Superpixel Description and Indexing for Visual Loop Closure Detection

Rihem El Euch*, Emilio Garcia-Fidalgo†, Alberto Ortiz†, Ferdaous Chaabane*, Adel Ghazel*

* *Higher School of Communication of Tunisia (Sup'Com)*, Tunis, Tunisia
{rihem.eleuch, ferdaous.chaabene, adel.ghazel}@supcom.tn

†*Department of Mathematics and Computer Science*, *University of the Balearic Islands*, Palma, Spain
{emilio.garcia, alberto.ortiz}@uib.es

***Abstract*—Recognizing whether the current place has been visited before or not is an important task in robotic navigation, since it helps to reduce the inconsistencies produced by the inherent noise of navigation sensors. When a camera is used as input for navigation, this process is known as visual loop closure detection. Under this context, in this work we propose a loop closure detection method based on superpixels and a Bag of Words scheme. A novel image description method for superpixels is proposed. Our approach also makes use of the concept of dynamic islands, which allows us to group images close in time and to avoid images taken from the same place to compete among them as loop closure candidates. The proposed method is validated using several outdoor public image sequences and compared to other state-of-the-art solutions.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) techniques have become an important task for autonomous robots. Normally, these approaches rely on loop Closure Detection (LCD) methods, which detect whether the robot is in a previously visited place. This information is crucial for SLAM, since it can be used to reduce the inaccuracies produced by using raw sensor data, generating more accurate maps. In the last decades, a high number of vision-based solutions have emerged, motivated by the low cost of cameras and the rich source of information that images provide. These solutions are usually known as *appearance-based* loop closure detection methods [1].

The Bag of Words (BoW) model, used in combination with an inverted file, has been shown as an effective method for indexing previously seen images. Due to this reason, it has been extensively used for visual loop closure detection [2]–[5]. Depending on when the dictionary is built, BoW-based methods can be mainly classified into offline and online solutions. Regarding offline solutions, where the dictionary is built during a training stage, one of the key works in this field is FAB-MAP [2], where the authors introduced a probabilistic framework using a visual vocabulary trained on SURF descriptors together with an inverted file to detect visited places. A binary vocabulary using BRIEF descriptors is trained in [3]. To detect loop closures, the authors introduced the concept of fixed size islands to group images close in time and thus a query image is matched to the island with the highest score. More recently, Bampis et al. [6] proposed a method for detecting loop closures by matching sequences of images instead of single frames. The main difference between their work and seqSLAM [7], which is also based on sequence matching, is in describing the sequence of images by combining the visual words found in the images into a single vector. In seqSLAM, sequence matching is based on accumulating matching scores between visual words vectors. Conversely, online solutions build the dictionary as the vehicle navigates, avoiding the training stage. In this regard, Angeli et al. [5] proposed an incremental BoW using SIFT descriptors to detect loop closures within a Bayesian filtering scheme. In [8], the authors introduced an Online Visual Vocabulary (OVV), where an incremental vocabulary is built using SURF features and an agglomerative clustering approach. Recently, IBoW-LCD [4] built the visual vocabulary online using ORB descriptors. The search for similar images is performed using an inverted file and then using the concept of dynamic islands to group images close in time. In this work, we make use of the concept of dynamic islands, but in combination with an offline BoW approach.

Methods described so far are mainly based on local feature descriptors. In this case, a high number of keypoints are typically used per image to describe it accurately. As an alternative, other authors have made use of global descriptors [7], [9], where the whole image is represented by just a single vector. Despite the fact that all these solutions have demonstrated good performance for LCD, one can still consider a third kind of description approach based on superpixels as an intermediate representation and a good compromise between local and global descriptors. A superpixel, which is a group of pixels of an image with similar properties, carry more information than single pixels and line up better with object boundaries than image patches. Some frameworks have already used superpixels for Visual Place Recognition (VPR), which can be seen as a very related field, more focused on severe appearance changes. In [10], an image is segmented into

This work is partially supported by projects ROBINS (EU-H2020, GA 779776), PGC2018-095709-B-C21 (MCIU/AEI/FEDER, UE) and PROCOE/4/2017 (Govern Balear, 50% P.O. FEDER 2014-2020 Illes Balears). This work is also supported by the ERASMUS+ KA107 mobility program.

superpixels using SLIC and Convolutional Neural Networks (CNNs) are next used to extract features. In [11], authors tried to predict the image changes for different seasons. They describe superpixels using color histograms combined with SURF descriptors. Later, a vocabulary is built for each scene condition and, then, an additional dictionary that translates between both vocabularies is trained and used for recognizing places.

Given the benefits that superpixels present, in this work we introduce a novel method for LCD based on superpixels instead of keypoints or global descriptors. A key difference of our work and [10], [11] is that the latter use superpixels for VPR in changing environements. Instead we use superpixels to detect visited places. To describe the image, we also introduce a method for describing superpixels that make use of different global description strategies. To retrieve similar images, an offline BoW model is used together with an inverted file, and, additionally, we use the concept of dynamic islands [4] in order to avoid images taken from the same place to compete among them as loop closure candidates. Our solution is validated against several public datasets and compared against several state-of-the-art solutions.

II. Image Description

A. Superpixel Segmentation

To describe an image, our approach begins by performing a superpixel segmentation of the corresponding frame. As mentioned before, a superpixel is a group of neighbouring pixels with similar color or texture properties. Superpixels usually represent small regions of the image and/or objects. A high number of superpixel algorithms can be found in the literature [12]. In this work, we consider two of them: a graph-based image segmentation method [13] and the Simple Linear Iterative Clustering (SLIC) method [14]. We have empirically found that the zero parameter version of SLIC (SLICO) results into more compact and regular-shaped superpixels. SLIC clusters pixels in the combined five-dimensional color (LAB) and image plane (x, y) space to efficiently generate compact, nearly uniform superpixels, in an efficient and fast way. This algorithm shows good boundary adherence and relatively stable superpixels in several of the datasets used in this work, which is of high importance for loop closure detection. Due to these advantages and its simplicity, we chose SLIC as a superpixel segmentation method.

B. Describing Superpixels

After the segmentation step, we describe each of the resulting superpixels. For this purpose, we consider three main visual properties of the superpixel: colour, texture and structure. Several feature vectors are computed for each of these properties and, finally, combined to create the final descriptor of the superpixel.

As a colour descriptor, we extract RGB histograms to represent the distribution of pixel colours in every superpixel of the image. Each color channel (red, green, blue) is used to create a histogram of 32 bins from pixels corresponding to superpixel k. Therefore, we compute three feature vectors V^R_{Hist}, V^G_{Hist}, V^B_{Hist}, one for each channel.

To represent texture, we use Uniform Local Binary Patterns (ULBP) histograms [15]. We convert the colour image to gray scale and then each pixel is compared in intensity with its circularly symmetric neighbourhood of *p* pixels. In this work, we use $p = 8$ surrounding pixels and a radius $r = 1$. If the intensity of the center pixel is higher than its neighbour's, the corresponding bit of the binary pattern is set to 0. Otherwise it is set to 1. Therefore, the LBP value for pixel c is calculated as:

$$LBP_{p,r}(c) = \sum_{j=0}^{p-1} S(I_j - I_c) \times 2^j, \tag{1}$$

where I_j and I_c are respectively the intensity values of the neighbour and central pixels and $S(x)$ is defined as:

$$S(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Next, a histogram of the LBP values V_{LBP} is computed for superpixel k. We consider the uniform patterns, which contain at most 2 transitions between 0 and 1, e.g. $(10001111)_2$, because they convey important texture properties. This results into a histogram of 10 dimensions, although we discard the last bin, since it accounts for the non-uniform patterns.

The Histogram of Oriented Gradients (HoG) [16] is employed as a structure descriptor for the superpixel. We denote this feature vector as V_{HOG}. Unlike the original work, where a high-dimensional descriptor is computed for recognition tasks, in our solution we extract a 16-bin histogram of gradient orientations for each superpixel.

Finally, each individual feature vector is normalized separately, and, then combined to create the final descriptor for the superpixel k:

$$V(k) = [\overline{V}^R_{Hist}, \overline{V}^G_{Hist}, \overline{V}^B_{Hist}, \overline{V}_{LBP}, \overline{V}_{HOG}]. \tag{3}$$

Given the sizes of the individual feature vectors, the length of the final descriptor is $32 + 32 + 32 + 9 + 16 = 121$. In this work, we calculate the dissimilarity between descriptors using the Euclidean distance.

III. Image Database

A. Training the Visual Vocabulary

In order to retrieve previous images, we use an offline BoW model in combination with an inverted index similarly to [3], although our method is based on superpixel descriptors. In an offline BoW scheme, the visual dictionary is created during a training stage. A set of training images is segmented into superpixels, and, next, a descriptor is computed for each one, as explained in section II-B. The set of superpixel descriptors is clustered through the k-means algorithm. Resulting representative cluster centers are selected as the visual words for the vocabulary. During this training stage, a weight is assigned to each visual word by means of the inverse document frequency (*idf*) term, which determines the importance of the word in the set of training images. Thus we scale up the weight of

the rare visual words and give lower weight to the frequent visual words because they are not discriminative. The *idf* is calculated by (4):

$$\text{idf} = \log\left(\frac{N}{N_w}\right), \tag{4}$$

where N is the total number of images and N_w is the number of images where the visual word w appears [17].

B. Retrieving Similar Images

Given a query frame I_q, we calculate first the superpixel descriptors. Next, similarity score $s(I_q, I_t)$ is initialized to 0 for all possible t previously seen frames. Subsequently, each superpixel descriptor in the query image is efficiently associated to its closest visual word in the vocabulary by means of a set of randomized kd-trees. Next, the inverted file allows us to retrieve a list of similar images where each visual word appears. For each visual word in I_q, we compute the term frequency (*tf*) weight that represents the frequency of a word in the image and we update the score s of the train images that share common words with the query frame as follows:

$$s(I_q, I_t) = s(I_q, I_t) + tf \times idf \tag{5}$$

After getting the list of similar images with their corresponding scores, we store the query image I_q in the inverted file.

IV. Loop Closure Detection

The method we use to detect loop closures is based on [4]. Given a query image at time t, we start searching similar images using the inverted file explained in the previous section. In order to avoid matching the query image with the immediate previous frames, we delay the storage of the most recent *N* frames into the inverted file. After getting a list of similar images with their corresponding scores we normalize the scores using min-max [4]: For a query image I_t at time stamp t having j similar images $C_t = \{I_{s_1}, \ldots, I_{s_j}\}$, ordered by their corresponding scores $s(I_t, I_k)$, the normalized score is defined by eq.(6):

$$\tilde{s}(I_t, I_k) = \frac{s(I_t, I_k) - s(I_t, I_{s_1})}{s(I_t, I_{s_j}) - s(I_t, I_{s_1})}, \tag{6}$$

where $s(I_t, I_{s_1})$ and $s(I_t, I_{s_j})$ are, respectively, the minimum and maximum scores. Then we filter the list of normalized scores according to a threshold and get the final list of image matches $\tilde{C}_t \subseteq C_t$.

We next use the concept of the *dynamic islands* to group together images in $\tilde{C}_t$ close in time. Each island is represented by the image with highest score. To find the loop closure candidate, we start by searching for the closest island X_n^m. If the time-stamp of I_t belongs to the time interval of the island, then the image is part of this island X_n^m and the time interval of X_n^m is updated to includeI_t. In case no close island is found, a new island is created. Next, we compute a score for every island which is the sum of scores of the images that belong to the island. We normalize the islands scores G by dividing them by the size of the island:

$$G(X_n^m) = \frac{\sum_{i=m}^{n} \tilde{s}(I_t, I_i)}{m - n + 1}, \tag{7}$$

and finally get a list of sorted island scores *L*. To select the final loop closure candidate, we search for *priority islands*, which are the islands found at time *t* that overlap with the best island selected at time *t-1*. If we find *priority islands* we consider the one with the highest score and select its representative image as loop closure candidate. If no priority islands are found, we choose the island with the highest score in *L*.

The last step comprises an epipolarity geometry check. Similarly to [4], we compute feature matchings between the query image and the final candidate, then we filter these matches using the ratio test and also by means of RANSAC using the epipolarity constraint as the model to fit.

V. Experimental Results

In this section we evaluate the proposed system using public datasets. We use the sequences 01, 02, 03, 04 and 07 of the KITTI dataset for the training process to build two vocabularies of 800000 and 100000 visual words using 5 million descriptors from 7930 images. Next, we use the sequences 00, 05 and 06 of the KITTI for testing.

We use $N = 100$ to avoid the previous *N* frames as loop closure candidates (see section IV). For the epipolarity check, 900 FAST keypoints are detected and described using BRISK descriptors.

For the evaluation, we utilize the available ground truth and use the precision-recall metrics. Precision is defined as the number of correct loop closure detections (true positives) divided by those correct matches plus the incorrect detects (false positives). Recall is defined as the ratio of the correct loop closure detections over the correct detections plus the detect discarded erroneously by the system (false negatives). The goal of our system is to achieve a high recall value at 100 % of precision. The precision-recall curves can be found in Fig. 1 and in Fig. 2 for both vocabularies.

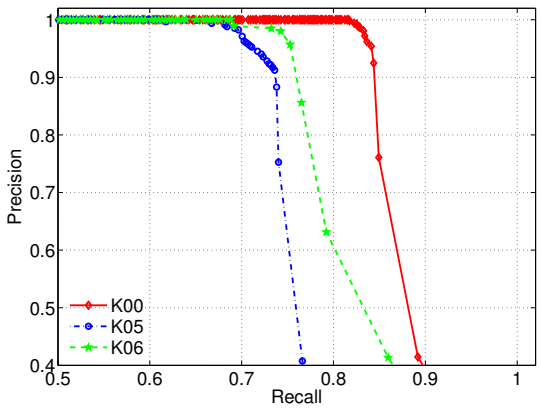

Fig. 1. Precision-recall (P-R) curves for the 100k visual words vocabulary.

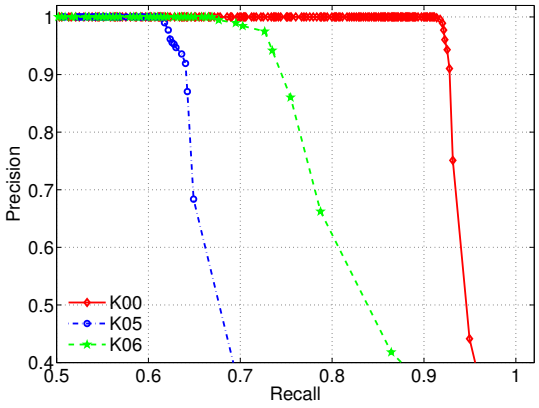

Fig. 2. Precision-recall (P-R) curves for the 800k visual words vocabulary.

We can see that the proposed method achieves high recall rates especially for the KITTI00 sequence for which the recall is above 90(%) at 100(%) of precision using the larger vocabulary. We show the precision-recall values for both vocabularies in table I. We can see that the size of the vocabulary does not have a big impact on the recall values, since we can still detect a large amount of loop closures with the smaller dictionary and requiring less computational time.

TABLE I
COMPARISON OF MAXIMUM RECALL AT 100% PRECISION.

	800K visual words R (%)	**100K visual words R (%)**
K00	91.38	81.62
K05	61.33	61.50
K06	67.65	68.77

Next in table II we compare our method using the 100K visual vocabulary with previous techniques available in the literature. The results [2] and [7] are taken from [4]. The term *n.a.* means that the corresponding metric value is not available. In the case of the sequence 00, our method attains higher recall compared to the other methods, achieving 81.62(%) recall at 100(%) precision. For the sequences K05 and K06, we can see that the incremental approaches [4] and [6] outperform our method, although ours is the second best. Summing up, we observe that by using superpixels we obtain similar performance than keypoints for fewer computational resources.

TABLE II
COMPARISON OF MAXIMUM RECALL AT 100% PRECISION.

	K00	**K05**	**K06**
Cummins [2]	49.21	32.15	55.34
Milford [7]	67.04	41.37	64.68
Bampis [6]	81.54	**84.80**	n.a.
Garcia-Fidalgo [4]	76.50	n.a.	**95.53**
Proposed	**81.62**	61.50	68.77

VI. CONCLUSION AND FUTURE WORK

In this work, we propose the use of superpixels for appearance-based loop closure detection. We make use of a descriptor combining color, structure and texture to describe the superpixels. To detect revisited places we rely on an offline Bag of Words model and an inverted file to retrieve similar images. This framework also uses the concept of islands to group similar images according to their time interval. To validate our approach we used a public dataset, the KITTI dataset, and compared our results with other state-of-the-art solutions.

Using a trained vocabulary takes a long time and limits the visual words to a certain environment. To overcome this limitation, our future work will focus on using an incremental solution. We also plan to consider Convolutional Neural Networks (CNN's) for describing superpixels.

REFERENCES

[1] E. Garcia-Fidalgo and A. Ortiz, "Vision-based topological mapping and localization methods: A survey," *Robotics and Autonomous Systems*, vol. 64, pp. 1–20, 2015.

[2] M. Cummins and P. Newman, "Appearance-only slam at large scale with fab-map 2.0," *The International Journal of Robotics Research*, vol. 30, no. 9, pp. 1100–1123, 2011.

[3] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[4] E. Garcia-Fidalgo and A. Ortiz, "ibow-lcd: An appearance-based loop-closure detection approach using incremental bags of binary words," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3051–3057, 2018.

[5] 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*, pp. 1027–1037, 2008.

[6] L. Bampis, A. Amanatiadis, and A. Gasteratos, "Encoding the description of image sequences: A two-layered pipeline for loop closure detection," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 4530–4536.

[7] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *Proc. IEEE International Conference on Robotics and Automation*, 2012, pp. 1643–1649.

[8] T. Nicosevici and R. Garcia, "Automatic visual bag-of-words for online robot navigation and mapping," *IEEE Transactions on Robotics*, vol. 28, no. 4, pp. 886–898, 2012.

[9] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, J. Yebes, and S. Gamez, "Bidirectional Loop Closure Detection on Panoramas for Visual Navigation," in *Proc. IEEE Intelligent Vehicles Symposium*, 2014, pp. 1378–1383.

[10] Q. Li, K. Li, X. You, S. Bu, and Z. Liu, "Place recognition based on deep feature and adaptive weighting of similarity matrix," *Neurocomputing*, vol. 199, pp. 114–127, 2016.

[11] P. Neubert, N. Sünderhauf, and P. Protzel, "Superpixel-based appearance change prediction for long-term navigation across seasons," *Robotics and Autonomous Systems*, vol. 69, pp. 15–27, 2015.

[12] D. Stutz, A. Hermans, and B. Leibe, "Superpixels: An evaluation of the state-of-the-art," *Computer Vision and Image Understanding*, vol. 166, pp. 1–27, 2018.

[13] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International journal of computer vision*, vol. 59, no. 2, pp. 167–181, 2004.

[14] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels," Tech. Rep., 2010.

[15] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 7, pp. 971–987, 2002.

[16] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE International Conference on Computer Vision & Pattern Recognition*, vol. 1, 2005, pp. 886–893.

[17] J. Sivic and A. Zisserman, "Video google: a text retrieval approach to object matching in videos," in *Proc. IEEE International Conference on Computer Vision*, Oct 2003, pp. 1470–1477 vol.2.