2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Object-based Place Recognition and Loop Closing with Jigsaw Puzzle Image Segmentation Algorithm

Chang Cheng, David L. Page, and Mongi. A. Abidi

Abstract— **In this paper we present a novel place recognition method. Instead of directly using large numbers of SIFT features as visual landmarks, we first use a jigsaw puzzle image segmentation algorithm to segment the input scene image into regions that may correspond to objects or parts of objects. Based on these image regions, we further detect a set of salient objects to represent a place and only those SIFT descriptors that were contained in these salient objects were kept in the database. We also designed a range-tree data structure to organize these salient objects to increase the matching efficiency. Experiments show that place recognition can be achieved accurately and efficiently with these salient objects.**

I. Introduction

Reliable place recognition is an important computer vision task for autonomous navigation of mobile vehicles, particularly, for the simultaneous localization and mapping (SLAM) problem. Although much progress has been made [7],[8],[9], today's SLAM technology still faces some serious problems that prevent it from becoming a robust and practical application system. A major problem SLAM faces is the big loop closure problem [1]: when a vehicle is closing a big loop, such as around a building or hallway corridors, it cannot detect that loop due to the gross error of location estimate and hence is unable to build a consistent map.

The key to solving the loop closure problem is enabling vehicles to quickly detect a loop while revisiting a previously visited area. One possible way to achieve that is to capture some special geometric or visual features from each scene so that the appearance dissimilarity of different scenes can be accurately measured. Most existing SLAM systems use range devices such as a range scanner and laser scanner to measure the geometric primitives (like corners and edges) of a workspace to build a map. These simple geometric features are often not special enough to build a unique "signature" for a place. In contrast, it is generally agreed that vision is one of the richest sources of information and it has the potential to provide enough information to uniquely identify the robot's position [12].

Good visual features should have the following properties: (1) robust to changes in view point. (2) robust to changes in scale and (3) robust to changes in illumination. When mobile vehicles are moving around in a complex environment, they can usually observe many landmarks over time from different angles, distances or illuminations. Some affine covariant visual features such as maximally stable extremal regions (MSER) [3] and the scale invariant feature transform (SIFT) [2] can basically fulfill these requirements. The performance evaluation of different affine covariant features can be found in [5],[6]. Promising results have been obtained by some recent appearance based approaches [1],[10],[11] that are based on the affine covariant features. In [1], the combination of saliency [4] and MSER features are used as visual landmarks. In [10],[11] SIFT features are used as the basic visual features. Typically, the limitation of these methods is that it is difficult to scale them to large environments.

Large number of affine covariant features may be extracted from a typical image. In a large environment, it is often the case that a database that may contain hundreds of

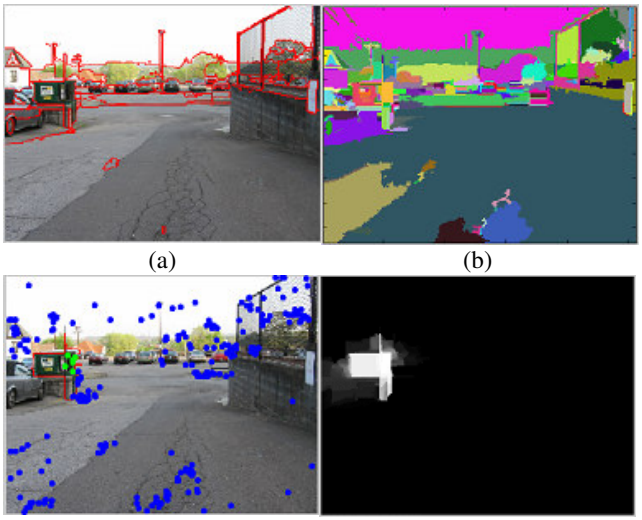


Fig. 1. Example of jigsaw puzzle segmentation algorithm: (a) The input scene image with final segmentation results (b) The superpixels of the scene image obtained with the method from [22] (c) A trash can consists of multiple superpixels, one of them is being tested: the green points are the seeds set in the testing superpixel, the blue points are the seeds set in background (the sky and ground) and all the known objects (d) Testing result: high probabilities is mapped to high intensity (white) and low probabilities is mapped to low intensity(black). It is clear to see that the whole trash can "looks" totally different from its background.

This work was supported in part by the University Research Program in Robotics under grant DOE-DE-FG52-2004NA25589.

Chang Cheng, David L. Page, and Mongi A. Abidi are with IRIS Lab, Electrical and Computer Engineering Department, The University of Tennessee, Knoxville, TN 37996 USA. e-mail: ccheng1@utk.edu.

thousand or even millions of features might be formed, which will make query very inefficient and therefore make on line loop detection difficult. A more efficient way is to separate some interesting objects from a scene through image segmentation and only keep those salient objects with a certain number of affine covariant features inside as landmarks. In this way we may greatly reduce the information amount needed for uniquely labeling a place and make the matching process more efficient.

In this work, we propose a novel jigsaw puzzle image segmentation algorithm to segment a scene image to regions that may correspond to objects or parts of objects. Based on these image regions, we can detect a set of salient objects from a scene. We also design a range tree database to speed up the scene matching process.

The remainder of this paper is organized as follows: In section II, we introduce the jigsaw puzzle image segmentation algorithm. In section III, we describe our place recognition and loop detection algorithm in detail, addressing how to detect salient objects and how to build a range tree database for efficient object matching. The experimental result is presented in Section IV. Section V concludes this paper.

II. JIGSAW PUZZLE IMAGE SEGMENTATION

In general, image segmentation approaches can be divided into two categories: methods based on low-level features (bottom-up) and methods based on prior knowledge about the scene (top-down)[14]. The goal of image segmentation is to divide the image into regions of coherent properties so that each region corresponds to a coherent object.

However, none of the many bottom-up segmentation methods can achieve this goal. It was recently argued that any bottom-up segmentation method cannot be possibly expected to partition an image into its constituent objects because it does not know where one object ends and another one begins [23]. As a result, bottom-up methods may cause a serious over-segmenting problem: an object may be segmented into multiple regions and it is difficult to regroup these regions back to that object because to achieve that, one needs to solve the object recognition problem first [18].

The top-down methods attack this difficulty by using prior knowledge about an object, such as its possible shape, color or texture, to guide the segmentation. The difficulty for top-down segmentation is that it requires a learning stage to acquire class-specific information and can only be applied to images from a specific class. Due to the large variability in the shape and appearance of objects within a given class, the top-down methods may not accurately delineate the object's figure-ground boundary [19]. Plus the high computation cost of top-down methods also makes them not applicable to real-time applications like robotic place recognition.

Regions in an over-segmented image are often called "superpixels" [20]. Although not having semantically correct information, these "superpixels" still provide certain space support that is necessary for segmentation at the scale of interest. (Superpixels are local, coherent and respect segment boundary[20],[21]). To identify a set of superpixels that belong to a single object, we need to have some knowledge about the scene. An important observation from [20] shows that over 97% of outdoor image pixels can be classified as either being part of the ground plane, belonging to a surface that sticks up from the ground or being part of the sky. This observation can also apply to indoor images (in a indoor case, the sky is replaced by a ceiling and wall, the ground plane is replaced by a floor). Superpixels belonging to the sky or ground plane can be easily detected because they are usually located in the top or bottom area of an image and in most cases should be uniform image patches.

Our new image segmentation algorithm is motivated by a jigsaw puzzle game. Each superpixel can be viewed as a "puzzle piece." Our task is to assemble these puzzle pieces into objects in the scene. One well known strategy for playing a jigsaw puzzle is starting by separating the edges from the inside pieces and connecting the outside edges, then working inward. Our method takes the similar strategy: first "assemble" the outside edges (sky and ground plane), then work inward. Our strategy involves always working on easy portions in which objects look totally different from their background first and leaving the difficult portions where multiple objects look similar with each other to the last stage when most of the neighbor areas of the portion are clear.

The basic rule to group a set of superpixels to a single object is based upon this observation: although there may be certain differences among the different parts (superpixels) of the object, the whole object should look totally different from its background if that object is distinguishable. We use the approach proposed in random walks image segmentation algorithm [13] to measure the difference between a single object and its background. Notice that our jigsaw puzzle image segmentation algorithm still belongs to the bottom-up approach because we do not have a learning stage and do not have any prior knowledge about the objects existing in a scene. Our goal here is just to attempt to detect enough objects from a scene and use these objects as landmarks to label a place, not to produce a perfect-segmenting image. Because to achieve that, one needs to apply some knowledge-based methods to make the correct decision everywhere in the image, which is out of the scope of our work. We may miss a few objects, but on the whole, we segment the most salient objects, which should be sufficient for place recognition.

A. Random walks image segmentation

Random walks [13] is an interactive image segmentation method. Assuming all pixels X in an image are connected to their neighbors with edges $e \in E \subseteq X \times X$. Given a small number of pixels X_M with user-defined or pre-defined labels, starting at each unlabeled pixel $x \in X_U$ ($X_M \cup X_U = X$), this algorithm can quickly determine the probability that a

random walker will first reach one of the pre-labeled pixels $x_m \in X_M$. Based on the connection between random walks on graphs and discrete potential theory, this probability can be calculated by solving the circuit theory that corresponds to a combinatorial analog of the Dirichlet problem.

If every region is correctly seeded (each constant region corresponding to a single object is assigned with a label), the random walks algorithm can produce a segmentation that respects even a weak object boundary. What is of interest to our research is how it assigns probabilities to these ambiguous regions (unseeded regions). For example, if a superpixel v_i corresponding to object i is assigned with a label and a set of superpixels V_B corresponding to the background like the sky and ground plane are assigned with another label, for a unlabeled superpixel v_j corresponding to object j, if it looks more like v_i (based on color), it will be assigned with higher probabilities. If it looks more like the background, it will be assigned with lower probabilities. This probability information is useful as a measurement for the degree of how similar the local neighbors of an object look like it comparing to the known background. For more details, see[13].

B. Jigsaw puzzle algorithm

Following [20], given a scene image, our first step is to apply the method in [22] to obtain a set of superpixels. We also extract a set of SIFT keypoints [2] from the image. These SIFT keypoints can be used to test the uniformity of the superpixels. Then a graph $G = (V, E)$ is built based on this set of superpixels. Each vertex $v \in V$ corresponds to a superpixel. An edge e_{ij} is added to E if v_i and v_j are neighboring. The next step is detecting the superpixels belonging to the sky and ground plane. First the uniform superpixels located in the top and bottom portion of the image are selected. The uniformity can be tested by the following equation:

$$uniform\ (v_i) = \exp(-\alpha \frac{n_i}{s_i}) \quad (1)$$

Where α is constant, n_i is the number of SIFT keypoints contained in superpixel v_i and s_i is the area of v_i.

If the $uniform(v_i) > threshold$ (in our work, we choose the $threshold = 0.98$), v_i is a uniform superpixel. If v_i is selected to being part of the sky or ground plane, then all its similar neighboring superpixels are also selected to being part of the sky and ground plane. The similarity is measured with cylindric distance in HSV color space [16]:

$$D_{cyl}(v_i, v_j) = \sqrt{\Delta M^2 + S_i^2 + S_j^2 - 2S_i * S_j * \cos(\Delta H)} \quad (2)$$

where S_i, S_j are the mean saturation of v_i and v_j, and ΔM, ΔH are the difference of the mean value and hue of v_i and v_j.

All the superpixels selected to being part of the sky or ground plane are put into one set V_k that contains all the known superpixels, the remaining superpixels are put into another set V_{uk} ($V_k \cup V_{uk} = V$). We further divide the V_{uk} into two parts: V_{uk_b} and V_{uk_s}. We sort all the superpixels in V_{uk} based on size and put the first m large superpixels into V_{uk_b}. The total size of these m superpixels accounts for more than 70% of the total size of all the superpixels in $V_{uk.}$. V_{uk_s} contains the remaining small superpixels in V_{uk}. Any superpixel $v_b \in V_{uk\_b}$ may correspond to the whole surface of a single object or the main part of surface of a single object. So we need to be very careful of grouping any two of the superpixels belonging to V_{uk_b} together unless we have enough evidence. The superpixels belonging to V_{uk_s} may correspond to the details or noise. Some mistakes of grouping these superpixels will not affect the final segmentation quality too much.

The next step is to go through each of the large superpixels in V_{uk_b}. For each $v_b \in V_{uk\_b}$, we want to measure the difference between v_b and its neighbors. This is done by repeating the following procedure: for each $v_i \in V_k$, find its untested neighbor $v_b \in V_{uk\_b}$ and put v_b into a set V_{ut} that contains all the untested superpixels. Then for each $v_j \in V_{ut}$, set a seed with *label=1* in v_j and set a set of seeds with *label=2* in each $v_i \in V_k$ and run the random walks algorithm to get the similarity probabilities for all the unseeded superpixels in V_{uk}. If all the neighbors of v_j have low probabilities, that means that v_j looks totally different from its neighbors and it may correspond to a single object surface. So v_j is directly put into V_k. If a neighbor $v_n \in V_{uk\_b}$ has high probabilities, that means that comparing to the background and the all known objects, v_n looks more similar to v_j. A decision should not be made until after v_n has been tested. The similarity probabilities relating to v_j is recorded in v_n and v_j is put into another set V_{ud} that contains all the undecided superpixels. This is repeated until all the superpixels in V_{uk_b} has been tested. At this point, all the superpixels originally belonging to V_{uk_b} should either be in V_k or be in V_{ud}.

Then for each superpixel in V_{ud}, find its undecided neighbors, see if any two of them both agree that they look similar to each other. If there exists a similar pair of superpixels, then merge them together. Otherwise leave each superpixel separated. After this merging process is finished, put all the superpixels in V_{ud} into V_k. An example of this testing process is shown in figure 1.

The last step is to group those small superpixels in V_{uk_s}. At this point, we have already made a decision on all the large superpixels and these small superpixels have recorded the similarity probabilities relating to their different large neighbors respectively. So a small superpixel may belong to any one of its large neighbors or belong to an independent small object. The basic rule here is: if a small superpixel has low similarity probability to all its large neighbors, it is treated as an independent small object. If it has high similarity probabilities to one large neighbor, assign it to that large superpixel. If it has high similarity probabilities to multiple large neighbors, assign it to the one with highest

similarity probabilities. Repeat this until all superpixels are processed.

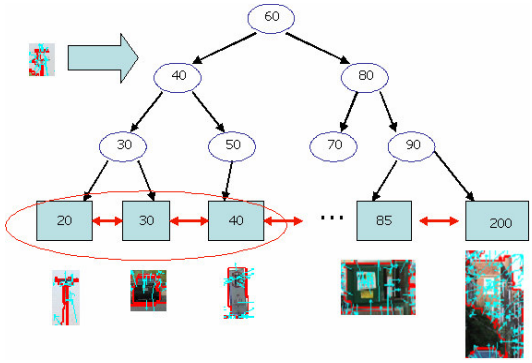


Fig. 2. Illustration of a range tree search: the query object contains 30 SIFT features, those objects that contain between 20~40 SIFT features in the database can be quickly found

III. PLACE RECOGNITION AND LOOP CLOSING

Based on our jigsaw puzzle image segmentation algorithm, we developed a new place recognition and loop closing method. Place recognition for human perception most likely operates on the level of objects [24] since when one looks at a place, he most likely does not remember every detail of the scene Instead, he tries to find several impressive objects in that scene and use them to uniquely label that place. Our approach takes a similar strategy.

A. Salient object detection

Based on the image regions produced by the jigsaw puzzle image segmentation algorithm, we want to detect a set of "impressive" objects. These kinds of objects are called salient objects and they are highly valuable as visual landmarks. We define two criteria to select such salient objects:

1. The objects should not be too big; and
2. The objects should be highly distinctive.

The reason for the first criterion is because when a vehicle revisits a place, most likely it will see that scene from a different view point. For those small-sized/middle-sized objects, it is highly possible that they can be totally seen from both different view points. In contrast, large-sized objects may be occluded by some other objects and hence look different from different view points. The second criterion is common sense: as landmarks, these salient objects should be distinctive enough to uniquely label a scene. Since SIFT descriptors are highly distinctive, the more SIFT features an object contains, the more distinctive it is.

Based on these two criterions, the saliency of regions is estimated by:

$$saliency\ (v_i) = \exp(-\beta \frac{s_i}{n_i}) \qquad (3)$$

where β is a constant; s_i is the area of the image region; n_i is the number of SIFT features contained inside the image region. Any region that has saliency value higher than a threshold value will be selected as a salient object. Each salient object is represented by a list of SIFT descriptors contained inside it. A set of these kinds of salient objects can be expected to be able to uniquely label a place.

B. Build range tree database

Each scene is represented by a set of salient objects detected by equation (3). We further select a subset of the salient objects that can be reliably observed from multiple positions as index objects. This is achieved by comparing the set of salient objects obtained in current position with those obtained in previous positions. If a vehicle is revisiting this area, then several of these index objects are expected to be observed. This small set of index objects may not be able to uniquely label a place, but in a large environment, they may help us to quickly narrow down the search to several potential sites. We will use these index objects to build a query database. In a large environment, it is often the case that even the query database may contain thousands of index objects. To increase query efficiency, we need to use some stable characteristics of these index objects as an index to quickly find the group of objects that are similar to the query object. Since these salient objects may be observed over time from different angles and distances and under different illuminations, the shape, size and color of the objects are all unstable. But the number of SIFT features contained in an object is stable in a certain range. According to [2], the stability of detection for SIFT features should be around 70%~90% under different degrees of affine distortion. So for a query object containing 30 SIFT features, most likely the matching object will be among those objects that contain 20~40 SIFT features. For this reason, we design a range-tree [15] structure database to help us quickly find a range of candidates in a large database.

A range tree is a balanced binary search tree where the data are sorted in the leaf nodes and these leaf nodes are linked in sorted order by use of a doubly linked list. The non-leaf nodes contain midrange values that enable discriminating between the left and right subtrees. A range search for [*B: E*] can be performed by searching the tree and finding the node with either the largest value ≤ *B* or the smallest value ≥ *B*, and then following the links until reaching a leaf node with a value greater than *E*. For *N* points, this process takes $O(\log_2 N + F)$ time. *F* is the number of objects found. An example of a range tree search is shown in Fig. 2.

C. Place recognition and loop closing algorithm

The complete place recognition and loop closing algorithm is summarized as follows:

INPUT: image I_c taken from current scene c with current

time t_c

OUTPUT: place recognition (loop detection) result

1. Detect SIFT features from I_c;
2. Run jigsaw puzzle algorithm to segment I_c;
3. Detect a set of salient objects O_c with equation (3) and a set of index objects $IO_c \subseteq O_c$;
4. Set $M = \phi$; For each index object $o_i \in IO_c$, search in the range tree. If a match o_m is found, output the scene number set s (records the these scenes that o_m appears) of o_m and let $s = s \cup c$, $M = M \cup s$; otherwise insert o_i into the range tree with scene number set c;
5. For each scene number $m \in M$, compare O_c and O_m, if $|O_c|$ is close to $|O_m|$ and around 50% of O_c get matched with O_m and there is a big difference between t_c and t_m, then detect a loop. (t_m is the time when m was visited); if $M = \phi$ or scene c is not matched with any scene in M ,output that c is a new scene.

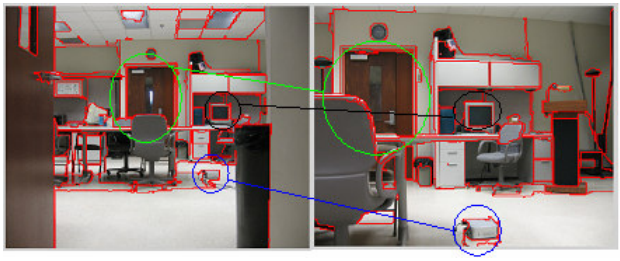
IV. Experimental Results

We tested our place recognition and loop closing algorithm using imagery from our *AndiBot* mobile robot which is designed for applications like automated perimeter surveillance for high security buildings and facilities (see Figure 3). The robot collected 100 images from a 150m long indoor loop. In the outdoor environment, it traveled over a trajectory of 800m, collecting 220 images. Figure 4 and 5 show two examples of the loop detection results in indoor and outdoor environments, respectively.

Fig. 3. *AndiBot* platform for perimeter surveillance

In both indoor and outdoor environments, our method can quickly detect a loop except for one case: in the hallway of the indoor environment. At two different places of the hallway, there are nothing but wall and floor. Our algorithm could not detect any salient objects from these two places and therefore could not distinguish these two places. These kinds of places are common challenge for any appearance-based place recognition method. Even human beings may have trouble distinguishing some places without any special characteristic like these.

Our method also shows certain robustness handling multiple instances of the same object class spreading in different sites. For example, our indoor environment consists of four office rooms where each room has similar types of chairs and computers. But our method did not get confused by these chairs and computers and successfully distinguished each room. The main reason is that in each room, we detected more than 15 objects, nearly half of them are unique for that room. In the future, we are considering adding relative position information of the salient objects to further increase the robustness.



(a) (b)

Fig. 4. Loop detection in an indoor environment: (a) was the image taken close to the point of loop-closure (b) was the image taken at the starting point. About 10 pair of objects got matched. The green, blue and black circles show three pairs of the matched objects.

(a) (b)

Fig. 5. Loop detection in an outdoor environment: (a) was the image taken close to the point of loop-closure (b) was the image taken at the starting point. About 12 pair of objects got matched. The green, blue and black circles show three pairs of the matched objects.

In both indoor and outdoor environment, only 30% of detected SIFT descriptors were recorded in the database. Table 1 summarizes the storage efficiency of our method in both environments.

TABLE 1
Storage efficiency in indoor and outdoor environments

	Total of detected objects	**Total of detected SIFT features**	**Total of SIFT features in database**
indoor	612	46,338	14,689
outdoor	3,078	402,281	114,931

Figure 6 presents the query efficiency for the SIFT database and the range-tree database. Six sets of images (ranging from 60 to 160) were selected from the outdoor

image collections. For each set of images, two databases were built. One was a SIFT database formed by all the SIFT features extracted from these images. The other one was a range-tree database formed by all the salient objects and index objects detected from these images. For each database, two queries were executed. One query image was selected from the set of images the database was built on and the other one was not. As expected, the query efficiency was much higher in the range-tree database than in the SIFT database. One can see that for the SIFT database, the query performance quickly dropped down when the database size increased. Especially for the unmatched query, all the SIFT features in the database had to be compared to decide that there was no match. For the range-tree, there was no big difference between the matched and unmatched query because it is an object-based database. For any query object, only a small range of objects in the database need to be compared. We claim that the range-tree is more efficient for a large environment.

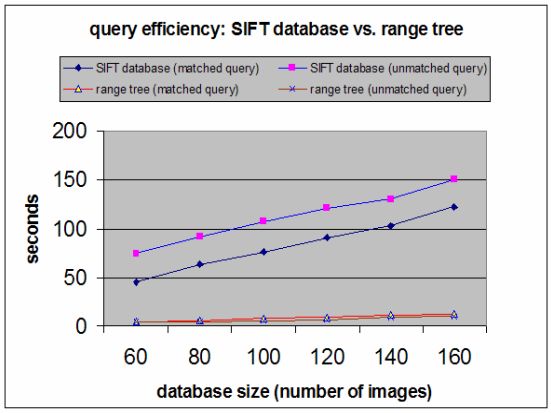


Fig. 6. Query efficiency: SIFT database vs. range-tree.

V. Conclusion

In this paper, we present a novel place recognition algorithm. Unlike the work in [17], our method does not require a learning stage. Like human beings, our method operates on the level of objects. Instead of directly using large numbers of SIFT features as visual landmarks, we first used jigsaw puzzle image segmentation algorithm to segment the input scene image to regions that may correspond to objects or parts of objects. Based on these image regions, we further detected a set of salient objects to represent a place and only those SIFT descriptors that were contained in these salient objects were kept in the database. We also designed a range tree structure database to quickly find the range of potential matching objects for any query object. Experiments showed that in both indoor and outdoor environments, in most cases our method can quickly detect a loop. In our approach, only less than 30% of total SIFT features were used for place recognition and the query process was also very efficient.

References

[1] P. Newman and K. Ho, "SLAM-Loop Closing with Visually Salient Features," Proceedings of the 2005 IEEE, International Conference on Robotics and Automation, April 2005.

[2] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision, vol. 60, No. 2, pp. 91-110, Nov.2004 9.

[3] J. Matas, O.Chum, M.Urban, and T. Pajdla. "Robust wide baseline stereo from maximally stable extremal regions,",Proceedings of the British Machine Vision Conference, 2002.

[4] T. Kadir and M. Brady. "Saliency, scale and image description," International Journal of Computer Vision, 45(2):83-105, 2001

[5] K. Mikolajczyk and C. Schmid. "A Performance Evaluation of Local Descriptors,".In IEEE Conference on Computer Vision and Pattern Recognition, volume 2, pages 257-264, 2003.

[6] K. Mikolajczyk, T. Tuytelaars, C. Schmid and A. Zisserman, "A Comparison of Affine Region Detectors,", International Journal of Computer Vision, Volume 65, Number 1/2 – 2005.

[7] N. Tomatis, I. Nourbakhsh and R. Siegwart, "Hybrid Simultaneous Localization and Map Building: A Natural integration of Topological and Metric," Robotics and Autonomous Systems 44 (2003) 3–14.

[8] M. Bosse, P. Newman, J. J. Leonard, and S. Teller. "SLAM in large scale cyclic environments using Altlas framework," International Journal of Robotics Research, 23(12): 1113-1139, Dec 2004.

[9] M. Montemerlo, S. Thrum, D. Koller and B. Wegbreit, "FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem," Proceedings of the AAAI National Conference on Artificial Intelligence, 2001.

[10] S. Se, D. Lowe and J. Little, "Mobile Robot Localization And Mapping with Uncertainty using Scale-Invariant Visual Landmarks," The International Journal of Robotics Research, 2002.

[11] S. Se, and D. G. Lowe, "Vision-based global localization and mapping for mobile robots,",IEEE Transactions on robotics, vol 21. No. 3, June 2005.

[12] P. Lamon, I. Nourbakhsh, B Jensen and R Siegwart, "deriving and matching image fingerprint sequences for mobile robot localization," Robotics and Automation, 2001. Proceedings 2001 ICRA.

[13] L. Grady. "Random Walks for Image Segmentation," IEEE Transaction on pattern analysis and machine intelligence, vol. 28, No. 11, Nov 2006.

[14] U. Rutishauser, D. Walther. "Is bottom-up attention useful for object recognition?" in Proc. CVPR, 2004

[15] H. Samet. "Foundations of multidimensional and metric data structures," Morgan Kaufmann Publishers, 2006.

[16] K. Plataniotis and A. Venetsanopoulos, "Color image processing and applications," Springer, Ch.1 pp268-269, 2000.

[17] M. Cummins and P. Newman, "Probabilistic appearance based navigation and loop closing," IEEE international conference on Robotics and Automation, April 2007.

[18] B.C. Russell, "Using Multiple Segmentations to discover objects and their extent in image collections," In Proc. CVPR, 2006.

[19] E. Borenstein, E. Sharon, "Combining top-down and bottom-up segmentation," In Proc. CVPR 2004.

[20] D. Hoiem, "Geometric context from a single image", In Proc. ICCV, 2005.

[21] X. Ren, "Learning a classification model for segmentation", In Proc. ICCV, 2003.

[22] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation", IJCV, vol. 59, no. 2, 2004.

[23] T. Malisiewicz and A.A. Efros. "Improving Spatial Support for Objects via Multiple Segmentations," BMVC, 2007.

[24] D. J. Simons and D. T. Levin, "What makes change blindness interesting," In *The Psychology of Learning and Motivation*, Eds. D. E. Irwin and B. H. Ross, Academic Press, San Diego, CA, vol. 42, pp. 295-322. 2003