2009 Ninth International Conference on Intelligent Systems Design and Applications

Context-based adaptive filtering of interest points in image retrieval

Giang P. Nguyen Hans Jørgen Andersen
Department of Media Technology,
Aalborg University, Denmark
Email: {gnp,hja}@imi.aau.dk

***Abstract*—Interest points have been used as local features with success in many computer vision applications such as image/video retrieval and object recognition. However, a major issue when using this approach is a large number of interest points detected from each image and created a dense feature space. This influences the processing speed in any runtime application. Selecting the most important features to reduce the size of the feature space will solve this problem. Thereby this raises a question of what makes a feature more important than the others? In this paper, we present a new technique to choose a subset of features. Our approach differs from others in a fact that selected feature is based on the context of the given image. Our experimental results show a significant reduction rate of features while preserving the retrieval performance.**

***Keywords*-Image retrieval, interest point detection.**

I. Introduction and related work

The growing of image databases with large varieties in image conditions such as geometrical and illumination changes leads to the need for invariant features. Local features are widely used because of their stability under different imaging conditions and their success in many computer vision applications [8], [13]. Among those, features extracted from interest points, which are found at different types of junctions, on contrast areas, or texture areas, are used very often [7], [5], [8]. A good overview of existing works can be found in [13]. There are several well-known techniques including Scale Invariant Feature Transform (SIFT) [7], PCA-SIFT [5], and Multi-Scale Oriented Patches (MOPS) [1]. These techniques usually introduce a large number of descriptors. For instance, SIFT descriptor in general creates approximately 2000 descriptors for an image with size 500×500 pixels [7]. Therefore, the main goal of runtime processing systems is to deal with the large amount of data.

In [5], the authors present a method of reducing the dimensionality of SIFT descriptors, using the PCA dimensional reduction method which projects the original SIFT feature space from 128 dimensions to 20 dimensions. The PCA-SIFT method achieves significant space benefits and requires a third of the time in the matching phase compared to the original SIFT. A different approach is put forward in [11] where a vocabulary tree is used to index descriptors. The K-means algorithm is used to cluster all descriptors and place them in the correct branch. For each query image, extracted descriptors are traced down the tree, a score list is given for all leaves, and the one with the highest score is returned as the best match. This approach has proved to be very fast and scalable to a very large number of descriptors. In [12], another approach is proposed that is instead of comparing all features, a subset of features that are within a fixed radius around each point is considered for computation.

The above approaches do not alter the original number of features. This means that one need to compute all the descriptors for every detected points before any further step takes place. The computing descriptors is much more time consuming compared to the finding interest points step. Therefore, another approach is to first reduce the number of interest points, then compute the descriptors of selected points only. In [10], the authors also experiment that not all extracted points are equally important i.e. some are irrelevant in the retrieval phase. In this reference, it is proved that having too many descriptors can reduce the recognition rate. For these reasons, attention should be focused only on those feature points that are informative.

In developing techniques for selecting descriptors, it is generally assumed that certain descriptors are more important than others. The terms "discriminative" and "informative" are usually used to describe significant descriptors. In [6], the authors observe that certain features are more stable and thus able to being better handle variations in scale and viewpoint. They therefore aim to select such features. For each feature extracted by means of the SIFT detector from each image at each location, they calculate a posterior probability. The probability values are used as ranking criteria. In [1], the authors present an adaptive non-maximal suppression (ANMS) algorithm that selects a subset of interest points based on their corner strength. The general idea of this algorithm is that for each point extracted through the process described above, they calculate the corner strength, and then select points that are maximum within their neighbourhood of radius k pixels. In all their experiments, the authors select a maximum of 500 points for each image. This means a set of 500 descriptors is used to describe the content of an image. Another technique for selecting informative (i-SIFT) descriptors, using the SIFT detector, can be found in [3]. For each given image, informative descriptors are defined as those that appear in discriminative regions. These regions are detected on the basis of an entropy-coded image derived by calculating posterior distribution. Also in [9], unique features which are stand out in the feature space are chosen. Most of these methods select a fixed number of features by considering their relations with nearby neighbors

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.25

only.

In this paper, we also focus on developing a technique for selecting a subset of descriptors. Our approach differs from existing ones where we take into account the context of the image. This means that an image with a complicated scene (e.g. busy background, lots of textures, or overlapped objects) should be expected to have more features than an image with a simple scene, as well as more features should be located in the part with more details in an image. Our approach assures that the distribution of selected points should reflect this variety of image context. Depending on the complexity of a given image, our approach adaptively finds an appropriate number of features. Moreover, there is always a trade off between speeding up the system and the accuracy or retrieval rate. In developing a new technique, we also aim at balancing these two issues i.e. keeping the performance while reducing the size of the feature space for an efficient process. In the next section, we will describe in more details our approach in selecting features. Experiments are carried out in section III. Conclusions wraps up the paper in section IV.

II. Our approach

General local feature detector consists of two steps. The first is to find interest points, and then describe a small region centered at each point and convert this to a descriptor. In this section, we will look into each step. Once all interest points have been found, we will apply our proposed technique for selecting the most informative set.

A. Multi-scales interest point detector

For finding interest points, multi-scale Harris detector is one of the most common approaches [13]. The Harris corner detector was first introduced in 1986 by Harris and Stephens [4]. This detector is based on the auto correlation matrix to describe local image structures. This method is able to find points that are located in areas where image significantly varies in both directions by measuring the cornerness. Points detected by this method are shown to be invariant under rotation and translation only. Alternatively, this method has been improved to be invariant to scale as well [8], [7], [1], which is so called multi-scales invariance. We chose the Multi-Scales Oriented Patches (MOPS) in [1] as a starting point for detecting local features. A brief description of this method is given as follows:

Let us take an image set $\mathcal{I}$. Each input image $I_i \in \mathcal{I}$ is incrementally smoothed with a Gaussian kernel $\{\sigma_t\}_{t=1..n}$. An image pyramid is then constructed by down-sampling the image at rate r (see figure 1). In the second step, interest points are extracted using the Harris corners detector at each level of the pyramid. This step yields a set of points at locations where the corner strength is a local maximum of a 3×3 neighborhood and above a threshold of 10 [1]. In the next step, the sub-pixel precision is found by means of a Taylor expansion (up to the quadric term) at those extreme points. An example of features extracted by MOPS is shown in figure 2.

At this stage, we are able to obtain a set of interest points. In the following section, we describe the technique for filtering non-important points.

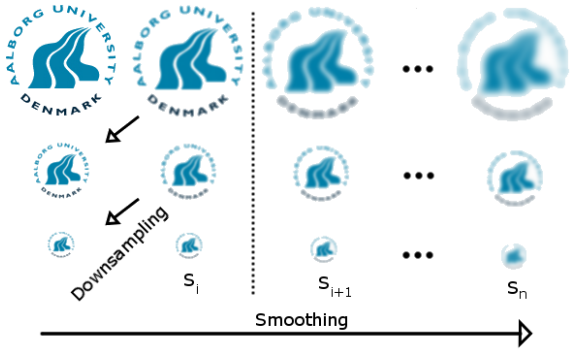


Fig. 1. An illustration of image pyramid defined in MOPS.

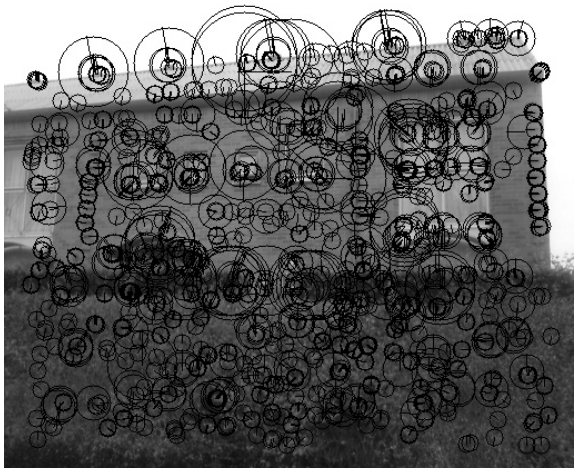


Fig. 2. An example using MOPS, the size of circles defines different scales in the image pyramid.

B. Image content representation

Different from existing techniques for selecting final interest points, our selection mechanism takes into account the whole image context. The image context often contains different patches where some contain more details than the others. For example, an image with a bicycle placing on a grass area, other methods will find many points in the grass area because of its textural surface. In our approach, we consider that the grass patch is homogenous region so it is only required a small number of points to represent the whole area, and more points should be located on the patch with the bicycle. For that we need to find a way to represent the whole image into a number of patches, where each patch is a homogenous region. The B-Tree triangular coding method introduced by Distasi et. al. [2] meets this requirement.

B-Tree triangular coding (BTTC) is a method originally designed for image compression. A given image I is considered as a finite set of points in a 3-dimensional space, i.e. $I = \{(x, y, c)|c = F(x, y)\}$ where (x, y) denotes pixel

position, and c is an intensity value. BTTC tries to approximate I with a discrete surface $B = \{(x, y, d)|d = G(x, y)$, defines by a finite set of polyhedrons. In this case, a polyhedron is a right-angled triangle (RAT). Let assume a RAT with three vertices (x_1, y_1), (x_2, y_2), (x_3, y_3) and $c_1 = F(x_1, y_1)$, $c_2 = F(x_2, y_2)$, $c_3 = F(x_3, y_3)$, we have a set $\{x_i, y_i, c_i\}_{i=1..3} \in I$. The approximating function $G(x, y), (x, y) \in$RAT is computed by the linear interpolation:

$$G(x, y) = c_1 + \alpha(c_2 - c_1) + \beta(c_3 - c_1) \quad (1)$$

where α and β are defined by the two relations:

$$\alpha = \frac{(x - x_1)(y_3 - y_1) - (y - y_1)(x_3 - x_1)}{(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)} \quad (2)$$

$$\beta = \frac{(x_2 - x_1)(y - y_1) - (y_2 - y_1)(x - x_1)}{(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)} \quad (3)$$

An error function is used to check the approximation:

$$\text{err} = |F(x, y) - G(x, y)| \leq \epsilon, \epsilon > 0 \quad (4)$$

If the condition does not meet then the triangle is divided along its height relative to the hypotenuse, introducing two other RAT. The coding scheme is recursively until no more division takes place. In the worst case, the process is stopped when it reaches to the pixel level i.e. three vertices of a RAT are three neighbor pixels and err=0. The decomposition is arranged in a binary tree. Without loss of generality, the given image is assumed having square shape, if not the image is padded in a suitable way. With this assumption, all RAT will be isosceles. Finally, all points at the leave level are used for the compression process. Figure 3 shows an illustration of the above process. Examples using BTTC to represent image context are shown in figure 4.

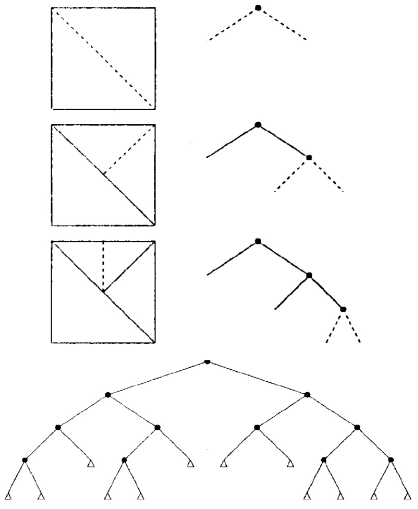
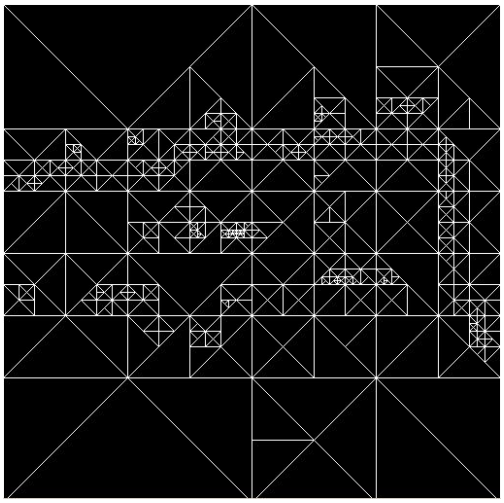


Fig. 3. An illustration of building BTree using BTTC. The last figure shows an example of a final BTree.

In the reference [2], experiments prove that BTTC produces images of satisfactory quality in objective and subjective point of view. Furthermore, this method is very fast in execution



(a)

(b)

Fig. 4. Examples using BTTC: (a) all RATs are drawn for observation. (b) vertices of RATS are embedded to the image.

time, which is also an essential factor in our selection process. We note here that for encoding purpose, the number of points (or RAT) is very high (up to several ten thousand points depends on the image context). However, we do not need that detail level, by setting the error approximation larger we can obtain less points while still preserve homogenous region criteria. We experimentally set $\epsilon = 50$ in all experiments.

C. Point filtering

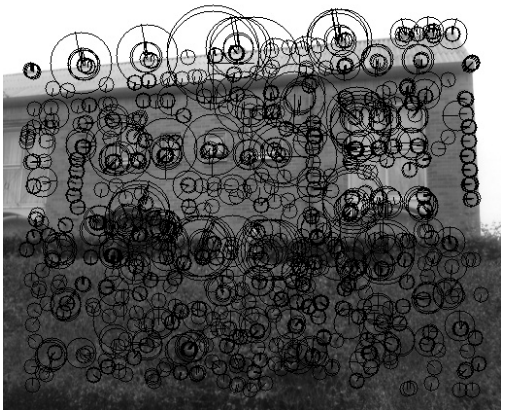
After finding all image patches, we will filter out non-important points. Because each patch is a homogenous region, only one point is needed to represent that region. We first superimpose all interest point found in section II-A and the BTTC representation into one image. Note that BTTC representation contains only RAT at the leave level. At each RAT, we find all points within that RAT including points lying on the edges. Given a RAT defined by three vertices

$P_1(x_1, y_1), P_2(x_2, y_2), P_3(x_3, y_3)$, and a point $P(x, y)$, we compute the Barycentric coordinates to detect if P is inside the triangle RAT(P_1, P_2, P_3):

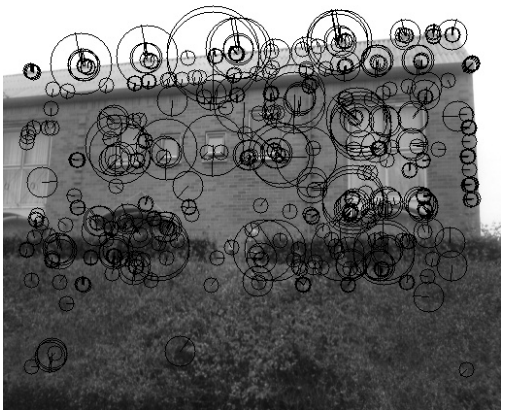
$$T_1 = ((P_2 \circ P_2) * (P_1 \circ P_3) - (P_1 \circ P_2) * (P_2 \circ P_3)) * T$$
$$T_2 = ((P_1 \circ P_1) * (P_2 \circ P_3) - (P_1 \circ P_2) * (P_1 \circ P_3)) * T$$

where $T = 1/((P_1 \circ P_1) * (P_2 \circ P_2) - (P_1 \circ P_2) * (P_1 \circ P_2))$ and $\circ$ denotes the dot product. Then, $P \in$ RAT(P_1, P_2, P_3) when $T_1, T_2 > 0$ and $T_1 + T_2 < 1$.

For selecting the representative point for that RAT, point with highest corner strength is chosen. The process is repeated until all the RAT are checked. Only for the set of representative, descriptors will be computed. Figure 5 show some examples before and after filtering with BTTC.



(a)



(b)

Fig. 5. Examples using BTTC for point filtering.

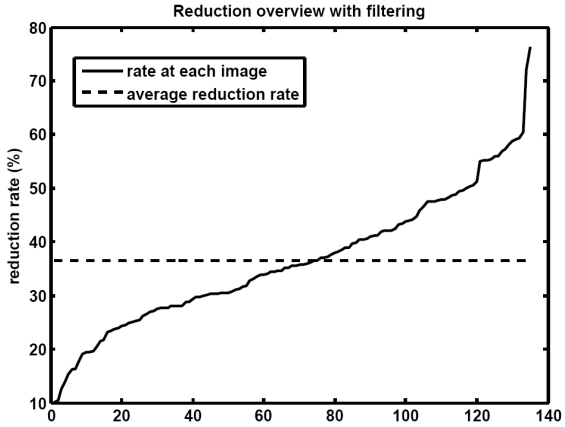
D. Descriptor computation

Computing descriptor is done similar to what described in [1]. Each representative point is described in terms of its orientation within a window of size 28×28 (corresponding to a Gaussian kernel with $\sigma = 4.5$), and through sampling of grey level values in a 40×40 neighborhood. The grey level values are sampled in a grid with a spacing of 5 pixels rotated according to the orientation. This gives a feature vector for each landmark consisting of 8×8 grey level values. Before matching, the feature vector is standardized by subtracting the mean and dividing it by its standard deviation. Then, as in [1], we perform a Haar wavelet transform on the 8×8 descriptor patch to form a feature vector of 64 dimensions F_j.

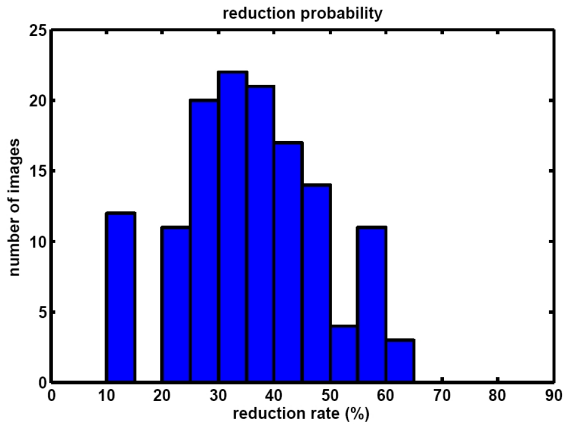
III. Experimental results

Our experiments are carried with two different datasets. The first dataset is called the AAU dataset, which contains of 135 images. Images are captured of 21 buildings in the Aalborg University area. The second dataset, which is called the centrum set, contains of images taken in the Aalborg center. This is a set of 442 images of 19 buildings. To create the diversity, images in these datasets are taken by different persons, at different times and different days during one year.

Our first experiment is setup to see how our approach can reduce the number of features. To do that, we compute for each image the number of interest points with and without using BTTC. Results are shown in figures 6 and 7 with the AAU dataset and the centrum set, respectively.

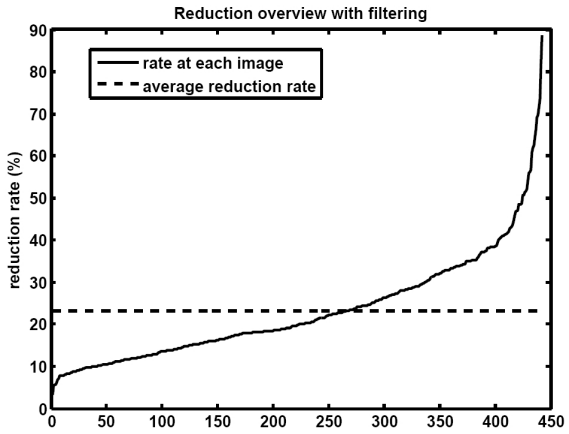


(a)

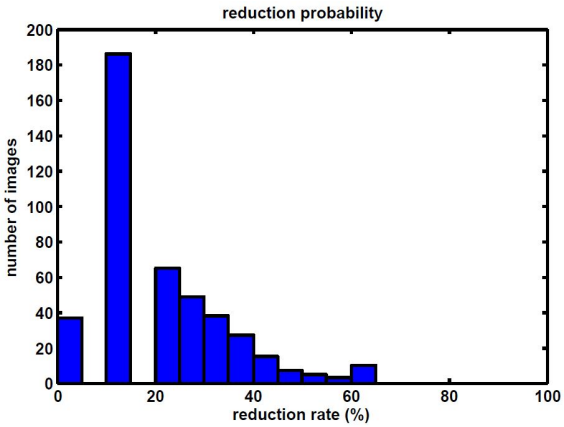


(b)

Fig. 6. Reduction rate of the AAU testset. (a) Reduction rate for each image (in a sorting order). (b) Reduction probability.



(a)

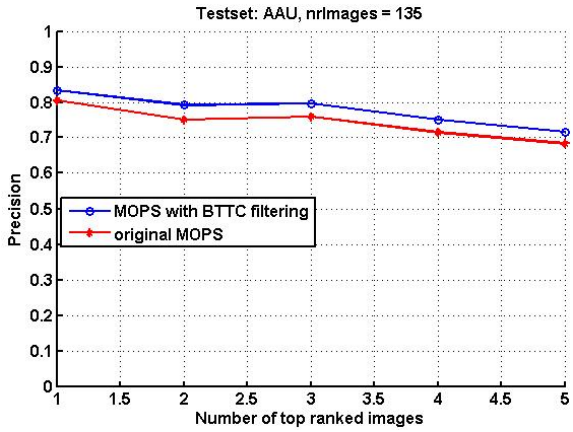


(b)

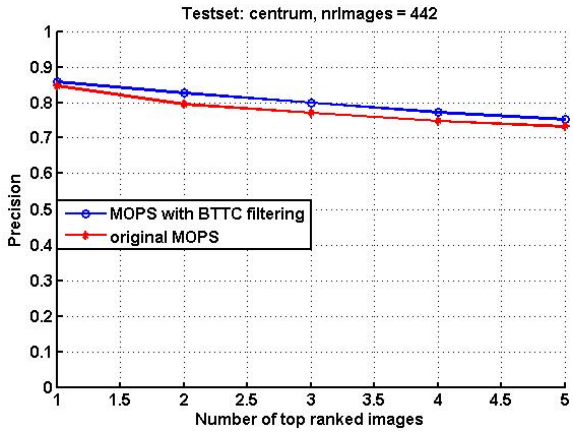
Fig. 7. Reduction rate of the Centrum testset. (a) Reduction rate for each image (in a sorting order). (b) Reduction probability.

The dotted-lines show the averaged reduction rate for the whole datasets. The other lines show reduction rate at each image, these lines are sorted increasingly. The maximum values on the x-axis are the sizes of the two datasets. In figure 6a, the average reduction rate for the AAU dataset is 37%. Approximate 50% of the images in this dataset have more than 37% of reduction rate, and the highest reduction rate is 78%. For a better view, we draw a histogram of the reduction probability where each column shows the number of images at a certain reduction rate (see figure 6b). In case of the centrum dataset, the average reduction rate is 23%. Compared to the AAU dataset, this dataset in general is much more complex in the image content, therefore, more points should be used to represent the image.

Now we know that the number of features can be reduced significantly, we need to prove that this process does not come at the cost of lower retrieval rate. The local feature detector is implemented in an offline process. All images from each dataset are calculated interest points. The BTTC is applied to select representative set for each image. Descriptors are then computed for representative points. These descriptors are stored in a database. When a query comes, the same process is implemented on the query image. A matching between query descriptors and the database is done to find a matching list. The matching list is ranked based on similarity values using the Euclidean distance. We only consider the top 5 in the matching list and compute the precision value. With the AAU dataset, we capture another 37 images of AAU buildings and use this set as a query set. In case of the centrum dataset, we consequently use each image from the dataset as a query one. The query image is compared to the rest of the dataset. For comparison, we implement the original MOPS without BTTC filtering. The same matching process is used and we get precision values in top 5 best matches for each query image. In figure 8, we calculate the precision for top 1,2,3,4, and 5 in the matching list. Results show that the proposed approach with much less points not only be able to preserve the retrieval rate, but even improve it further in both experiments.



(a)



(b)

Fig. 8. Precision with the AAU and the Aalborg centrum datasets: comparison between using standard interest points and points after filtering with BTTC.

To evaluate the speed, we store the computing time of the BTTC filtering for both datasets. The BTTC is applied to all 25 images in the image pyramid including 5 different scales and 5 different down-samples. The average time for processing one image pyramid is 0.05 second. This is a small number compared to the time computing descriptors (approximate 1 second for each image pyramid). Moreover, with the reduction of the feature size, the time complexity in the retrieval process achieves 20 and 40% for the AAU and the centrum dataset, respectively. In overall, the adding of BTTC for filtering interest points speeds up the search significant.

IV. Conclusion

Reducing the size of searching space to speed up the search process is an essential factor in many computer vision systems, especially when dealing with a fast growing of image databases and realtime

applications. Different ways to approach the goal can be considered. In this paper, we focus on reducing the size of the feature space by removing non-important features. Our technique is different from existing ones where we consider the representation of image context as the criteria for selecting informative features. We proved that the proposed technique can reduce a significant number of features. Our experiments show that the system can even improve the retrieval results a bit further with informative features. Considering the context of the image in filtering will remove unnecessary features and will be able to present important detail in the image content.

ACKNOWLEDGEMENT

This research is supported by the IPCity project (FP-2004-IST-4-27571), a EU-funded Sixth Framework program Integrated project on Interaction and Presence in Urban Environments.s

REFERENCES

[1] M. Brown, R. Szeliski, and S. Winder. Multi-image matching using multi-scale oriented patches. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 510–517, 2005.

[2] R. Distasi, M. Nappi, and S. Vitulano. Image compression by B-Tree triangular coding. *IEEE Transactions on Communications*, 45(9):1095–1100, 1997.

[3] G. Fritz, C. Seifert, and L. Paletta. Urban object recognition from informative local features. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 132–138, 2005.

[4] C. Harris and M. Stephens. A combined corner and edge detector. *Proceedings of the Alvey Vision Conference*, pages 147–151, 1988.

[5] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 506–513, 2004.

[6] F. Li and J.Kosecka. Probabilistic location recognition using reduced feature set. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3405–3410, 2006.

[7] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[8] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 27(10):1615–1630, 2005.

[9] G.P. Nguyen and H.J. Andersen. Uniqueness filtering for local feature descriptors in urban building recognition. In *Proceedings of the International Conference on Image and Signal Processing*, pages 85–93, 2008.

[10] G.P. Nguyen and H.J. Andersen. Urban building recognition during significant temporal variations. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pages 1–6, 2008.

[11] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2161–2168, 2006.

[12] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *Proceedings of the International Conference on Computer Vision*, pages 1508–1511, 2005.

[13] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008.