections.

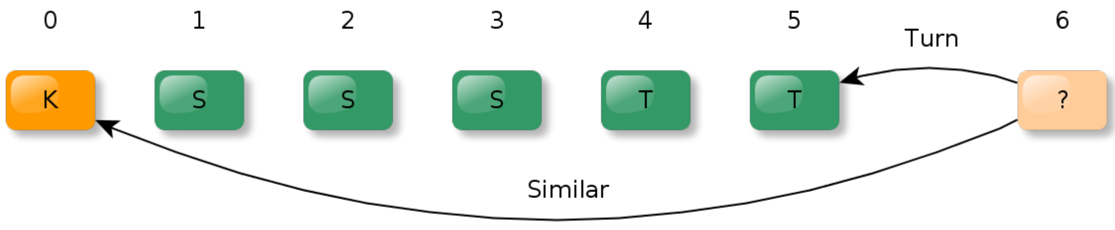


Figure 2: Image selection policy. The current image taken by the camera (6) is matched with the image that represents the current keyframe (0) and the last received image in the sequence (5) in order to determine if it is a useful frame. *K* represents the current location (keyframe), *S* and *T* represent images discarded because they are, respectively, similar enough to the current location or camera turns.

Algorithm 1 Appearance-Based Mapping and Localization

```
/* Variables */
$I = \{I_0, \ldots, I_{N-1}\}$: Sequence of N input images.
$G$: Graph representing the environment topology.
$k$: Current keyframe index.
$F_t$: Set of SIFT features obtained from image $I_t$.
$c$: Candidate keyframe index for closing a loop.
$M^i_j$: Matchings between images $I_i$ and $I_j$.
$C$: List of nodes not inserted in the filter as hypotheses yet.
$l$: Boolean variable indicating whether a loop was detected or not.

$k = 0$
$F_0$ = describe($I_0$)
addToCache($C, 0$)
for $t = 1$ to $N - 1$ do /* While there are images */
    $F_t$ = describe($I_t$)
    updateFilter($F_t$)
    $M^t_k$ = match($F_t, F_k$)
    $M^t_{t-1}$ = match($F_t, F_{t-1}$)
    if useful($M^t_k, M^t_{t-1}$) then /* Useful Image */
        $l, c$ = detectLoopClosure($G, F_t$)
        if $l$ then /* Loop Closure Detected */
            addLink($G, k, c$)
            refineMap($G$)
            $k = c$
        else /* New node */
            addNode($G, t$)
            addLink($G, k, t$)
            addToCache($C, t$)
            $k = t$
        end if
    end if
    releaseHypotheses($C$)
end for
```

3.1 Image Description and Matching

As commented above, in our approach, each image is described using the SIFT [35] algorithm. It has been empirically verified to be invariant to uniform scaling and orientation, and partially invariant to affine distortion and illumination changes. SIFT has been used for several applications, including object recognition, robot localization and mapping, image stitching, gesture recognition and 3D scene modeling. Another reason for using SIFT is that our algorithm uses a kd-tree-based algorithm for indexing features. These algorithms assume the features exist in a real vector space where each dimension of the features can be continuously averaged. For this reason, binary features like BRISK [40], ORB [41] or FREAK [42] are not suitable to our needs, despite their lower detection and description times.

SIFT defines interest points as maxima and minima of a difference of Gaussians function applied, in scale space, to a series of resampled images. Each feature is then described using a histogram of gradient orientations around the point at the selected scale, resulting in a 128-dimensional descriptor. In this work, we compare these descriptors using Euclidean distance.

The loop closure detection algorithm, as we will see shortly, needs to match efficiently the features of the current image with features of all previously considered keyframes, in order to determine whether it is a revisited place. Therefore, a method for an efficient nearest neighbour search is needed in order to match these high-dimensional descriptors. Tree structures have been widely used to this end, since they reduce the search complexity from linear to logarithmic. To the same purpose, we maintain a set of randomized kd-trees containing all the SIFT descriptors of previously detected keyframes. An inverted index structure, which maps each feature to the keyframe where it was found, is also created. Given a query descriptor, these structures allow us to obtain, traversing the tree just once, the top K nearest keypoints among all keyframes in an efficient way.

3.2 Probabilistic Loop Closure Detection

Given a new image, a discrete Bayes filter is used to detect loop closure candidates. This filter estimates the probability that the current image closes a loop with previously seen locations, allowing us to deal with noisy measurements and uncertainty in the robot location and helping us to discard false recognitions. The Bayesian framework is also used for ensuring temporal coherency between consecutive predictions, integrating past estimations over time. It can also be used for fusing sensory information from different sources, such as cameras, lasers or IMUs, providing an observation model for each one. In our case, we only use a camera as input.

Given the current image I_t at time t, we denote z_t as the set of SIFT descriptors extracted from this image. These are the observations in our filter. We also denote L_i^t as the event that image I_t closes a loop with image I_i, where $i < t$. Using these definitions, we want to detect the image of the map I_c whose index satisfies:

$$c = \underset{i=0,\dots,t-p}{\arg\max} \{ P \left(L_i^t | z_{0:t} \right) \} , \tag{1}$$

where $P \left(L_i^t | z_{0:t} \right)$ is the full posterior probability at time t given all previous observations up to time t. As in [16], the most recent p images are not included as hypotheses in the computation of the posterior since I_t is expected to be very similar to its neighbours and then false loop closure detections will be found. This parameter p delays the publication of hypotheses and needs to be set according to the frame rate or the velocity of the camera.

Separating the current observation from the previous ones, the posterior can be rewritten as:

$$P\left(L_i^t | z_{0:t}\right) = P\left(L_i^t | z_t, z_{0:t-1}\right) \ , \tag{2}$$

and then, using conditional probability properties[1], the next equality holds:

$$P\left(L_i^t | z_t, z_{0:t-1}\right) P\left(z_t | z_{0:t-1}\right) = P\left(z_t | L_i^t, z_{0:t-1}\right) P\left(L_i^t | z_{0:t-1}\right) \ , \tag{3}$$

from where we can isolate our final goal to obtain:

$$P\left(L_i^t | z_t, z_{0:t-1}\right) = \frac{P\left(z_t | L_i^t, z_{0:t-1}\right) P\left(L_i^t | z_{0:t-1}\right)}{P\left(z_t | z_{0:t-1}\right)} \ . \tag{4}$$

where $P\left(z_t | z_{0:t-1}\right)$ 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:

$$P\left(L_i^t | z_{0:t}\right) = \eta P\left(z_t | L_i^t\right) P\left(L_i^t | z_{0:t-1}\right) \ , \tag{5}$$

where η represents the normalizing factor, $P\left(z_t | L_i^t\right)$ is the observation likelihood and $P\left(L_i^t | z_{0:t-1}\right)$ is the probability distribution after a prediction step. Decomposing the right side of (5) using the Law of Total Probability, the full posterior can be written as:

$$P\left(L_i^t | z_{0:t}\right) = \eta P\left(z_t | L_i^t\right) \sum_{j=0}^{t-p} P\left(L_i^t | L_j^{t-1}\right) P\left(L_j^{t-1} | z_{0:t-1}\right) \ , \tag{6}$$

where $P\left(L_j^{t-1} | z_{0:t-1}\right)$ 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.

[1] $P(A \mid B, C)\, P(B \mid C) = P(B \mid A, C)\, P(A \mid C) \Rightarrow \frac{P(A \cap B \cap C)}{P(B \cap C)} \frac{P(B \cap C)}{P(C)} = \frac{P(A \cap B \cap C)}{P(A \cap C)} \frac{P(A \cap C)}{P(C)}$

Unlike Angeli [16] and Cummins [14], we do not model explicitly the probability of no loop closure in the posterior. If the loop closure probability of I_t with I_c ($P\left(L_c^t|z_{0:t}\right)$) is not high enough, we discard L_c^t as a possible loop candidate.

3.2.1 Transition Model

Before updating the filter using the current observation, the loop closure probability at time t is predicted from $P\left(L_j^{t-1}|z_{0:t-1}\right)$ 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 four of its neighbours ($j-2$, $j-1$, j, $j+1$, $j+2$) using coefficients (0.1, 0.2, 0.4, 0.2, 0.1), i.e. 0.9 $\times$ (0.1, 0.2, 0.4, 0.2, 0.1). The remaining 10% is shared uniformly across the rest of loop closure hypotheses according to $\frac{0.1}{\max\{0,t-p-5\}+1}$. 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.

Our model is similar to the one presented by Angeli [16] but using different coefficients in order to give more importance to the central image of the Gaussian. FAB-MAP employs a Gaussian-like function using only two neighbours, reducing the speed transition of the filter.

3.2.2 Observation Model

Once the prediction step is performed, the current observation needs to be included in the filter. We have to compute the most likely locations given the current image I_t and its keypoint descriptors z_t, but we want to avoid comparing I_t with each previous keyframe, since this is not tractable. To this end, we use the structures described in section 3.1. Note that if the robot revisits the same place several times and the current image I_t closes this loop again, each descriptor in z_t can be close to descriptors from different previous images in the Euclidean space. This fact is taken into account in the computation of our likelihood.

Since we do not use a BoW model, we can not rely on solutions created for these representations like the TF-IDF score [2] used by Angeli [16], or on an observation likelihood based on a precomputed Chow-Liu tree like Cummins [14]. Instead, our observation model provides an efficient way of obtaining loop closure candidates using local scale invariant features and an indexing structures such as kd-trees.

For each hypothesis i in the filter, a score $s\left(z_t, z_i\right)$ is computed. This score represents the likelihood that the current image I_t closes the loop with image I_i given their descriptors, z_t and z_i respectively. Initially, these scores are set to 0 for all frames from 0 to $t-p$. For each descriptor in z_t, the K closest descriptors among the previous keyframe images are retrieved without taking into account the p previous frames; next, each of them, denoted by n, adds a weight w_n to the score of the image where it appears. This value is normalized using the total distance of the K candidates retrieved:

$$w_n = 1 - \frac{d_n}{\sum\limits_{k \in K} d_k}\, , \forall n = 1, \ldots, K\, , \tag{7}$$

where d is the Euclidean distance between the considered query descriptor in z_t and the nearest neighbour descriptor found in the tree structure. This value is accumulated onto a score according to:

$$s\left(z_t, z_{j(n)}\right) = s\left(z_t, z_{j(n)}\right) + w_n\, , \forall n = 1, \ldots, K\, , \tag{8}$$

being $j(n)$ the index of the image where the candidate descriptor n was extracted. The computation of the scores is finished when all descriptors in z_t have been processed. Then, the likelihood function is calculated according to the following rule (similarly to [16]):

$$P\left(z_t | L_i^t\right) = \begin{cases} \frac{s(z_t, z_i) - s_\sigma}{s_\mu} & \text{if } s\left(z_t, z_i\right) \geq s_\mu + s_\sigma \\ 1 & \text{otherwise} \end{cases}\, , \tag{9}$$

being respectively s_μ and s_σ the mean and the standard deviation of the set of scores. Notice that by means of (9), given the current observation z_t, only the most likely locations update their posterior. After incorporating the observation to our filter, the full posterior is normalized in order to obtain a probability distribution.

Our observation model enables us to obtain similar past scenes in challenging situations such as illumination changes, appearance modifications, camera rotations and scene occlusions. This will be shown in the experimental results section, where the observation likelihood for these kinds

Algorithm 2 Visual Loop Closure Detection

```
/* Variables */
$B$: Discrete Bayes filter.
$F_t$: Set of SIFT features obtained from image $I_t$.
$c$: Candidate image index for closing a loop.
$P_c$: Probability of candidate image $c$.
$n_{hyp}$: Number of hypotheses in the Bayes filter.
$E_j^i$: Set of matchings surviving the epipolarity constraint-based filter.
$L$: Output boolean variable for indicating the existence of a loop.
$L_{im}$: Output integer with the index of the loop closure image.

/* Thresholds */
$T_{loop}$: Minimum probability to consider a loop candidate.
$T_{ep}$: Minimum number of surviving matchings after epipolar geometry validation.
$T_{hyp}$: Minimum number of hypotheses for considering loop candidates.

$c$, $P_c$ = getCandidate($B$) /* Getting the best loop candidate */
if $P_c > T_{loop}$ and $n_{hyp} > T_{hyp}$ then
    $E_c^t$ = epipolarGeometry($F_t$, $F_c$)
    if length($E_c^t$) $> T_{ep}$ then
        $L =$ True; $L_{im} = c$
    else
        $L =$ False; $L_{im} = -1$
    end if
else
    $L =$ False; $L_{im} = -1$
end if
```

of loop closure situations presents clear peaks despite their complexity.

3.2.3 Selection of a Loop Closure Candidate

In order to select a final candidate, we do not search for high peaks in the posterior distribution, because loop closure probabilities are usually diffused between neighbouring images. This is due to visual similarities between consecutive keyframes in the sequence. Instead, for each location in the filter, we sum the probabilities along a predefined neighbourhood. This neighbourhood is the same as defined in section 3.2.1, i.e. frames ($j-2$, $j-1$, j, $j+1$, $j+2$) for image j.

The image I_j with the highest sum of probabilities in its neighbourhood is selected as a loop closure candidate. If this probability is below a threshold T_{loop}, the loop closure hypothesis is not accepted. Otherwise, an epipolarity analysis between I_t and I_j 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 surviving matchings is above a threshold T_{ep}, the loop closure hypothesis is accepted; otherwise, it is definitely rejected.

Finally, we define another threshold T_{hyp} to ensure a minimum number of hypotheses in the filter, so that loop closure candidates are meaningful. This step counteracts the fact that first images inserted in the filter tend to attain a high probability of loop closure after the normalization step, what leads to incorrect detections. The full loop closure detection approach is outlined in Alg. 2.

3.3 Map Refinement

Visual topological maps tend to contain redundant nodes and paths due to several reasons. On the one hand, sometimes the current image acquired by the robot is blurred, what makes difficult to identify loop closures at the right time and therefore new nodes are added to the map. The loop closure is identified once the image stream becomes stable again. The net result is that a redundant path is generated because of the noisy images. On the other hand, the Bayes filter does not detect a revisited place instantaneously, but needs some frames to become aware of the loop closure: along these frames, the posterior moves from one keyframe (hypothesis) to another, while a new path containing the unmatched frames is created. In this work, we present a map refinement framework based on the visual information obtained from each node of the environment in order to correct these problems, which are common of many vision-based topological mapping solutions, and in order to maintain the map structure as simple as possible in storage and computational terms.

Our method is executed each time a loop closure is detected. The idea is to refine the local area of the map around the loop-closing node, since the redundant paths are generated within this zone. To this end, its k-neighbourhood is obtained. This is the set of nodes from which we can reach the loop-closing node in k steps or less, where k was set experimentally to 10. For each element in this set, all paths to the loop-closing node are obtained using an adjacency list. If there is only one path between the nodes, there are no redundant paths and this route is left unaltered. Otherwise, a further analysis of the different paths is performed. To this end, a path P between nodes i and j is defined as:

$$P_j^i = \{N_0, N_1, \ldots, N_n\}, 0 \leq n \leq k+1, \quad (10)$$

being N_0 the starting node of P_j^i and N_n the loop-closing node. We define the *erasability* of a

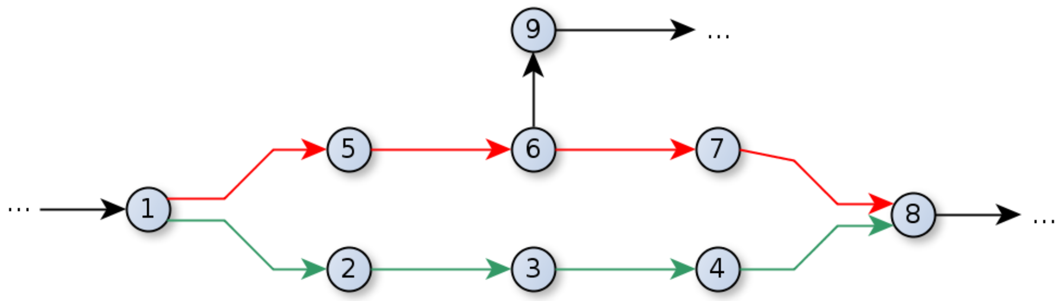


Figure 3: Example of erasability. Two paths exist between node 1 and node 8. The red path (up) is classified as *non-erasable*, since the inner node 6 does not hold condition (11). This path can not be removed without losing the path starting at node 9. The green path (bottom) is classified as *erasable* and is a candidate to be removed.

path as:

$$M(P_j^i) = (deg^-(N_i) = 1) \wedge (deg^+(N_i) = 1)\,, \forall i = 1, \dots, n-1\,, \tag{11}$$

where deg^- and deg^+ are, respectively, the input degree and the output degree of a vertex. Therefore, a path is classified as *erasable* if (11) holds for each inner node of the path. Otherwise, the path is classified as *non-erasable*. The meaning of an erasable path in our context is that the route can be deleted without breaking the topology of the environment. Examples of erasable and non-erasable paths are shown in Fig. 3.

Once all paths have been classified according to their erasability, a decision about which ones can be deleted is made, taking into account that real alternative paths need to be preserved, since they correspond to parts of the environment. To this end, we propose to generate a model path taking into account the visual features of all paths and comparing each erasable path against this model to verify if this is a redundant or a real path. In order to create the model path, a k-means clustering process with 100 centroids is performed using the SIFT features of the keyframes that are included in all paths between the corresponding nodes. This gives us a set of reference virtual descriptors representing all the paths as a whole. Then, k-means is also used for quantizing the descriptors of the nodes of each erasable path, obtaining representatives for each route. A set of randomized kd-trees is created using the reference virtual descriptors. The virtual descriptors of each path are matched against the model using these trees in order to obtain a distance between each erasable path and the model path. This distance is computed as the average distance of the matched centroids.

Algorithm 3 Map Refinement

```
/* Variables */
$n_l$: Input loop closing node.
$N$: K-neighbourhood of a node.
$P$: The set of paths between two nodes.
$M$: Array to store the erasability of each path in a set.
$D$: Array to store the distances between each path in a set and a reference model.
$R_m$: Reference model path. It is computed using k-means clustering.
$P_{max}$: Path with maximum distance to the $R_m$.
$k$: Number of steps that define the neighbourhood under consideration.

/* Thresholds */
$T_p$ : Maximum distance to consider two paths as similar to one another.

$N$ = getKNeighbours($n_l$, $k$)
for all $n_n \in N$ do
    $P$ = getPaths($n_n, n_l$)
    if length($P$) $> 1$ then
        $M$ = array(length($P$))
        $D$ = array(length($P$))
        $R_m$ = computeModelPath(P)
        for $p \in$ indexOf($P$) do
            $M[p]$ = computeErasability($P[p]$)
            $R_p$ = computePathDescriptor($P[p]$)
            $D[p]$ = distance($R_m, R_p$)
        end for
        $P_{max}$ = computeMaxPath($D$)
        for $p \in$ indexOf($P$) do
            if $M[p]$ and $D[p] < T_p$ and (existNonErasable($P$) or $P[p] \neq P_{max}$) then
                deletePath($P[p]$)
            end if
        end for
    end if
end for
```

For each erasable path between the nodes, if the distance is below a threshold, this path is considered similar to the others and thus is regarded as redundant. If there is at least one non-erasable route between the nodes, the inner nodes belonging to the rest of erasable paths are deleted. Otherwise, if all paths are classified as erasable, the most different path (higher distance) is left unaltered, and the remaining ones are removed. The full algorithm for map refinement is outlined in Alg. 3.

Fig. 4 shows several examples of situations that our map refinement strategy is able to overcome. In (a), (b) and (c), the removed paths were selected because the distances to the model path are lower than the others. In (d), since there exists a non-erasable path and the distances of the other paths to the model are lower than T_p, both erasable routes are deleted. In (e), any of the routes can be deleted since they are non-erasable paths. In (f), there exists a non-erasable path and then, the erasable one could be deleted. However, since the distance of the path to the model is higher than T_p, indicating that this is a real alternative path, in this case the path can not be

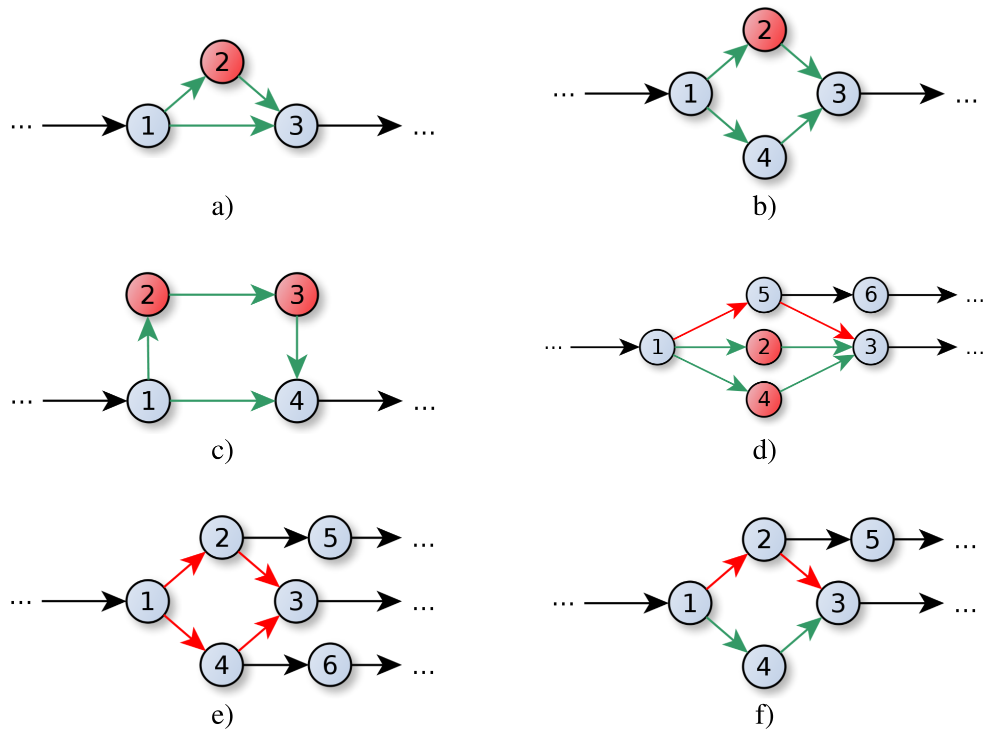


Figure 4: Examples of situations solved by our map refinement strategy. Green and red paths are, respectively, *erasable* and *non-erasable* paths. Red nodes indicate that they will be removed by our approach. When there are several erasable paths, the decision is taken according to the distance of the paths to the model path, as explained in the text.

removed.

4 Experimental Results

In this section, we will report about the results of several experiments, assessing our approach from different points of view. This section is organized as follows: first, the loop closure detection algorithm is evaluated irrespective of the mapping and localization process; then, results for the full approach for mapping and localization algorithm are shown; finally, experiments for validating our map refinement algorithm are reported.

4.1 Loop Closure Detection

Several experiments were carried out in order to validate the suitability of our framework for loop closure detection tasks. We processed datasets from indoor and outdoor environments, providing results under different environmental conditions. In the following, the main features of each dataset are presented first. Next, results from particular cases are highlighted. Finally, global results and considerations for the whole set of sequences are reported.

In more detail, the *Lip6Indoor* dataset comprises 388 images of two loops along the corridors of a research building, what leads to strong perceptual aliasing conditions during the loop closure analysis. The *Lip6Outdoor* is a longer dataset of 1063 images that completes a large loop outdoors under sunny weather conditions.

The *UIBSmallLoop* and *UIBLargeLoop* datasets were recorded by ourselves around the Anselm Turmeda building at our University campus. They consist of 388 and 997 images, respectively, taken under bad weather conditions, for verifying the performance of our approach under these situations. Finally, the *UIBIndoor* dataset, also recorded by ourselves inside the Anselm Turmeda building, presents an indoor environment which means a number of difficulties for loop closure. First of all, the camera velocity is not constant. This is due to the fact that we needed to climb up and down the stairs during the recording. This difficulty enables us to validate the capability of the filter to self-adapt under camera speed changes. Besides, a number of images of white walls result when the camera is at the stairs, what gives rise to the detection of very few features. Moreover, the dataset presents some parts with significative illumination changes, what leads on some occasions to overexposed images. Some examples of these problems are shown in Fig. 5.

Fig. 6 illustrates the performance of the observation likelihood for detecting loop closures within the Lip6Indoor dataset. The right picture shows the likelihood function values for every pair of frames I_i and I_j while the left picture is the ground truth. As can be seen, our likelihood presents high values for real loop closures, which are shown as diagonals in the images. There are more noise in the likelihood at the beginning of the sequence because there are less images in the trees, which implies that nearest neighbours for each descriptor are shared between a minor number of images. This effect decreases as we move forward along the sequence.

Fig. 7 shows the suitability of the Bayes framework in a loop closure detection situation. In this case, the camera visited twice the same place. When it returns to this place again, two high peaks corresponding to the previous visits can be observed in the likelihood, representing

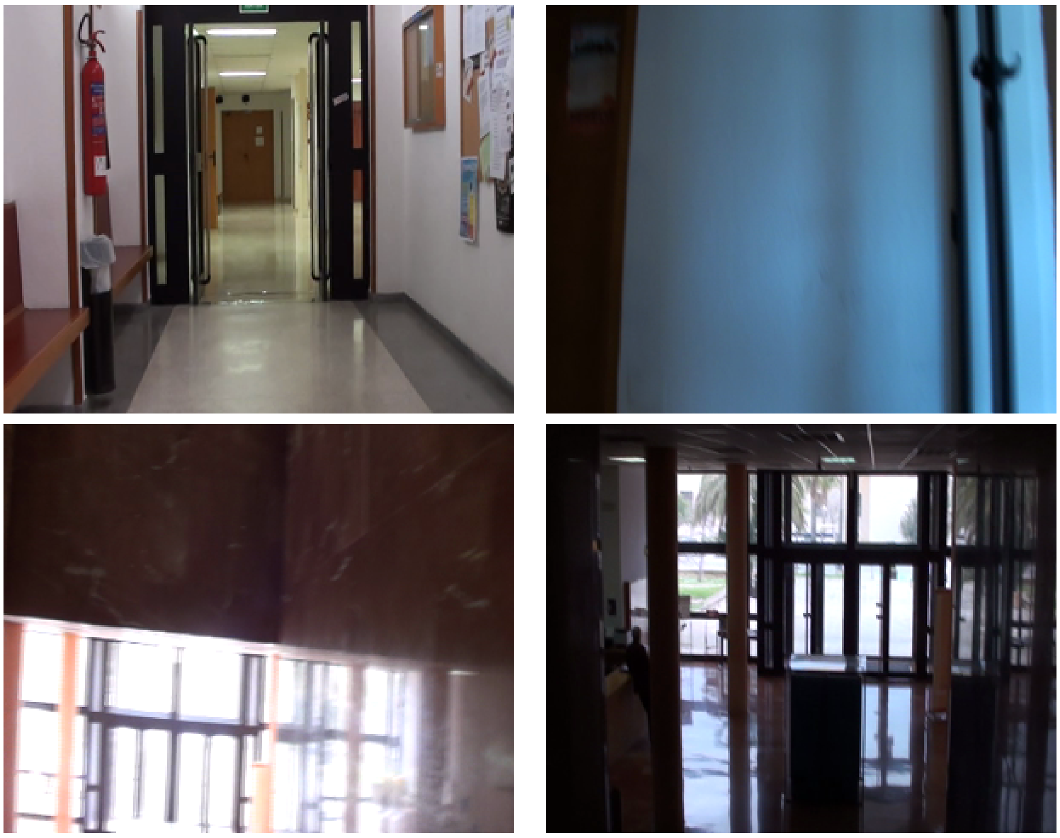


Figure 5: Examples of images from the UIBIndoor environment. (Top, Left) First image in the sequence. (Top, Right) Image taken from the stairs. (Bottom, Left) Overexposed image. (Bottom, Right) Image after camera stabilization.

possible loop candidates for the current image. After the prediction, update and normalization steps, the posterior presents only one single peak at the second candidate image, i.e. the filter ensures temporal coherency between predictions. This figure also shows an example of situation where a loop is detected despite there is a person in the image who was not in the previous visit, what proves the ability of the filter for detecting loops when the appearance of the environment changes. Our approach accepts the loop closure since the epipolar constraint between the two images is satisfied. It is also able to detect loop closures under camera rotations, as can be seen in Fig. 8.

If an overexposed image or with not enough features is considered by the filter, the full posterior does not present high peaks and a false negative is generated. However, as soon as the image stream becomes stable, the algorithm reacts and starts detecting loop closures again. This shows that our approach is able to manage these challenging kinds of situations.

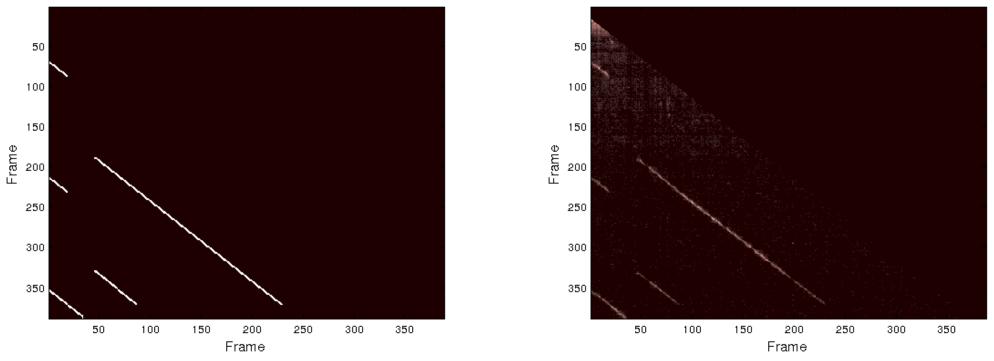


Figure 6: (Left) Ground truth loop closure matrix for the Lip6Indoor dataset. (Right) Likelihood matrix computed using our approach.

An example of loop closure detection for one of the UIB outdoor datasets under bad weather conditions and camera rotations can be found in Fig 9.

In order to obtain global performance measures, each dataset was provided with a ground truth, which indicates, for each image in the sequence, which images can be considered as a loop closure with it. The assessment against this ground truth has been 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, the two following metrics were computed:

- *Precision.* Ratio between real loop closures and total amount of loop closures detected $\left(\frac{TP}{TP+FP}\right)$.
- *Recall.* Ratio between real loop closures and total amount of loop closures existing in the sequence $\left(\frac{TP}{TP+FN}\right)$.

The results for each sequence are shown in Table I. As can be seen, no false positives resulted in any case. This is essential, since false positives can induce errors in mapping and localization tasks. As a consequence, the classifier always reaches 100% in precision for all datasets. The best recall rates for 100% precision are shown in the table.

As can be seen, a high rate of correct detections were obtained from all experiments. False negatives are due to, on the one hand, the sensitivity of the filter. In effect, when an old place is revisited, the likelihood associated to that hypothesis needs to be higher than the other likelihood

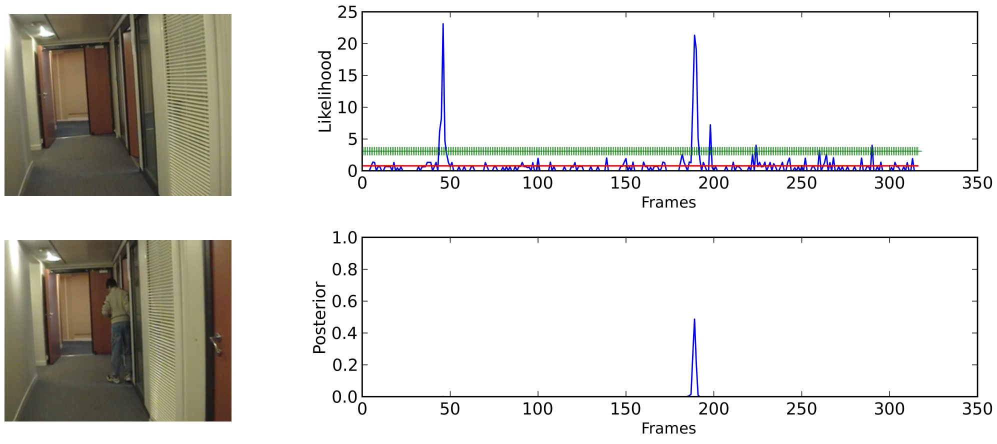


Figure 7: Example of loop closure detection visiting several times the same place and with changes in the environment in the Lip6Indoor dataset. Image 331 (Top, Left) closes a loop with image 189 (Bottom, Left) and image 48 (not shown). As can be seen in (Top, Right), the current likelihood presents two strong peaks despite a person in the current image occludes part of the same. Peaks correspond to loop candidates. After the normalization step, the posterior (Bottom, Right), shows a single peak in the last candidate frame. Red and green lines show respectively s_μ and $s_\mu + s_\sigma$ values.

			Our Approach						FAB-MAP	
Dataset	#Imgs	Size	TP	TN	FP	FN	Pr	Re	Pr	Re
Lip6Indoor	388	240×192	191	151	0	31	100	86	33	66
Lip6Outdoor	1063	240×192	551	435	0	52	100	91	98	13
UIBSmallLoop	388	300×240	194	172	0	2	100	99	98	28
UIBLargeLoop	997	300×240	439	491	0	47	100	90	97	19
UIBIndoor	384	300×240	157	177	0	30	100	84	88	8
	3220		1532	1426	0	162	100^a	90^a	83^a	27^a

Table I: Results for the five datasets using our approach and for FAB-MAP 2.0. Precision (Pr) and Recall (Re) columns are expressed as percentages. See text for details. [a]Average for all sequences.

values during several consecutive images in order to increase the posterior for this hypothesis. This introduces a delay in the loop closure detection, which derives in false negatives. This sensitivity can be tuned by modifying the transition model of the filter, although a higher sensitivity can introduce loop detection errors, i.e. false positives. On the other hand, false negatives can also be due to camera rotations. When the camera is turning around a corner, it is difficult to find and match features in the images, which prevents the hypothesis from satisfying the epipolar constraint and leads to the loop closure hypothesis to be rejected, despite the posterior for this image is higher than T_{loop}. However, in spite of the difficulties of the UIBIndoor dataset, our approach is able to succeed, as can be seen in Table I.

In order to validate the reliability of our loop closure algorithm against other existing solutions, we performed a comparison with the state-of-the-art FAB-MAP 2.0 algorithm [22], whose

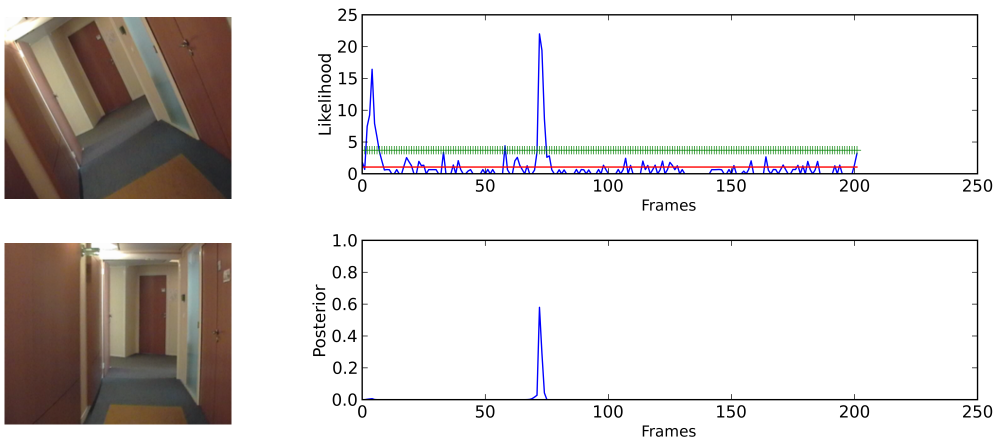


Figure 8: Example of loop closure detection under camera rotations. Despite there is a camera rotation, image 216 (Top, Left) closes a loop with image 72 (Bottom, Left). The likelihood (Top, Right) presents two high peaks since it is the third time the camera visits this place. (Bottom, Right) shows the final posterior, proving that the filter ensures the temporal coherency between loop detections. Red and green lines show respectively s_μ and $s_\mu + s_\sigma$ values.

binaries and visual vocabularies for indoors and outdoors are available online. We executed FAB-MAP configured with default parameters against the datasets, using the indoor vocabulary for Lip6Indoor and UIBIndoor sequences, and the outdoor vocabulary for the rest. The output is a matrix, the n-th entry 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, the output matrix was rectified by removing the most recent probabilities for each row and normalizing the final distribution. A loop closure is detected if the probability is above a predefined threshold p. The output loop detections of FAB-MAP are compared against the ground truth of each sequence and measured in terms of precision-recall. The final results are shown graphically in Fig 10, varying the p parameter. The curves for our approach were plotted modifying the threshold for loop acceptance (T_{loop}). Clearly, our approach outperforms FAB-MAP in all datasets, obtaining a higher recall for a 100% of precision. As can be seen, our solution is also more stable, specially in indoor environments, where the performance of FAB-MAP decreases dramatically. We think this is due to the use of the indoors vocabulary and the complexity of finding features in these kinds of environments. As a further benefit, our approach can deal better with the sensor noise. Notice that a precision below 100%

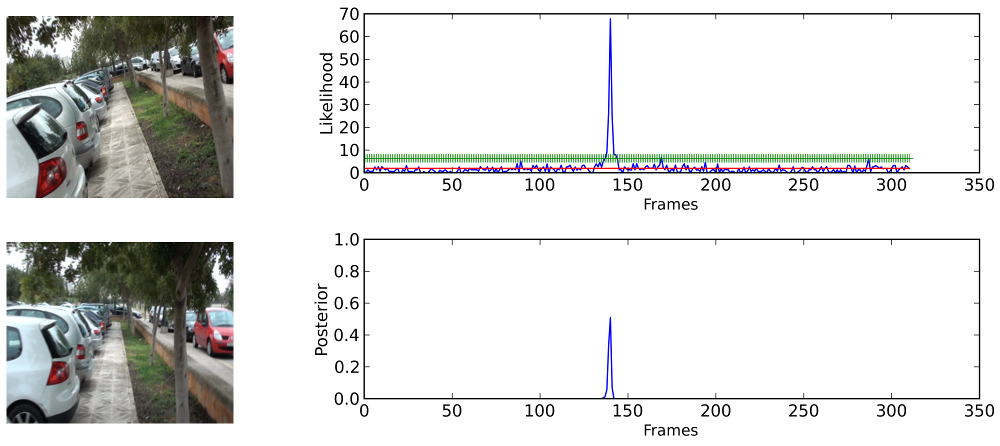


Figure 9: Example of loop closure detection under bad weather conditions and camera rotations for the UIBSmallLoop dataset. Image 330 (Top, Left) closes a loop with image 139 (Bottom, Left). (Top, Right) Likelihood given the current image. (Bottom, Right) Full posterior after the normalization step. Red and green lines show respectively s_μ and $s_\mu + s_\sigma$ values.

implies the presence of false positives, what will have an influence over the generated map. Our algorithm allows us to obtain a higher recall than FAB-MAP maintaining the maximum precision possible. The maximum precision of FAB-MAP together with its recall for each dataset is also shown in Table I.

The paths followed by the camera in the UIB datasets are shown in Fig 11 and Fig 12. Whenever the camera explores new places, no loop closures are found. When a place is revisited, the algorithm starts to find loop closures. Several images are usually needed until closing the loop, due to the filter inertia. These images correspond to the false negatives found.

4.2 Mapping and Localization

The same sequences used in the previous experiments were also employed to validate our framework regarding mapping and localization. To this end, the loop closure detection algorithm was adapted to be used with the detected keyframes. A real map of the environment and the topological map generated by our approach are shown for each sequence. The main zones of these maps were labelled with letters to simplify the identification of each part in the topological structure, since these maps do not preserve the shape. The results are shown from Fig. 13 to 18.

As can be seen, the maps generated by our framework represent topologically the real scenario. Connections between each part of the topological map are the same of the real environment, and

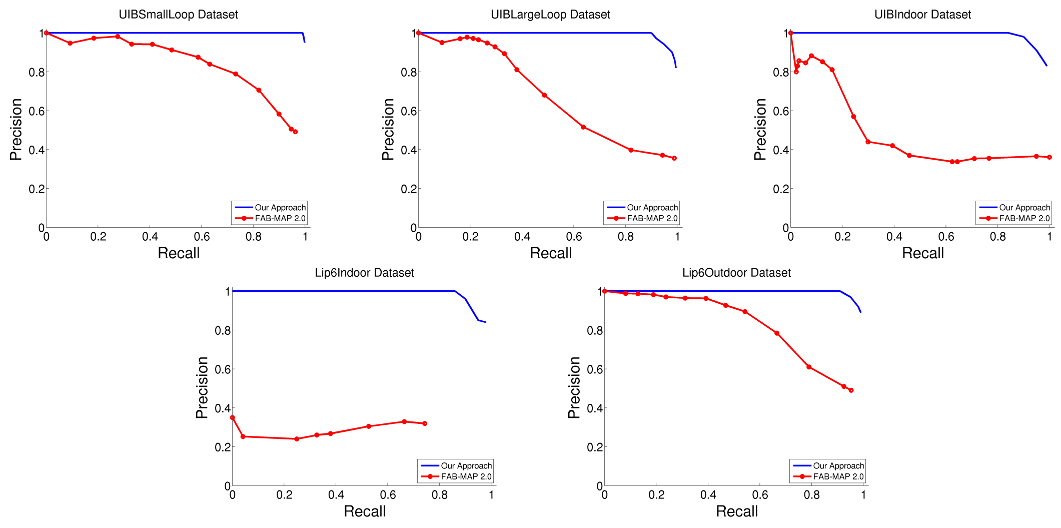


Figure 10: Precision-recall curves for each dataset using our approach and FAB-MAP 2.0.

the maps do not contain redundant paths or spurious nodes between locations, saving storage space and improving the computational efficiency of the localization process. Therefore, we can conclude that our map refinement strategy helps us to clean the final structure, correcting the problems generated by the blurred images and the delays inherent to the loop closure detection process.

Maps are mainly created during the first exploration of the environment, so that revisiting a place normally turns into reassigning the current location of the robot to an existing node of the map. However, sometimes maps are completed with new nodes corresponding to images which are visually in-between two nodes. Generally, they provide unregistered information about the robot scenario, as can be seen for example in Fig. 16.

To finish, it is typical that a few nodes at the beginning of the sequence do not close any loop, generating a short tail in the map. This is due to the prediction of the Bayes filter, which tends to move the probability away from the beginning of the sequence, producing that the first loop is closed with the subsequent frames. Notice that this, however, does not affect the final result of the localization process.

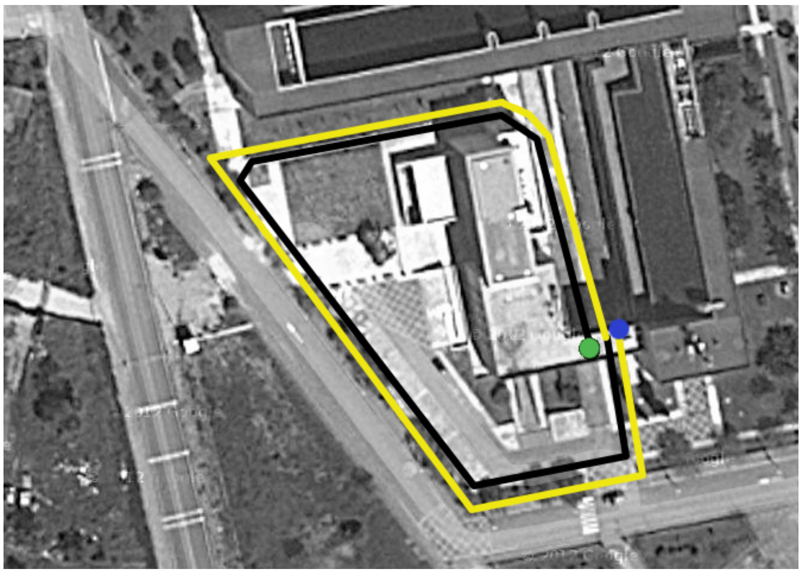


Figure 11: Path followed by the camera during the UIBSmallLoop experiment. Green and blue points indicate respectively the beginning and the end of the sequence; the black lines show no loop closure detections (highest posterior probability is under T_{loop}) and the yellow lines represent loop closure detections (highest probability is above T_{loop} and the epipolar constraint is satisfied). Notice that the camera passes through the same place in successive loops, but the lines are drawn in parallel for visualization purposes.

4.3 Map Refinement

The main goal of this last section is to verify the quality of the refined maps, in terms of storage space, computational times and usefulness/efficiency. We want to assess that the generated maps are representative of the environment and can be used for localization without compromising the original performance. To this end, we compare the maps of the five datasets used in this work with and without refinement. The former appear from Fig. 13 to 18, while the latter are shown from Fig. 19 to 23. As can be seen, the original maps contain spurious nodes and alternative redundant paths between nodes, incrementing the execution time of mapping and localization processes, since more nodes need to be considered at each step. The refined maps shown in section 4.2 present better the environment.

An additional experiment was performed in order to verify whether refined maps could be employed for localization with a similar performance to the original ones. To this end, we first generated the map of the environment and, for each image, the assigned keyframe was stored. After that, the sequence was processed again using the localization filter to determine, for each image, the closest location in the map. If that location was the same as the one stored during the

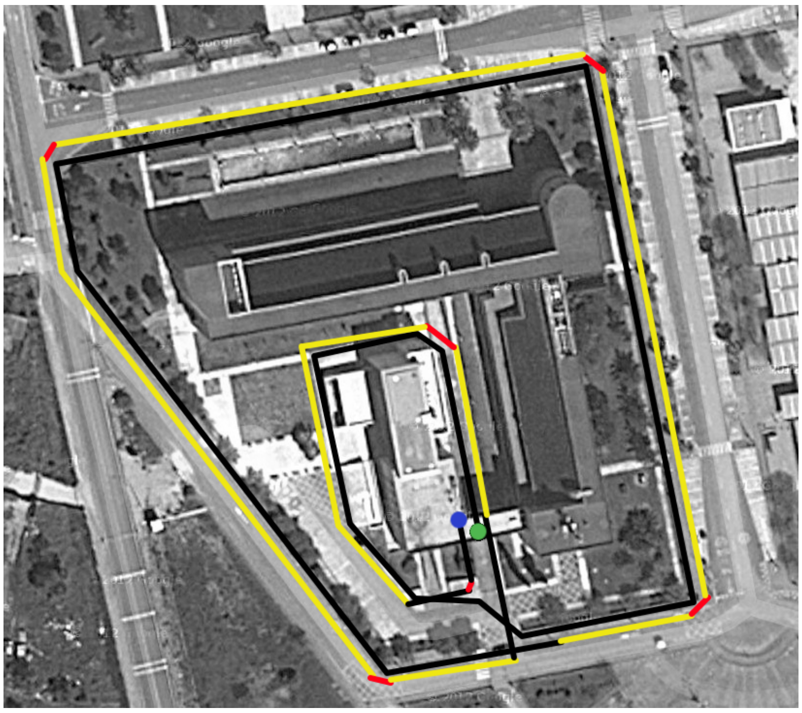


Figure 12: Path followed by the camera during the UIBLargeLoop experiment. Green and blue points indicate respectively the beginning and the end of the sequence; the black lines show no loop closure detections (highest posterior probability is under T_{loop}), the red lines show rejected hypoteses (no epipolar geometry is satisfied) and the yellow lines represent loop closure detections (highest probability is above T_{loop} and the epipolar constraint is satisfied). Notice that the camera passes through the same place in successive loops, but the lines are drawn in parallel for visualization purposes.

mapping process, the image was considered as a correct localization (CL). For each sequence, we also obtained the total mapping and localization times, as well as the number of nodes generated in the graph. These values were measured for each sequence with and without the refinement step. The results can be found in Table II. As it is shown, map refinement leads to less nodes than without it. Despite the correct localization rate is slightly lower for some environments, refining the map improves the computational times of the mapping and localizations processes. This effect increases with the length of the sequence, as is the case of the Lip6Outdoor and the UIBLargeLoop datasets. The UIBSmallLoop dataset presents small differences between the two versions of the map. This is because the resulting structures in the maps are practically the same, resulting into similar processing times. From the table we can also observe that, in general, the outdoor environments are more affected by the refinement step, since the correct localization rates are lower for these cases. In general terms, we can argue that the map refinement strategy proposed

	With Map Refinement				Without Map Refinement			
Dataset	N	M	L	%CL	N	M	L	%CL
Lip6Indoor	40	137.37	6.27	64	62	155.36	9.08	63
Lip6Outdoor	103	1005.52	29.05	63	141	1150.87	42.95	61
UIBSmallLoop	59	152.2	8.25	75	61	155.3	8.57	77
UIBLargeLoop	100	728.94	44.29	74	111	798.53	60.98	78
UIBIndoor	40	118.4	16.2	76	59	190.39	24.64	73

Table II: Results for the map refinement experiment. *N*: number of nodes; *M*: mapping time in seconds; *L*: localization time in seconds; *%CL*: ratio between correct localizations and total number of elements. See text for details.

in this work can be used for saving space in memory and for improving the speed of the mapping and localization tasks without compromising the performance.

5 Conclusions and Future Work

A complete appearance-based mapping and localization framework based on local invariant features is presented here. When a new useful image is acquired, a discrete Bayes filter is used to select a loop closure candidate and decide whether this frame is a loop closure or a new node to be added to the map. This probabilistic filter presents a novel observation model based on an efficient matching scheme between the current image and the features of the current nodes in the map, using an index based on a set of randomized kd-trees. As a result, a topological map of the environment is obtained, which represents the scenario of the robot as a graph.

Using probabilistic filters for mapping and localization tasks usually produces spurious nodes and redundant paths over the graph. This is due to imperfections in the acquired images and the delays introduced by the filter. A key contribution of this work is a map refinement strategy for solving these problems, producing cleaner maps and saving storage space and computation resources for the mapping and localization tasks. This framework is executed each time a loop is closed, and a predefined neighbourhood is refined in each step. The final decision of deleting nodes is taken according to the visual features of each path, avoiding to delete the real paths of the environment.

In order to validate our solution, results from an extensive set of experiments, using datasets from different environments, have been reported. These results are very promising, showing that our mapping and localization approach using a map refinement phase can be employed for generating topological maps of the environment that, if they are provided with odometry information, can also be used for navigating in the current scenario in an efficient way. We also compared our approach with the state-of-the-art FAB-MAP 2.0 algorithm.

Referring to future work, we intend to explore: (a) the use of other kinds of image descriptors based on local invariant features, such as binary descriptors, since they can improve our approach in computational terms; (b) the execution of the Bayes filter in a Graphics Processing Unit (GPU) to further speed up the loop closure detection; and (c) the use of the full algorithm for mapping larger environments, since unused descriptors in the tree structures should be purged to maintain a reasonable response time of the loop closure detection algorithm.

References

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms," *Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[2] J. Sivic and A. Zisserman, "Video Google: A Text Retrieval Approach to Object Matching in Videos," in *International Conference on Computer Vision*, pp. 1470–1477, 2003.

[3] H. Zhang, "BoRF: Loop-Closure Detection with Scale Invariant Visual Features," in *International Conference on Robotics and Automation*, pp. 3125–3130, 2011.

[4] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, "Incremental Vision-Based Topological SLAM," in *International Conference on Intelligent Robots and Systems*, pp. 22–26, 2008.

[5] O. Booij, B. Terwijn, Z. Zivkovic, and B. Krose, "Navigation Using an Appearance Based Topological Map," in *International Conference on Robotics and Automation*, no. April, pp. 3927–3932, 2007.

[6] E. Garcia-Fidalgo and A. Ortiz, "Probabilistic Appearance-Based Mapping and Localization Using Visual Features," in *Pattern Recognition and Image Analysis*, vol. 7887 of *Lecture Notes in Computer Science*, pp. 277–285, 2013.

[7] Z. Zivkovic, B. Bakker, and B. Krose, "Hierarchical Map Building Using Visual Landmarks and Geometric Constraints," in *International Conference on Intelligent Robots and Systems*, pp. 2480–2485, 2005.

[8] I. Ulrich and I. Nourbakhsh, "Appearance-Based Place Recognition for Topological Localization," in *International Conference on Robotics and Automation.*, pp. 1023–1029, 2000.

[9] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool, "Markerless Computer Vision Based Localization using Automatically Generated Topological Maps," in *European Navigation Conference*, pp. 235–243, 2004.

[10] D. G. Sabatta, "Vision-based Topological Map Building and Localisation using Persistent Features," in *Robotics and Mechatronics Symposium*, pp. 1–6, 2008.

[11] A. Kawewong, N. Tongprasit, S. Tungruamsub, and O. Hasegawa, "Online and Incremental Appearance-Based SLAM in Highly Dynamic Environments," *International Journal of Robotics Research*, vol. 30, no. 1, pp. 33–55, 2011.

[12] G. Singh and J. Kosecka, "Visual Loop Closing using Gist Descriptors in Manhattan World," in *Omnidirectional Robot Vision workshop, held with IEEE ICRA*, 2010.

[13] A. C. Murillo, C. Sagues, and J. Guerrero, "From Omnidirectional Images to Hierarchical Localization," *Robotics and Autonomous Systems*, vol. 55, pp. 372–382, May 2007.

[14] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[15] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, "Real-Time Visual Loop-Closure Detection," in *International Conference on Robotics and Automation*, pp. 1842–1847, 2008.

[16] 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 Transactions on Robotics*, vol. 24, no. 5, pp. 1027–1037, 2008.

[17] H. Zhang, "Indexing Visual Features: Real-Time Loop Closure Detection Using a Tree Structure," in *International Conference on Robotics and Automation*, pp. 3613–3618, 2012.

[18] F. Fraundorfer, C. Engels, and D. Nister, "Topological Mapping, Localization and Navigation Using Image Collections," in *International Conference on Intelligent Robots and Systems*, pp. 3872–3877, 2007.

[19] E. Johns and G.-Z. Yang, "Feature Co-Occurrence Maps: Appearance-Based Localisation Throughout the Day," in *International Conference on Robotics and Automation*, pp. 3212–3218, 2013.

[20] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," in *International Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 2161–2168, 2006.

[21] M. Cummins and P. Newman, "Accelerating FAB-MAP With Concentration Inequalities," *IEEE Transactions on Robotics*, vol. 26, no. 6, pp. 1042 –1050, 2010.

[22] M. Cummins and P. M. Newman, "Appearance-Only SLAM at Large Scale with FAB-MAP 2.0," *International Journal of Robotics Research*, vol. 30, no. 9, pp. 1100–1123, 2011.

[23] A. J. Glover, W. P. Maddern, M. Milford, and G. F. Wyeth, "FAB-MAP + RatSLAM: Appearance-Based SLAM for Multiple Times of Day," in *International Conference on Robotics and Automation*, pp. 3507–3512, IEEE, 2010.

[24] D. Filliat, "A Visual Bag of Words method for Interactive Qualitative Localization and Mapping," in *International Conference on Robotics and Automation*, pp. 3921–3926, 2007.

[25] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, “Visual Topological SLAM and Global Localization,” in *International Conference on Robotics and Automation*, pp. 4300–4305, 2009.

[26] T. Nicosevici and R. García, “On-line Visual Vocabularies for Robot Navigation and Mapping,” in *International Conference on Intelligent Robots and Systems*, pp. 205–212, 2009.

[27] A. Oliva and A. Torralba, “Modeling the Shape of the Scene : A Holistic Representation of the Spatial Envelope,” *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[28] Y. Liu and H. Zhang, “Visual Loop Closure Detection with a Compact Image Descriptor,” in *International Conference on Intelligent Robots and Systems*, pp. 1051–1056, 2012.

[29] C. Siagian and L. Itti, “Rapid Biologically-Inspired Scene Classification using Features Shared with Visual Attention,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 300–12, 2007.

[30] N. Sünderhauf and P. Protzel, “BRIEF-Gist - Closing the Loop by Simple Means,” in *International Conference on Intelligent Robots and Systems*, pp. 1234–1241, 2011.

[31] D. M. Bradley, R. Patel, N. Vandapel, and S. Thayer, “Real-Time Image-Based Topological Localization in Large Outdoor Environments,” in *International Conference on Intelligent Robots and Systems*, pp. 3670–3677, 2005.

[32] C. Weiss, A. Masselli, H. Tamimi, and A. Zell, “Fast Outdoor Robot Localization using Integral Invariants,” in *International Conference on Computer Vision Systems*, p. 24, 2007.

[33] J. Wang, H. Zha, and R. Cipolla, “Efficient Topological Localization Using Orientation Adjacency Coherence Histograms,” in *International Conference on Pattern Recognition*, pp. 271–274, IEEE Computer Society, 2006.

[34] H. Badino, D. F. Huber, and T. Kanade, “Real-Time Topometric Localization,” in *International Conference on Robotics and Automation*, pp. 1635–1642, IEEE, 2012.

[35] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[36] C. Valgren, T. Duckett, and A. Lilienthal, “Incremental Spectral Clustering and Its Application to Topological Mapping,” *International Conference on Robotics and Automation*, no. April, pp. 10–14, 2007.

[37] B. Bacca, J. Salvi, J. Batlle, and X. Cufi, “Appearance-Based Mapping and Localisation Using Feature Stability Histograms,” *Electronics Letters*, vol. 46, no. 16, pp. 1120–1121, 2010.

[38] J. L. Blanco, J. A. Fernandez-Madrigal, and J. Gonzalez, “Towards a Unified Bayesian Approach to Hybrid Metric-Topological SLAM,” *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 259–270, 2008.

[39] H. Zhang, B. Li, and D. Yang, “Keyframe Detection for Appearance-Based Visual SLAM,” in *International Conference on Intelligent Robots and Systems*, pp. 2071–2076, 2010.

[40] S. Leutenegger, M. Chli, and R. Siegwart, “BRISK: Binary Robust Invariant Scalable Keypoints,” *International Conference on Computer Vision*, vol. 0, pp. 2548–2555, 2011.

[41] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An Efficient Alternative to SIFT or SURF,” in *International Conference on Computer Vision*, pp. 2564–2571, 2011.

[42] A. Alahi, R. Ortiz, and P. Vandergheynst, “FREAK: Fast Retina Keypoint,” in *International Conference on Computer Vision and Pattern Recognition*, pp. 510–517, 2012.

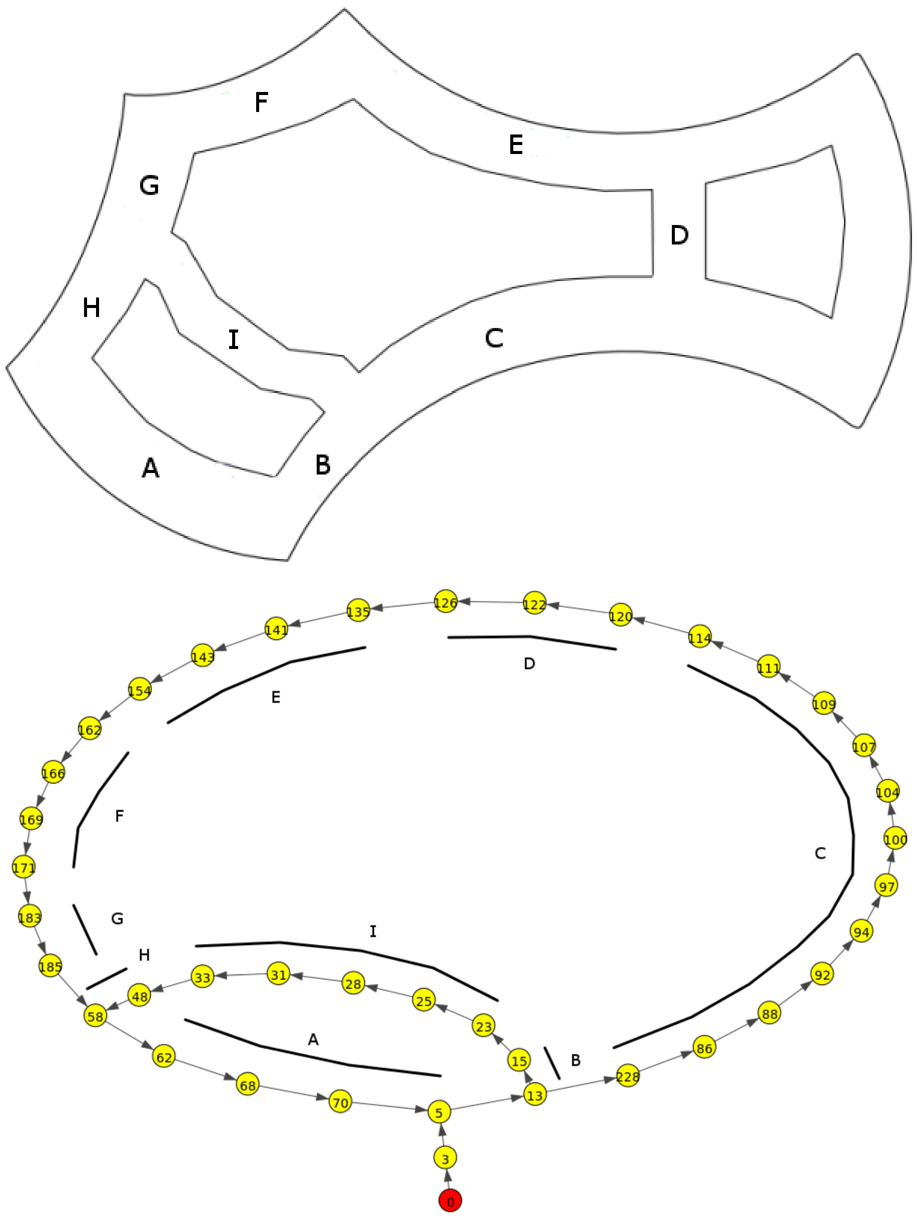


Figure 13: (Top) Reference map for the Lip6Indoor dataset. (Bottom) Topological map generated using our approach. Each part of the map is identified with a letter in both maps. The red node identifies the beginning of the sequence. The reference map comes from http://cogrob.ensta-paristech.fr/loopclosure.html. Maps locations are visited in the following order: A-B-I-H-A-B-C-D-E-F-G-H-A-B-C-D-E-F-G-H-A-B-I.

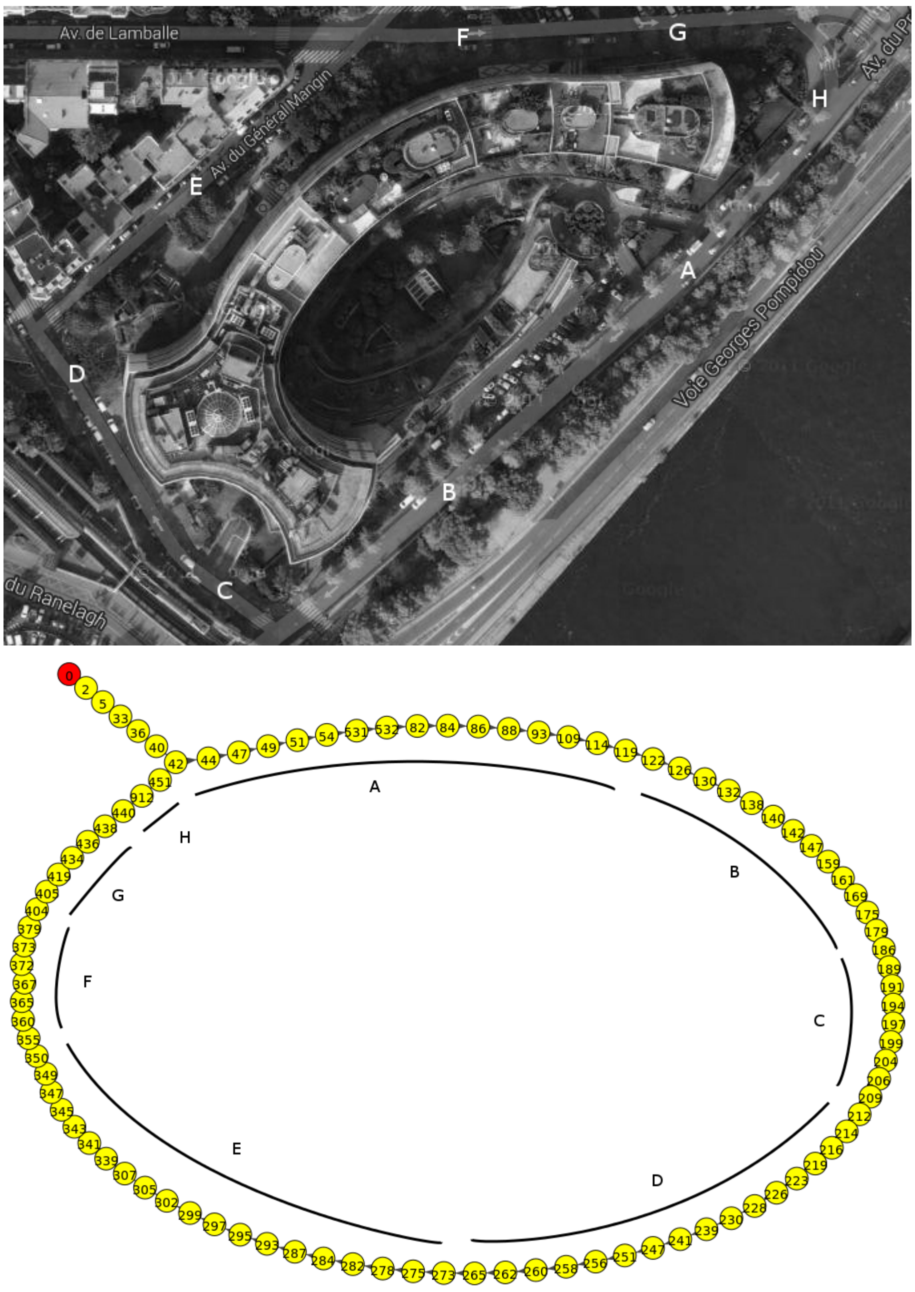


Figure 14: (Top) Reference map for the Lip6Outdoor dataset. (Bottom) Topological map generated using our approach. Each part of the map is identified with a letter in both maps. The red node identifies the beginning of the sequence. Maps locations are visited in the following order: A-B-C-D-E-F-G-H-A-B-C-D-E-F-G-H-A-B

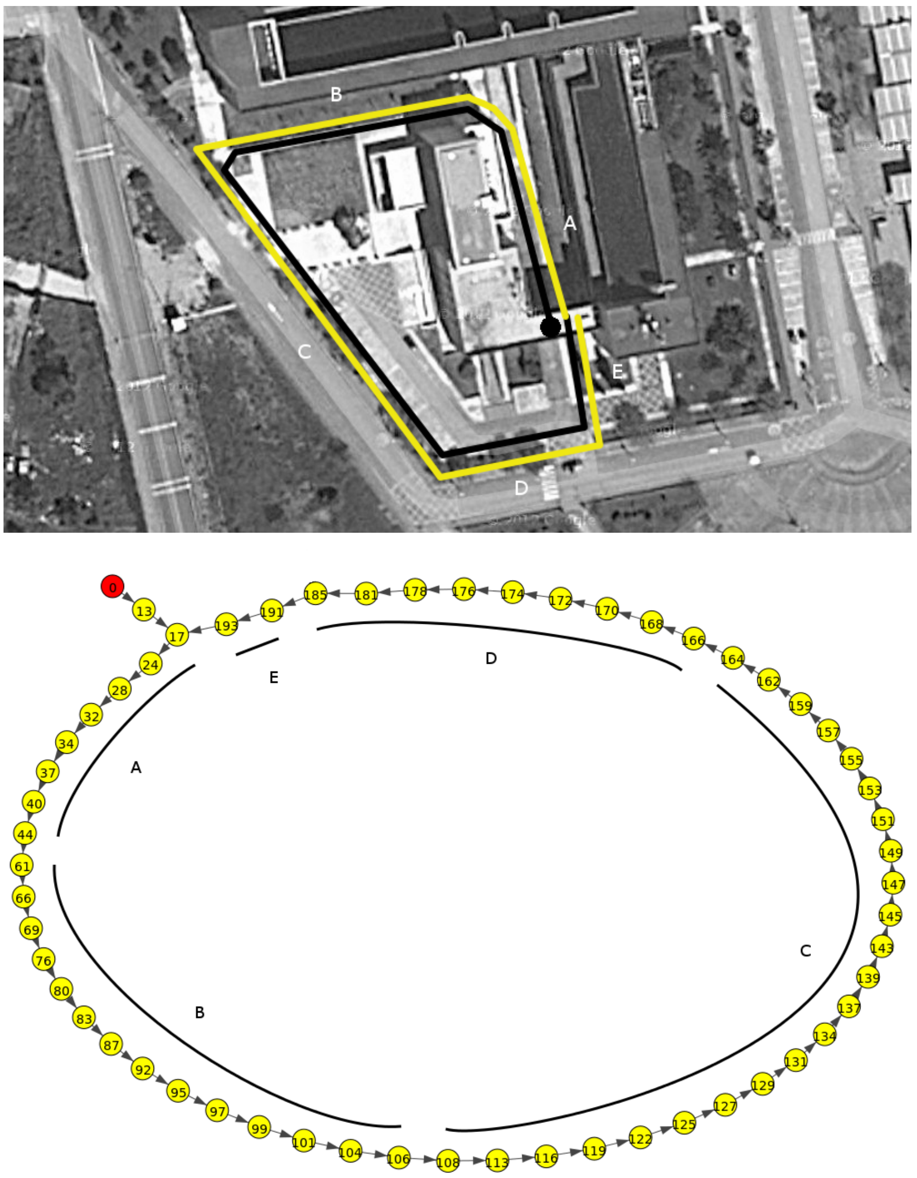


Figure 15: (Top) Reference map for the UIBSmallLoop dataset. (Bottom) Topological map generated using our approach. Each part of the map is identified with a letter in both maps. The red node identifies the beginning of the sequence. Maps locations are visited in the following order: A-B-C-D-E-A-B-C-D-E.

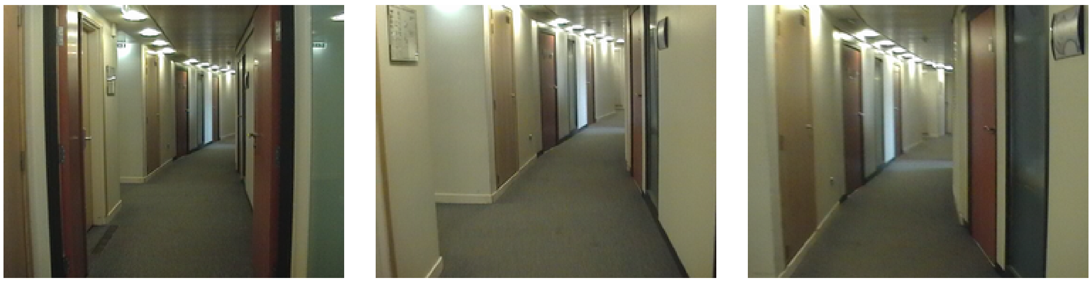
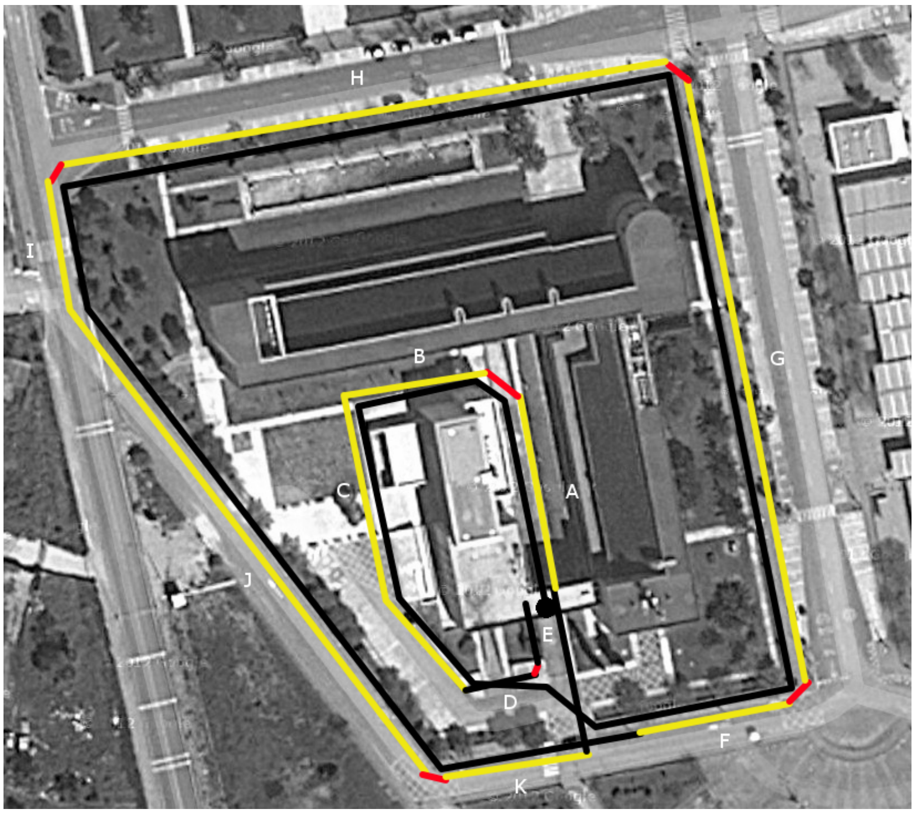


Figure 16: Example of adding intermediate nodes in the Lip6Indoor dataset. Images 13 (Left) and 86 (Right) were added to the map at the first loop. Image 228 (Center) was added to the map the next time the camera visits the same place. As can be seen, image 228 is visually in-between the left and right images.



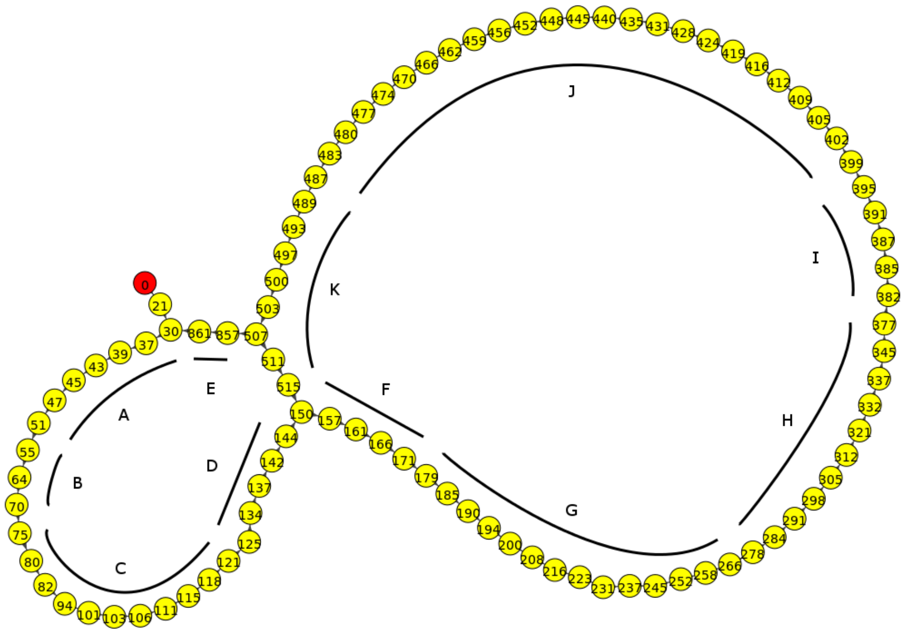


Figure 17: (Top) Reference map for the UIBLargeLoop dataset. (Bottom) Topological map generated using our approach. Each part of the map is identified with a letter in both maps. The red node identifies the beginning of the sequence. Maps locations are visited in the following order: A-B-C-D-F-G-H-I-J-K-F-G-H-I-J-K-E-A-B-C-D-E.

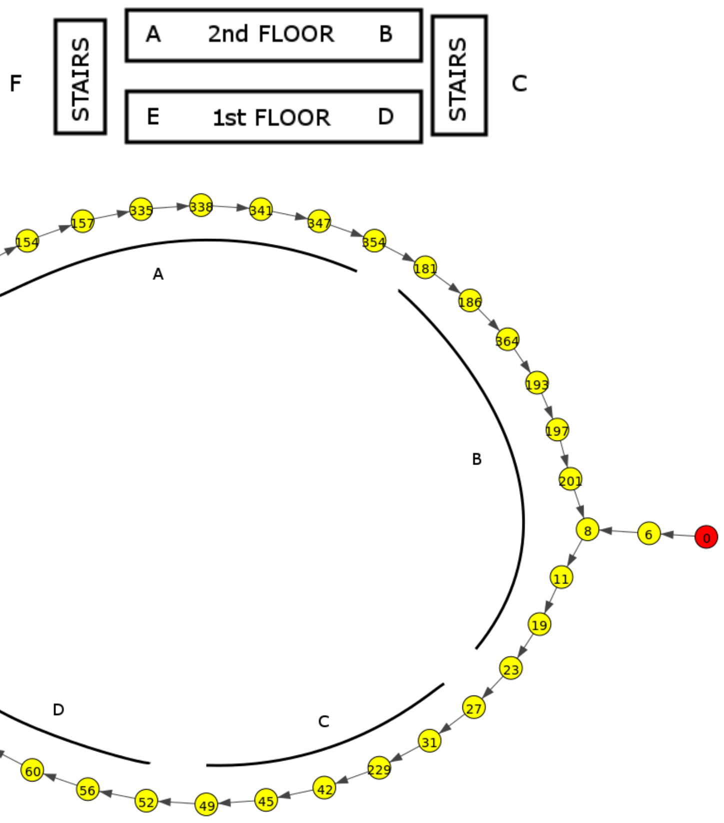


Figure 18: (Top) Reference map for the *UIBIndoor* dataset. (Bottom) Topological map generated using our approach. Each part of the map is identified with a letter in both maps. The red node identifies the beginning of the sequence. Maps locations are visited in the following order: B-C-D-E-F-A-B-C-D-E-F-A-B.

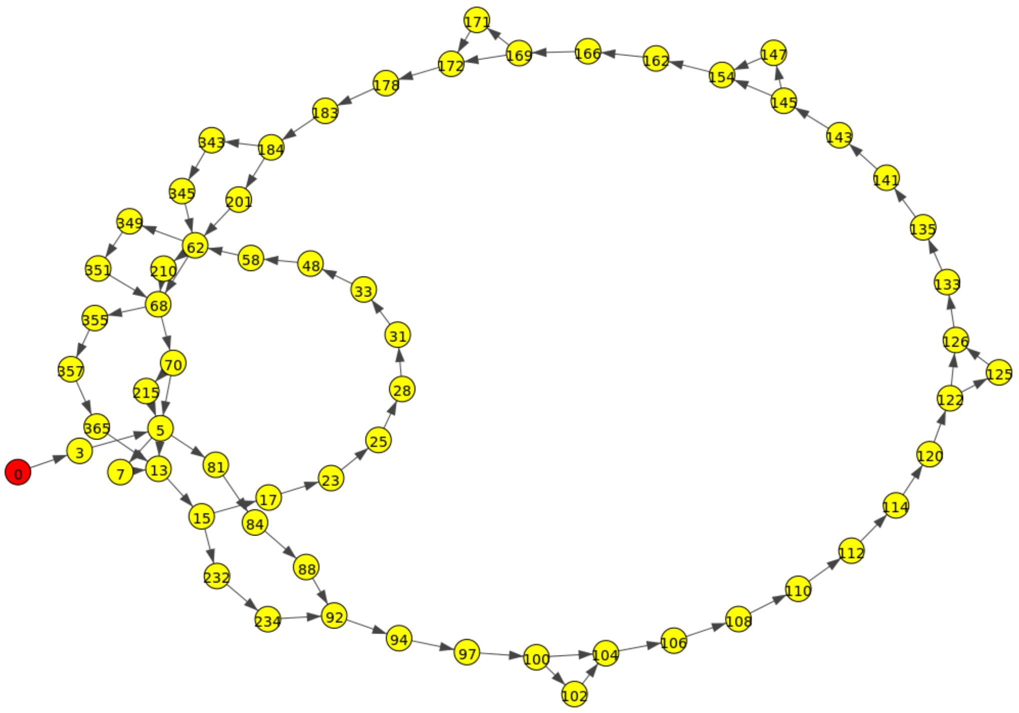


Figure 19: Map of the Lip6Indoor dataset obtained without using the map refinement strategy. The red node identifies the beginning of the sequence.

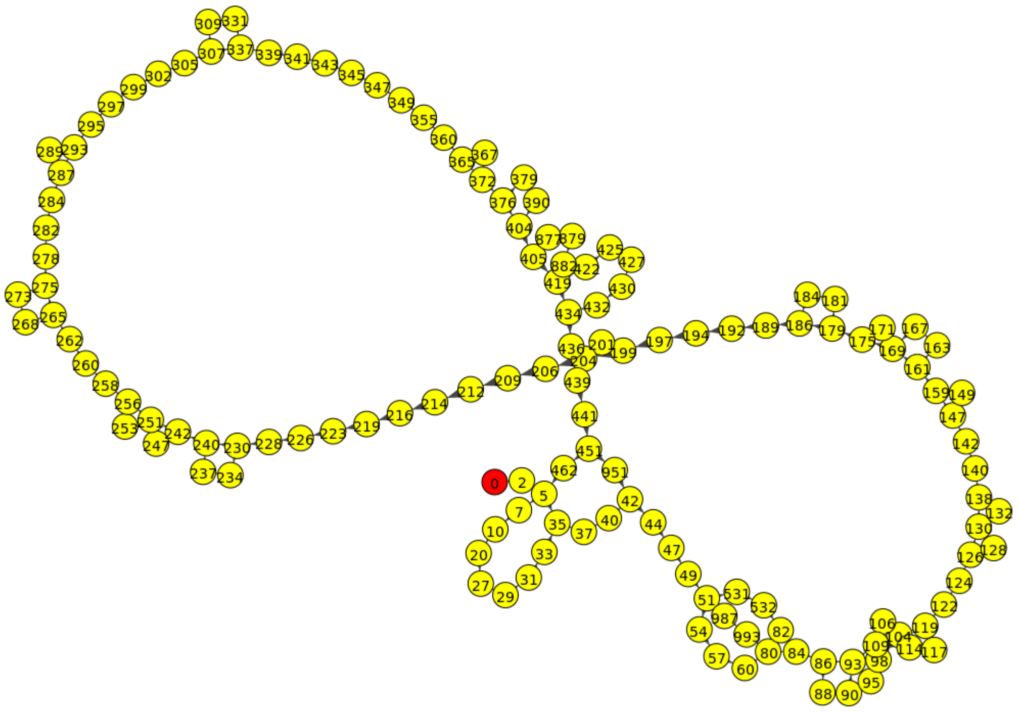


Figure 20: Map of the Lip6Outdoor dataset obtained without using the map refinement strategy. The red node identifies the beginning of the sequence.

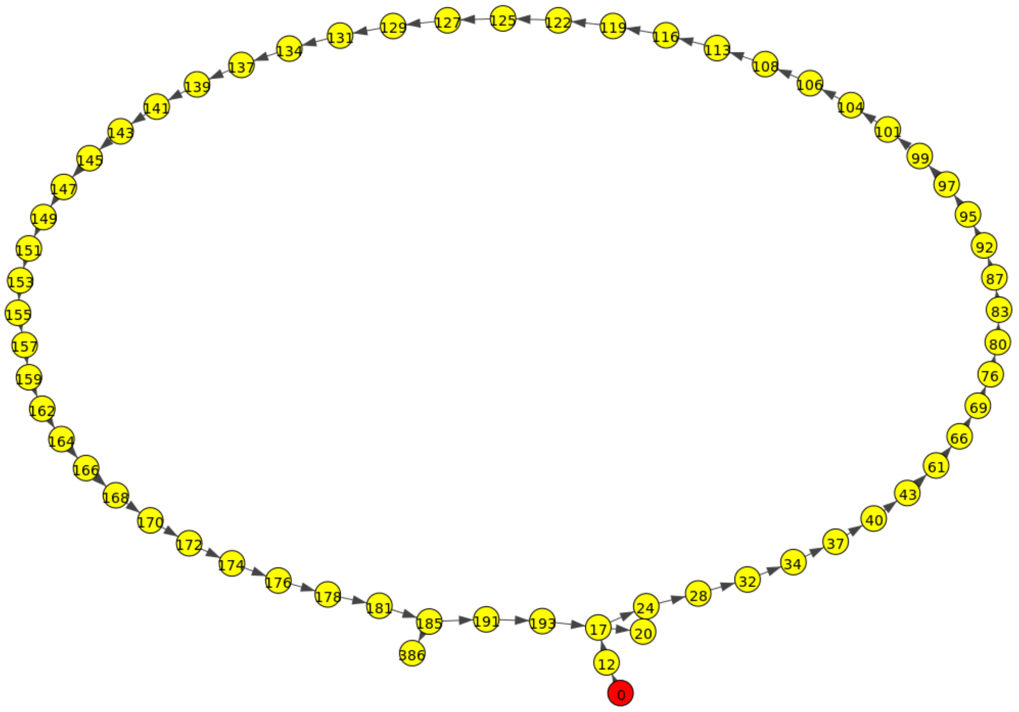


Figure 21: Map of the UIBSmallLoop dataset obtained without using the map refinement strategy. The red node identifies the beginning of the sequence.

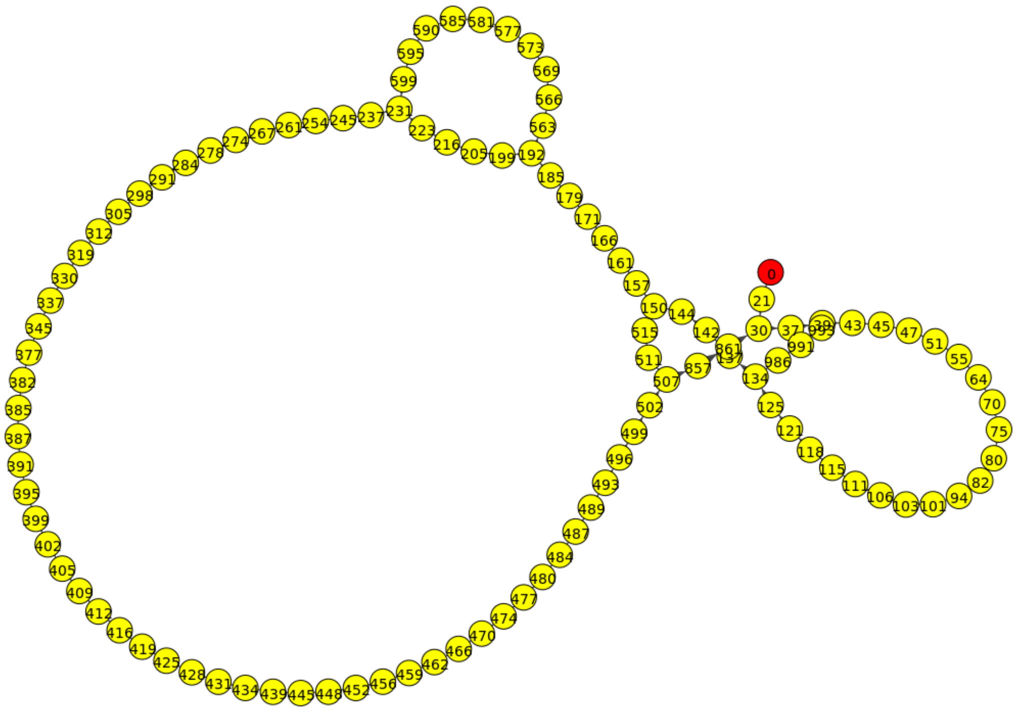


Figure 22: Map of the UIBLargeLoop dataset obtained without using the map refinement strategy. The red node identifies the beginning of the sequence.

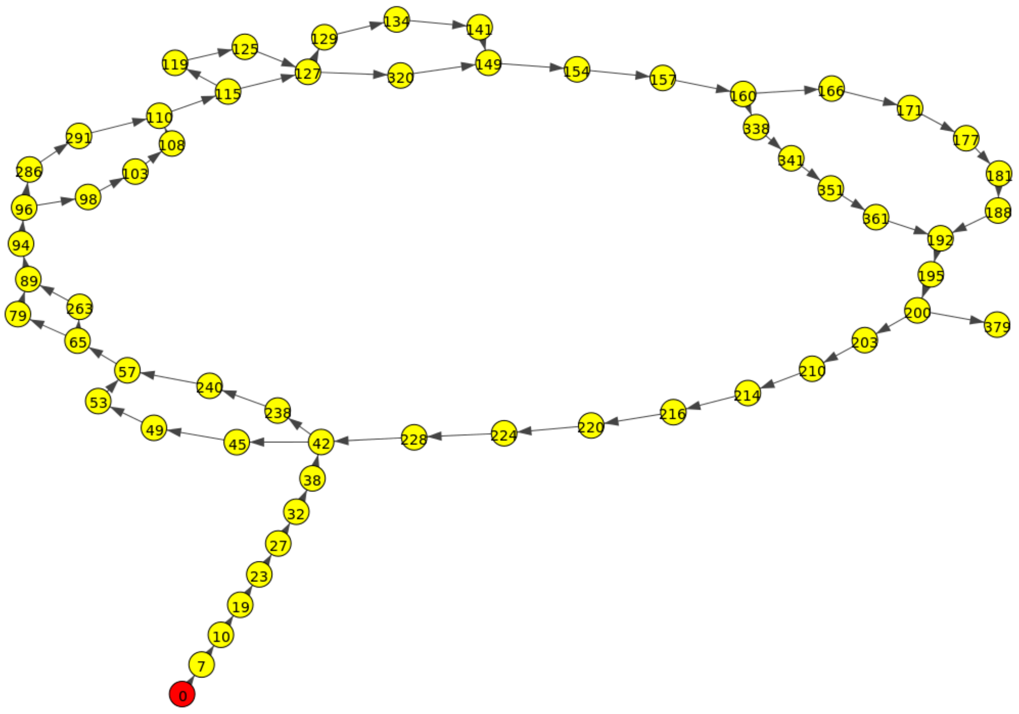


Figure 23: Map of the UIBIndoor dataset obtained without using the map refinement strategy. The red node identifies the beginning of the sequence.