Real-Time Loop Detection with Bags of Binary Words

Dorian Gálvez-López, Juan D. Tardós

***Abstract*— We present a method for detecting revisited places in a image sequence in real time by using efficient features. We introduce three important novelties to the bag-of-words plus geometrical checking approach. We use FAST keypoints and BRIEF descriptors, which are binary and very fast to compute (less that 20μs per point). To perform image comparisons, we make use of a bag of words that discretises the binary descriptor space and an inverse index. We also introduce the use of a direct index to take advantage of the bag of words to obtain correspondence points between two images efficiently, avoiding a matching of complexity $\Theta(n^2)$. To detect loop closure candidates, we propose managing matches in groups to increase the reliability of the candidates returned by the bag of words. We present results in three real and public datasets, with 0.7–1.7 Km long trajectories. We obtain high precision and recall rates, spending 16 ms on average per image for the feature computation and the whole loop detection process in sequences with 19000 images, one order of magnitude less than other similar techniques.**

I. Introduction

When a mobile robot traverses an environment, the challenge to notice if a place has already been visited is called *loop closure detection*. It is an important task that must be solved by a *Simultaneous Localisation and Mapping* (SLAM) algorithm to produce consistent maps. Cameras are nowadays a cheap sensor that produce very rich information, so that more and more robots are provided with them. For this reason, there has been a big focus of the community on appearance-based algorithms to detect loop closures. The basic technique consists in adding the images the robot collects online to a database, so that when a new image is acquired, the most similar one from the database is retrieved. If they are similar enough, a loop closure is detected.

In recent years, many algorithms that exploit this idea have appeared ([1], [2], [3], [4], [5]), basing the image matching on comparing them in the bag-of-words space [6]. A bag of words is a structure that allows to represent an image as a single numerical vector. This makes it possible to perform comparisons with thousands of images in dozens of milliseconds [7]. Bags of words result in very effective image matchers, but they are not a perfect solution for closing loops, due mainly to perceptual aliasing [5]. For this reason, a verification step is performed later by checking the matching images to be geometrically consistent, requiring feature correspondences.

Dorian Gálvez-López and Juan D. Tardós are with the Instituto de Investigación en Ingeniería de Aragón (I3A), Universidad de Zaragoza, Zaragoza 50018, Spain. {dorian, tardos}@unizar.es.

This research has been partly funded by the European Union under project RoboEarth FP7-ICT-248942, the Dirección General de Investigación of Spain under projects DPI2009-13710, DPI2009-07130 and the Ministerio de Educación (scholarship FPU-AP2008-02272).

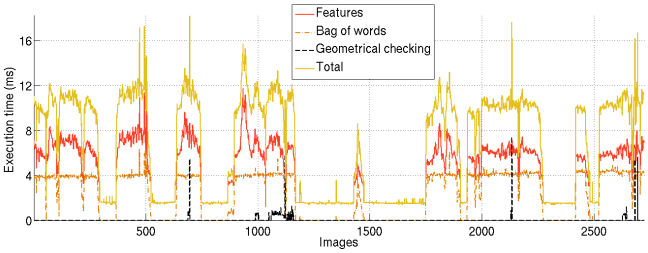


Fig. 1. Execution time of the proposed system in a trajectory with 2723 images, obtaining 100% precision and 57.86% recall.

The bottleneck of the loop closure algorithms is usually the extraction of features, that is around ten times more expensive in computation cycles than the rest of steps. In addition, the dimension of the descriptor vectors can be high, and this may slow down the feature matching. All this may cause SLAM algorithms to run in two decoupled threads: one to perform the main SLAM functionality, and the other just to detect loop closures and to calculate the data association, as in [4].

In this paper, we present a fast algorithm to detect loops and establishing correspondences between matching scenes in real time, with a usual CPU and a single camera. It is based on the bag-of-words plus geometrical checking approach, but presents several novelties. We propose using a slightly modified version of the state-of-the-art binary descriptor BRIEF [8], along with FAST keypoints [9], to obtain speedy features, as explained in Section III. The BRIEF descriptor is a binary vector where each bit is the result of an intensity comparison between a pair of pixels around a keypoint. We modify them to use only local pairs of pixels. Although BRIEF hardly tolarates scale or rotation changes, these descriptors have shown a very good compromise between distinctiveness and computation time, as well as its feasibility for appearance recognition, according to our experiments. The modified version we use yields even more distinctiveness for the same computation time. Furthermore, we introduce a bag of words that discretises a binary space, and augment it with a direct index, in addition to the usual inverse index, as explained in Section IV. To the best of our knowledge, this is the first time a binary vocabulary is used for loop detection. These three elements compose our image database. The inverse index is used for fast retrieval of images potentially similar to a given one. We show a novel use of the direct index to efficiently obtain correspondences between images, speeding up the geometrical checking during the image matching. Similarly to our previous work, [4], [5], to decide that a loop has been closed, we verify the temporal

consistency of the image matches obtained, as explained in Section V. One of the additions in this paper is a technique to prevent images collected in the same place from competing among them when the database is queried. We achieve this by grouping together those images that depict the same place during the matching. We name these groups *islands*.

We present the results achieved by our technique after evaluating it in three public real datasets (from the European Rawseeds project [10]) with 0.7 and 1.7 Km long trajectories. We demonstrate that we can run the whole loop detection procedure, including the feature computation, in around 16 ms in a sequence with more than 2700 images (6 ms on average) (see Fig. 1), or in 49 ms in 19000 images (16 ms on average), outperforming other similar techniques by more than one order of magnitude. See Section VI for a detailed explanation of the conducted experiments and the results.

II. Related work

Using visual appearance as a means to detect loop closures is obtaining great attention in the robotics community since very good results have been achieved. An example of this is FAB-MAP [1], that detects loops with an omnidirectional camera obtaining 100% precision with a recall of 48.4% and 3.1% in 70 Km and 1000 Km trajectories, respectively. FAB-MAP has become the gold standard regarding loop detection, but its robustness decreases when the images depict very similar structures for a long time [4]. In the work of Angeli et al. [3], two visual vocabularies (for appearance and colour) are created online in an incremental fashion. The two bag-of-words representations are used together as input of a Bayesian filter that estimates the matching probability between two scenes, taking into account the matching probability of previous cases. In contrast to these probabilistic approaches, we rely on a temporal consistency checking to consider previous matches and enhance the reliability of the detections. This technique has proven successful in our previous work [4], [5]. Our work also differs from the ones above in that we use a bag of binary words for the first time, as well as propose a technique to prevent images close in time from competing between them during the matching, so that we can work at a higher frequency. To verify loop closing candidates, a geometrical checking is usually performed. As in [3], we apply an epipolar constraint to the best matching candidate, but we take advantage of a direct index to calculate correspondence points faster. Konolige et al. [2] use visual odometry with a stereo camera to create in real time a view map of the environment, detecting loop closures with a bag-of-words approach as well. Their geometrical checking consists in computing a spatial transformation between the matching images. However, they do not use any filter to consider consistency with previous matches, what forces them to apply the geometrical checking to several loop closing candidates.

In [1], [3], [4], [5] the features used are SIFT [11] or SURF [12]. These are the most popular ones in this context because they are invariant to lighting, scale and rotation changes and show a good behaviour in view of slight perspective changes. However, these features usually require between 100 and 700 ms to be computed, as reported by the above publications. Apart from GPU implementations [13], there are other similar features that try to reduce this computation time by, for example, approximating the SIFT descriptor [14] or reducing the dimensionality with PCA (PCA-SIFT [15], GLOH [16]). Regarding this aspect, [2] offers a qualitative change, since its authors utilise compact randomised tree signatures [17], [18]. This approach calculates the similarity between an image patch and other patches previously trained in an offline stage. The descriptor vector of the patch is computed by concatenating these similarity values, and its dimensionality is finally reduced with random ortho-projections. This yields a very fast to compute descriptor, suitable for real-time applications. Our work bears a resemblance with [2] in that we also reduce the execution time by using fast descriptors. We use BRIEF descriptors [8] which are binary and require very little time to be computed. An advantage of the binary features is that their information is very compacted, so that they occupy less memory and are faster to compare. This also permits to create a bag of words in the binary space, allowing a much faster image conversion into bag-of-words vectors.

III. Binary features

Extracting local features (keypoints and their descriptor vectors) is usually very expensive in terms of computation time when comparing images. This is often the bottleneck when this kind of techniques are applied to real-time scenarios. To overcome this problem, in this work we use FAST keypoints [9] and the state-of-the-art BRIEF descriptors [8]. FAST keypoints are corner-like points detected by comparing the grey intensity of some pixels in a Bresenham circle of radius 3. Since only a few of pixels are checked, these points are very fast to obtain, proving successful for real-time applications.

For each FAST keypoint, we draw a square patch around them and compute a BRIEF descriptor. The BRIEF descriptor of an image patch is a binary vector where each bit is the result of an intensity test between two of the pixels of the patch. The patches are previously smoothed with a Gaussian kernel to reduce noise. Given beforehand the size of the patch, S_b, the pairs of pixels to test are randomly selected in an offline stage. In addition to S_b, we must set the parameter L_b: the number of tests to perform (i.e. the length of the descriptor). Given a point $\mathbf{p}$ in an image, its BRIEF descriptor vector $B(\mathbf{p})$ is:

$$B_i(\mathbf{p}) = \begin{cases} 1 & \text{if } \mathbf{p}+\mathbf{x_i} < \mathbf{p}+\mathbf{y_i} \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in [1..L_b] \qquad (1)$$

where $B_i(\mathbf{p})$ is the i-th bit of the vector, and $\mathbf{x_i}$ and $\mathbf{y_i}$ are the offset of the test points (randomly selected in advance), whose value must lie in $\left[-\frac{S_b}{2} \dots \frac{S_b}{2}\right] \times \left[-\frac{S_b}{2} \dots \frac{S_b}{2}\right]$. Note that this descriptor does not need training, just an offline stage to select random points that hardly takes time. The original BRIEF descriptor proposed by Calonder et al. [8] selects the test points $\mathbf{x}$ and $\mathbf{y}$ according to a normal

distribution $\mathcal{N}(0, \frac{1}{25}S_b^2)$. However, we found that using close test pairs yielded better results (Section VI-A). We select these pairs by sampling the distributions $\mathbf{x} = \mathcal{N}(0, \frac{1}{25}S_b^2)$ and $\mathbf{y} = \mathcal{N}(\mathbf{x}, \frac{4}{625}S_b^2)$. Note that this approach was also reported by [8], but not used in their final experiments. For the descriptor and patch sizes, we chose $L_b = 256$ and $S_b = 48$, because it resulted in a good compromise between distinctiveness and computation time.

The main advantage of BRIEF descriptors is that they are very fast to compute (Calonder et al. [8] reported $17.3\mu s$ per keypoint when $L_b = 256$ bits) and to compare. Since one of these descriptors is just a vector of bits, measuring the distance between two vectors can be done by counting the amount of different bits between them (Hamming distance), that is implemented with a *xor* operation. This is more suitable in this case than calculating the Euclidean distance, as usually done with SIFT or SURF descriptors.

IV. Image Database

In order to detect revisited places we use an image database composed of a hierarchical bag of words [6], [7] and direct and inverse indexes, as shown in Fig. 2.

The bag of words is a technique that allows to convert with a visual vocabulary a set of local features coming from an image into a sparse numerical vector, allowing to manage big sets of images. The visual vocabulary is created offline by discretising the descriptor space into W visual words. Unlike with other features like SIFT or SURF, we discretise a binary descriptor space, creating a more compact vocabulary. In the case of the hierarchical bag of words, the vocabulary is structured as a tree. To build it, we extract a rich set of features from some training images, independently of those processed online later. The descriptors extracted are first discretised into k_w binary clusters by using the k-means++ algorithm [19]. Since k-means++ requires to compute the centroid of several descriptor vectors, we round the value of the centroids to obtain binary clusters. These clusters form the first level of nodes in the vocabulary tree. Subsequent levels are created by repeating this operation with the descriptors associated to each node, up to L_w levels. We finally obtain a tree with W leaves, which are the words of the vocabulary. Each word is given a weight according to its relevance in the training corpus, decreasing the weight of those words which are very frequent and, thus, less discriminative. During the training, we weight each word w_i with its inverse document frequency (*idf*):

$$\text{idf}(i) = \log \frac{N}{n_i} \tag{2}$$

where N is the number of training images, and n_i, the number of occurrences of word w_i in these images.

To convert an image I_t, taken at time t, into a bag-of-words vector $v_t \in \mathbb{R}^W$, the binary descriptors of its features traverse the tree from the root to the leaves, by selecting at each level the intermediate nodes that minimise the Hamming distance. This allows us to calculate the term

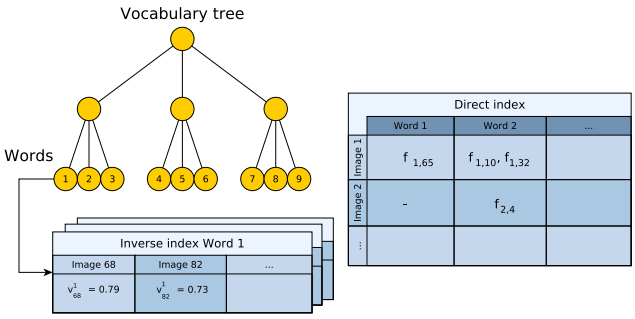


Fig. 2. Bag of words and direct and inverse indexes that compose the image database.

frequency (*tf*) of each word in this image:

$$\text{tf}(i, I_t) = \frac{n_{iI_t}}{n_{I_t}} \tag{3}$$

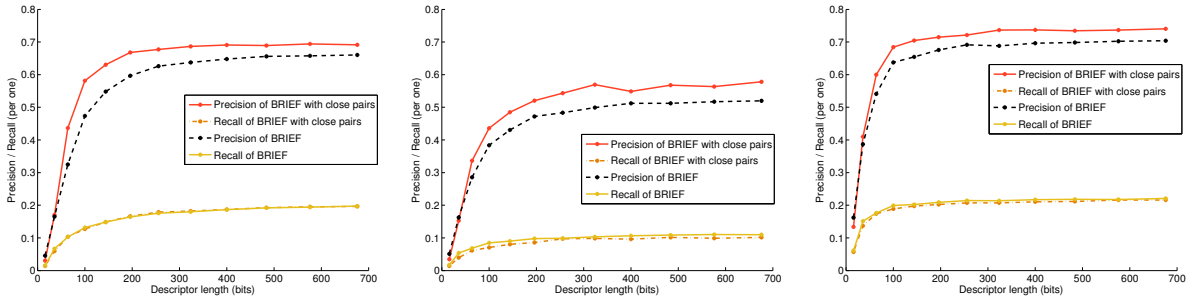
where n_{iI_t} stands for the number of occurrences of word w_i in image I_t, and n_{I_t}, for the number of words in I_t. The i-th entry of v_t is finally given the value $v_t^i = \text{tf}(i, I_t) \times \text{idf}(i)$, obtaining the *tf-idf* weight as proposed in [6]. To measure the similarity between two bag-of-words vectors v_1 and v_2, we calculate a L_1-score $s(v_1, v_2)$, whose value lies in $[0..1]$:

$$s(v_1, v_2) = 1 - \frac{1}{2}\left|\frac{v_1}{|v_1|} - \frac{v_2}{|v_2|}\right| \tag{4}$$

When the database is queried, all the matches are ranked by their scores.

Along with the bag of words, an inverse index is maintained. This is a structure that stores for each word w_i in the vocabulary a list of images I_t where it is present. This is very useful when querying the database, since it allows to perform comparisons only against those images that have some word in common with the query image. We augment the inverse index to store pairs $< I_t,\ v_t^i >$ to quickly access the weight of the word in the image. The inverse index is updated when a new image I_t is added to the database, and accessed when the database is searched for some image.

These two structures (the bag of words and the inverse index) are often the only used in the bag-of-words approach for searching images. However, as a novelty in this general approach, we also make use of a direct index to store for each image I_t the list of words w_i it contains, as well as what local features f_{tj} are associated to each word. We take advantage of the bag-of-words tree to use it as a means to approximate nearest neighbours in the BRIEF descriptor space. So, to compute correspondences between a query image and any image in the database, only those features that belong to the same word need to be compared. This prevents us from matching all the features of two images, which has complexity $\Theta(n^2)$ in the number of features. The direct index is updated when a new image is added to the database, and accessed when a candidate matching is obtained and geometrical checking is necessary.



(a) Scene 1 (b) Scene 2 (c) Scene 3

Fig. 3. Precision and recall in 3 test scenes of BRIEF and BRIEF with close pairs for several descriptor length L_b values, with patch size $S_b = 48$.

V. Loop detection algorithm

To detect loop closures, we use a method based on our previous work [4], [5] that follows these steps:

1) We search the database for the current image to retrieve those scenes whose matching score reaches a threshold.
2) Matches against images gathered close in time are grouped together as a single match.
3) The match with the highest score is checked for temporal consistency with previous scenes to obtain loop closing candidates.
4) Finally, if the best candidate passes a geometrical verification, the loop closure is accepted.

We use the image database to store and to retrieve scenes similar to any given one. When the last image I_t is acquired, it is converted into the bag-of-words vector v_t. The database is searched for v_t, resulting in a list of matches $< v_t,\ v_{t_1} >$, $< v_t,\ v_{t_2} >$, ..., associated to their scores $s(v_t, v_{t_j})$. The range these scores varies depends on the distribution of words in v_t. We then normalise these scores with the best score we expect to obtain in this sequence for the vector v_t, obtaining a new score η:

$$\eta(v_t, v_{t_j}) = \frac{s(v_t, v_{t_j})}{s(v_t, v_{t-\Delta t})} \tag{5}$$

Here, we approximate the expected score of v_t with $s(v_t, v_{t-\Delta t})$, where $v_{t-\Delta t}$, is the bag-of-words vector of the previous image. Those cases where $s(v_t, v_{t-\Delta t})$ is small (e.g. when the robot is turning) can erroneously cause high scores, so that we skip the images that do not reach a minimum $s(v_t, v_{t-\Delta t})$ or a required number of features. We then reject those matches whose $\eta(v_t, v_{t_j})$ does not achieve a minimum value α.

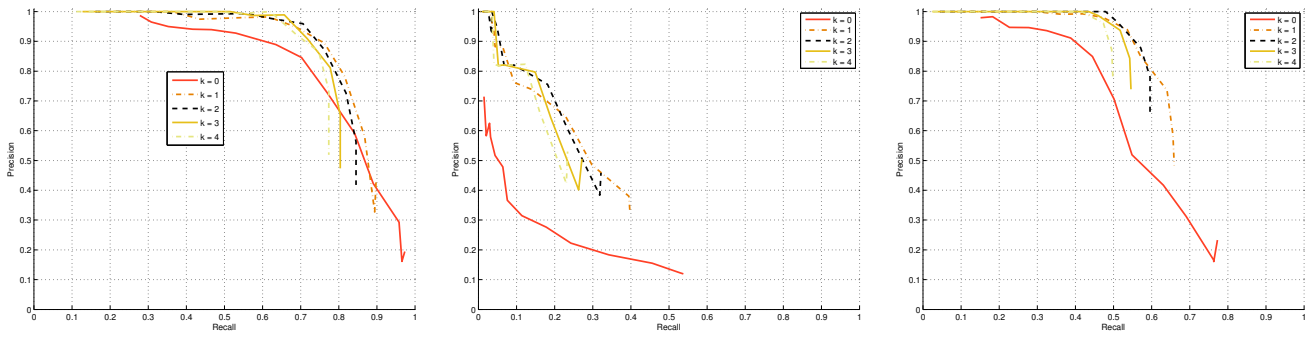
As a novelty, to prevent images that are close in time to compete among them when the database is queried, we group them into *islands* and treat them as only one match. We use the notation T_i to represent the interval composed of time stamps $t_{n_i}, \ldots, t_{m_i}$, and V_{T_i} for an island that groups together the matches with entries $v_{t_{n_i}}, \ldots, v_{t_{m_i}}$. Therefore, several matches $< v_t,\ v_{t_{n_i}} >, \ldots, < v_t,\ v_{t_{m_i}} >$ are converted into a single match $< v_t,\ V_{T_i} >$ if the gaps between consecutive time stamps in $t_{n_i}, \ldots, t_{m_i}$ are small. The islands are also ranked according to a score H:

$$H(v_t,\ V_{T_i}) = \sum_{j=n_i}^{m_i} \eta(v_t,\ v_{t_j}) \tag{6}$$

The island with the highest score H is selected as the matching group. Besides avoiding clashes between consecutive images, the islands can help to establish correct matches. If I_t and $I_{t'}$ represent a real loop closure, I_t is very likely to be similar also to $I_{t'\pm\Delta t}$, $I_{t'\pm 2\Delta t}$, ..., producing long islands. Since we define H as the sum of scores η, the H score favours matches with long islands as well.

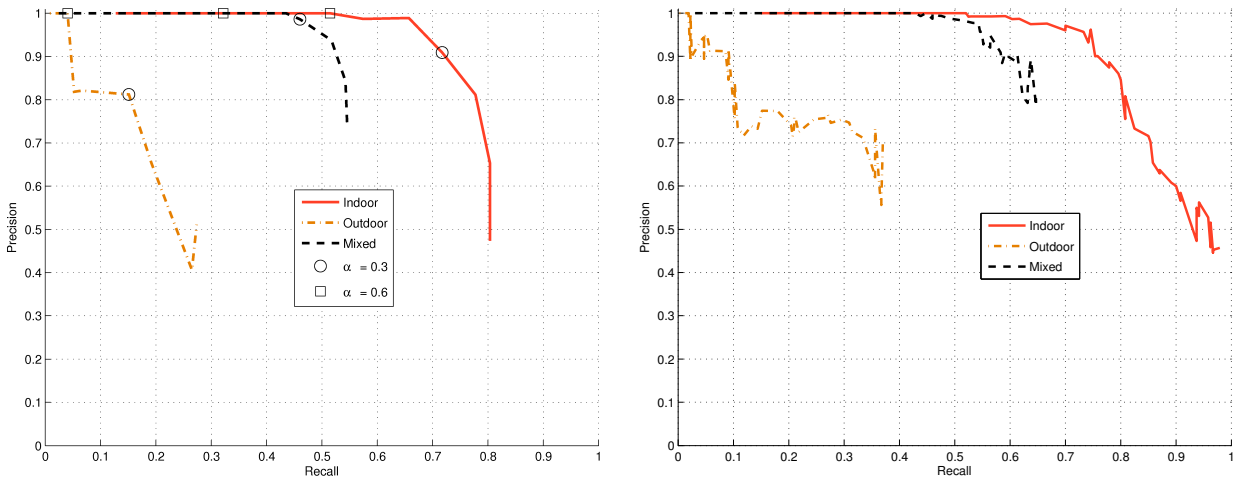
After obtaining the matching island $V_{T'}$, we apply a temporal constraint before accepting the match as valid. In this paper we extend the temporal constraint applied in [4], [5] to support islands. The match $< v_t,\ V_{T'} >$ must be consistent with k previous matches $< v_{t-\Delta t},\ V_{T_1} >$, ..., $< v_{t-k\Delta t},\ V_{T_k} >$, such that the intervals T_i and T_{i+1} are close to overlap. If the island passes the temporal constraint, we keep only the match $< v_t,\ v_{t'} >$, for the $t' \in T'$ that maximises the score η, and consider it the loop closure candidate.

We apply a geometrical checking between the candidate matching scenes. This checking consists in finding with RANSAC [20] a fundamental matrix between I_t and $I_{t'}$ supported by at least 12 correspondences [21]. To compute these correspondences, we must compare the local features of the query image with those of the matched one. There are several approaches to perform this comparison. The easiest and slowest one is the linear search, that consists in measuring the distance of each feature of I_t to the features of $I_{t'}$ in the descriptor space, to take later the pairs with smallest distance. This is a $\Theta(n^2)$ operation in the number of features. A second technique consists in calculating approximate nearest neighbours by arranging the descriptor vectors in k-means or k-d trees [22]. Following the latter idea, we take advantage of our bag-of-words vocabulary and reuse it to approximate nearest neighbours. For this reason, when adding an image to the database, we store the list of pairs of words and features in the direct index. Therefore, to obtain correspondences between I_t and $I_{t'}$, we look up $I_{t'}$ in the direct index and perform the linear search only with those features that are associated to the same word in the vocabulary. We only require the fundamental matrix



(a) Indoor dataset (b) Outdoor dataset (c) Mixed dataset

Fig. 4. Precision-recall curves of the method, with no geometrical checking, for several values of α and k consistent temporal matches



(a) BRIEF (b) SURF

Fig. 5. Precision-recall curves for several values of α and without geometrical checking. On the left, our current method using BRIEF, and on the right, our previous algorithm using SURF [5] on the same datasets. BRIEF features attain similar results than SURF, but require much less computational resources.

for verification, but note that after calculating it, we could provide the data association between the scenes matched to any SLAM algorithm that would run beneath, with no extra cost.

VI. Experimental evaluation

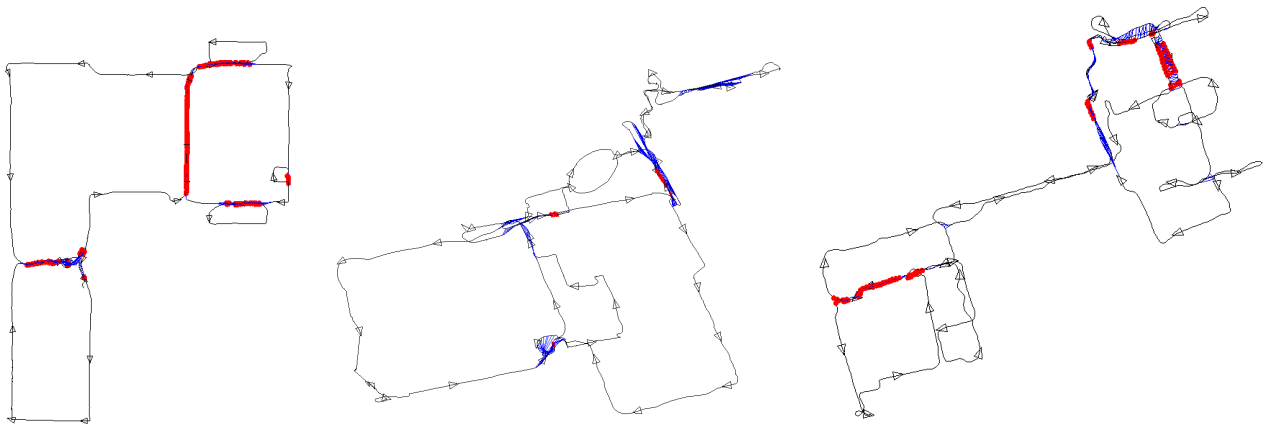
A. Selecting BRIEF parameters

We ran several tests to compare the performance of BRIEF by varying the descriptor length L_b, the patch size S_b and the method to select the test pairs $\mathbf{x}$ and $\mathbf{y}$. We tried two methods to choose the test pairs. The first one consisted in choosing both $\mathbf{x}$ and $\mathbf{y}$ from a normal distribution $\mathcal{N}(0, \frac{1}{25}S_b^2)$, and the second one, in choosing close pairs such that $\mathbf{x} = \mathcal{N}(0, \frac{1}{25}S_b^2)$, $\mathbf{y} = \mathcal{N}(\mathbf{x}, \frac{4}{625}S_b^2)$.

To do this test, we took several images (between 14 and 23) of three different scenes. By means of bundle adjustment [23], we reconstructed the 3D of the scenes and obtained the groundtruth correspondences of every pair of images. We computed both versions of the BRIEF descriptor for patches of size $S_b = 24$, 48, 64 and 80 pixels around a dense set of FAST keypoints in each image, and matched them. A match was set if the distance between two descriptors was minimum, and the ratio between this and the distance to the second closest descriptor was lower than a threshold. Due to the random nature of the descriptor, we repeated this for 5 different test pair patterns. In Fig. 3 we show the average precision and recall against the length of the descriptor when $S_b = 48$ of BRIEF and BRIEF with close pairs. The other patch sizes showed the same relation between both techniques, but $S_b = 48$ exhibited better results. We can see that the precision of BRIEF with close pairs is always higher than that of BRIEF with more general pairs for the same level of recall. This suggests the locality of the pairs provides more distinctiveness to the descriptor. We also see that the precision and recall improve when the number of pairs increases, up to some length. We finally chose BRIEF with close pairs, descriptor length $L_b = 256$ and patch size $S_b = 48$ pixels, for the rest of the experiments.

B. Loop closure detector

We evaluated our loop closing technique in three public real datasets (from the European Rawseeds project [10]: *Bicocca 2009-02-25b*, *Bovisa 2008-10-04* and *Bovisa 2008-*



(a) Indoor (b) Outdoor (c) Mixed

Fig. 6. Loops detected in the three datasets. Black lines and triangles denote the trajectory of the robot; deep blue lines, actual loops, and light red lines, loops detected. There are no false positives.

10-06). These datasets present static indoor, static outdoor and dynamic mixed indoor and outdoor environments, and were obtained in two university campuses in different dates. We use the data of one of the cameras, composed of between 26K and 35K 640×480 b/w images, collected at 15 Hz from 0.7–1.7 Km trajectories.

We used a vocabulary tree with $k_w = 10$ branches and $L_w = 6$ depth levels, which yielded one million words. This was trained with around 10K images and 9M features acquired from a fourth Rawseeds dataset. During the experiments, the features extracted from each image were limited to 300, with a threshold in the corner response function of FAST of 10 units. When querying the database, only the $50f$ returned results were kept, ignoring those which were close to the current time stamp, being f the frequency at which the images were processed.

We began testing our system with $f = 1$ Hz and without any geometrical constraint. We show in Fig. 4 the precision-recall curves obtained in each dataset by varying the α value. We also show the effect of increasing the number k of temporal consistent matches required to accept a match. The curve with $k = 0$ was obtained by disabling the temporal consistency. We can see there is a big improvement between $k = 0$ and $k > 0$. This shows the temporal consistency is a valuable mechanism to avoid mismatches. As one could expect, as k increases, a higher recall is attained with 100% precision. This behaviour does not hold for very high values of k, since only very long closures would be found. We chose $k = 3$ since it showed a good precision-recall balance in the three datasets. Note that for $f = 1$ Hz, this means a loop must last 3 seconds at least.

We show in Fig. 5(a) the precision-recall with $k = 3$ for several values of α. We see the system performed very well, even without geometrical verification. It obtained high recall, except for the outdoor dataset, without false positives. These results bear resemblance to those presented in our previous work [5] (shown in Fig. 5(b)), where we used a similar system with SURF features. Apparently, the curve of the outdoor dataset behaves worse with BRIEF than with SURF. However with 100% precision, the desired working point, the recall is slightly higher in the BRIEF case (4.1%). In both cases, the recall in this dataset is lower than that shown in the other two datasets. As we noticed in our previous work [4], [5], this outdoor dataset is particularly challenging because the depth of the scenes produced very similar-looking images. In [5], we showed that with our previous technique, in the best case and without geometrical constraint, we achieved 1.9% recall with no false positives. In [4], we also showed that other techniques as FAB-MAP [24] did not reach 100% precision on this dataset. This proves that BRIEF descriptors, despite of lacking scale and rotation invariance, are much faster and as reliable as SURF features for loop detection problems with in-plane camera motion.

TABLE I

PRECISION AND RECALL OF OUR SYSTEM

Dataset	Length (m)	Precision (%)	Recall (%)	# Images
Indoor	760	100	57.86	2723
Outdoor	1365	100	5.83	2345
Mixed	1750	100	28.08	2247

According to Fig. 5, we could select a restrictive value of α, e.g. 0.6 as in [5], to obtain 100% precision, but this may depend on the dataset. Another option is to set a lower value and to verify the matches with the geometrical constraint, to get maximum precision improving the recall. Following the latter idea, we set $\alpha = 0.3$ and added the fundamental matrix restriction. Fig. 6 shows the loops finally detected by our whole system. Black lines and triangles denote the trajectory of the robot; deep blue lines, actual loops, and light red lines, loops detected. The figures of these results are detailed in Table I, showing that we attain 100% precision in the indoor, outdoor and mixed datasets, with recall 57.86%, 5.83% and 28.08%, respectively. The number of checked images is higher in the indoor dataset because many of them depicted just blank walls without enough features.

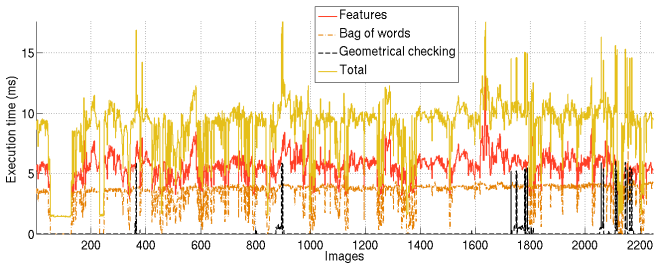


Fig. 7. Execution time of the system in the Mixed dataset for $f = 1$ Hz

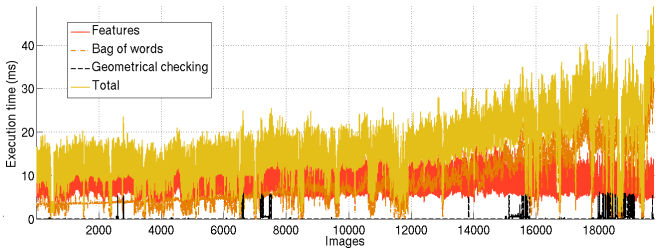


Fig. 8. Execution time of the system in the Mixed dataset for $f = 10$ Hz

TABLE II
EXECUTION TIME FOR 2723 IMAGES (MS)

		Mean	Std	Min	Max
Features	FAST	2.39	1.07	1.37	7.84
	Smoothing	1.27	0.05	1.21	1.43
	BRIEF	1.44	0.37	0.11	1.74
Bag of words	Conversion	3.00	0.78	0.21	3.66
	Query	0.17	0.13	0.00	0.62
	Islands	0.06	0.02	0.01	0.11
	Insertion	0.02	0.04	0.00	0.13
Verification	Correspondences and RANSAC	0.82	1.29	0.25	5.79
Whole system		6.15	4.03	1.37	16.34

TABLE III
EXECUTION TIME OF GEOMETRICAL CHECKING (MS)

	Mean	Std	Min	Max
With direct index	0.82	1.29	0.25	5.79
Without direct index	17.82	9.67	0.14	57.07

TABLE IV
EXECUTION TIME FOR 19344 IMAGES (MS)

		Mean	Std	Min	Max
Features	FAST	4.52	2.16	1.39	16.66
	Smoothing	1.65	0.75	1.21	6.90
	BRIEF	1.54	0.54	0.10	4.77
Bag of words	Conversion	2.98	0.78	0.21	7.00
	Query	4.95	5.44	0.00	36.38
	Islands	0.07	0.02	0.01	0.18
	Insertion	0.02	0.01	0.00	0.15
Verification	Correspondences and RANSAC	1.03	1.59	0.02	6.12
Whole system		15.69	6.88	1.39	48.91

C. Performance

In Fig. 7 we show the execution time consumed per image in the mixed dataset. This was measured in a Intel Core @ 2.67GHz machine. We also show in Table II the required time of each stage for 2723 images. The *features* time involve computing FAST keypoints and removing the less persistent ones when there are too many, as well as smoothing the image with a Gaussian kernel and computing BRIEF descriptors. The *bag-of-words* time is split into four steps: the conversion of image features into a bag-of-words vector, the database query to retrieve similar images, the creation and matching of islands, and the insertion of the current image into the database (this also involves updating the direct and inverse indexes). The *verification* time include both computing correspondences between the matching scenes, by means of the direct index, and the RANSAC loop to calculate a fundamental matrix.

We see that all the steps are very fast, including extracting the features and the maintenance of the direct and inverse indexes. The required time of managing the islands is negligible. This allows to obtain a system that runs in 6 ms per image, with a peak of less than 17 ms, far below the 33 ms usually available in a real-time video acquiring images at 30 Hz. The feature extraction step presents the highest maximum; even so, we have achieved a reduction of more than one order of magnitude with respect to other features, such as SIFT or SURF, removing the bottleneck of these loop closure detection algorithms. The conversion of image features into bag-of-words vectors is the second stage that requires more time. Its execution time depends on the number of features and the size of the vocabulary. We could reduce it by using a smaller vocabulary, since we are using a relatively big one (1M words, instead of 10–60K [1], [4]). However, we found that a big vocabulary produces more sparse inverse indexes associated to words. Therefore, when querying, fewer database entries must be traversed to obtain the results. This reduces the execution time strikingly when querying, trading off, by far, the time required when converting a new image. We conclude that big vocabularies can improve the computation time when using big image collections.

We can see in Table II that the execution time of the geometrical verification is very low. This is because we use a direct file to retrieve the features of an image associated to each word. We only compute correspondences between two features associated to the same word. In Table III we compare the execution time of computing correspondences and calculating a fundamental matrix with and without using the direct index. By using the direct index, we decrease the execution time by around 90%.

We conducted the same experiments with a frequency $f = 10$ Hz to test the system in a longer sequence. We show in Table IV the execution time of the system for 19344 images. Fig. 8 shows a graph with the results in the mixed dataset. The execution time is higher than that for the previous case because the number of entries in the database is greater now. This causes queries to check more possible matches, demanding more computation cycles. Even so, the required time for querying is still very low, between 5 and 37 ms, suggesting this stage scales well with tens of thousands

images. The rest of the stages do not present big differences with the results of Table II. Our complete algorithm requires less than 49 ms for closing a loop against a database with more than 19K images.

VII. Conclusions and Future work

We have presented a real-time technique to detect loops in monocular sequences, introducing several novelties to the bag-of-words and geometrical checking approach. We have demonstrated we can speed up the execution of these algorithms by more than one order of magnitude by using BRIEF descriptors, discretised by a hierarchical bag of binary words. We have shown that BRIEF descriptors with close pairs are as reliable for loop detection as SURF features with in-plane camera motion. To prevent images taken in close positions from clashing and, thus, to improve the matching, we group close matches into *islands*. We have shown a technique based on our previous work [4], [5] to impose a temporal constraint on the islands. We have also introduced the novel use of a direct index to optimise the geometrical verification, obtaining a speed up of around 90% when computing correspondence points. However, the direct index may intrinsically forbid some correspondences between features that do not belong to the same visual word. We could ease this condition by computing correspondences between features whose word had some parent node in common. The number of levels to go up in the vocabulary tree to find those common nodes would trade off speed and chances to find more correct correspondences.

The reliability and efficiency of our proposal have been shown on three real and public datasets from the European Rawseeds project. We have concluded that a big vocabulary can lower the execution time when using big databases. We have shown we can detect loops against databases with 2.7K images in 16 ms (6 ms on average), and against 19K images in 49 ms (16 ms on average). This shows an improvement of more than one order of magnitude from the more than 100–700 ms required by algorithms based on SIFT or SURF [1], [3], [4], [5]. Our algorithm outperforms as well the execution time of the loop detector of [2], that uses compact randomised tree signatures [17], [18], proven very efficient. According to the figure 6 of [2], their method requires around 200 ms to perform the complete loop detection against a database with 2700 images.

Our system has been effective detecting loops in scenarios with perceptual aliasing. In the future, we will research the automatic learning of the system parameters to test our algorithm in a wide variety of challenging environments, as urban areas with highly dynamic objects, as cars or people.

References

[1] R. Paul and P. Newman, "FAB-MAP 3D: Topological mapping with spatial and visual appearance," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, pp. 2649–2656.

[2] K. Konolige, J. Bowman, J. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua, "View-based maps," *The International Journal of Robotics Research*, vol. 29, no. 8, p. 941, 2010.

[3] A. Angeli, D. Filliat, S. Doncieux, and J. Meyer, "Fast and incremental method for loop-closure detection using bags of visual words," *Robotics, IEEE Transactions on*, vol. 24, no. 5, pp. 1027–1037, 2008.

[4] P. Piniés, L. M. Paz, D. Gálvez-López, and J. Tardós, "CI-Graph SLAM for 3D Reconstruction of Large and Complex Environments using a Multicamera System," *International Journal of Field Robotics*, vol. 27, no. 5, pp. 561–586, September/October 2010.

[5] C. Cadena, D. Gálvez-López, F. Ramos, J. Tardós, and J. Neira, "Robust place recognition with stereo cameras." IEEE International Conference on Intelligent Robots and Systems, October 2010, pp. 5182–5189.

[6] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proceedings of the International Conference on Computer Vision*, vol. 2, Oct. 2003, pp. 1470–1477. [Online]. Available: http://www.robots.ox.ac.uk/~vgg

[7] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2, 2006, pp. 2161–2168.

[8] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," in *European Conference on Computer Vision*, September 2010.

[9] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," *Computer Vision–ECCV 2006*, pp. 430–443, 2006.

[10] RAWSEEDS. (2007-2009) Robotics Advancement through Web-publishing of Sensorial and Elaborated Extensive Data Sets (Project FP6-IST-045144). http://www.rawseeds.org/rs/datasets.

[11] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[12] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," *Computer Vision–ECCV 2006*, pp. 404–417, 2006.

[13] S. Heymann, K. Maller, A. Smolic, B. Froehlich, and T. Wiegand, "SIFT implementation and optimization for general-purpose GPU," in *Proceedings of the International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, 2007.

[14] M. Grabner, H. Grabner, and H. Bischof, "Fast approximated SIFT," *Computer Vision–ACCV 2006*, pp. 918–927, 2006.

[15] Y. Ke and R. Sukthankar, "PCA-SIFT: a more distinctive representation for local image descriptors," vol. 2, pp. II–506 – II–513 Vol.2, june-2 july 2004.

[16] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1615–1630, 2005.

[17] M. Calonder, V. Lepetit, P. Fua, K. Konolige, J. Bowman, and P. Mihelich, "Compact signatures for high-speed interest point description and matching," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2010, pp. 357–364.

[18] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1465–1479, 2006.

[19] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

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

[21] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[22] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *International Conference on Computer Vision Theory and Application VISSAPP'09)*. INSTICC Press, 2009, pp. 331–340.

[23] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle adjustment – A modern synthesis," in *Vision Algorithms: Theory and Practice*, ser. LNCS. Springer Verlag, 2000, pp. 298–375.

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