Zheng, Z.: Two novel real-time local visual features for omnidirectional vision. Pattern Recognition 43(12), 3938–3949

ARTICLE *in* PATTERN RECOGNITION · DECEMBER 2010

Impact Factor: 3.1 · DOI: 10.1016/j.patcog.2010.06.020 · Source: DBLP

2 AUTHORS, INCLUDING:

Huimin Lu
National University of Defense Technology
23 PUBLICATIONS **83** CITATIONS

SEE PROFILE

Two Novel Real-Time Local Visual Features for Omnidirectional Vision

Huimin Lu*, Zhiqiang Zheng

Department of Automatic Control, College of Mechatronics Engineering and Automation, National University of Defense Technology, Changsha, China

Abstract

Two novel real-time local visual features, namely FAST+LBP and FAST+CSLBP, are proposed in this paper for omnidirectional vision. They combine the advantages of two computationally simple operators by using FAST as the feature detector, and LBP and CS-LBP operators as feature descriptors. The matching experiments of the panoramic images from the COLD database were performed to determine their optimal parameters, and to evaluate and compare their performance with SIFT. The experimental results show that our algorithms perform better, and features can be extracted in real-time. Therefore our local visual features can be applied to those computer/robot vision tasks with high real-time requirements.

Keywords: Local Visual Feature, Omnidirectional vision, FAST, LBP, CS-LBP, Feature Detector, Feature Descriptor

1. Introduction

In comparison with global visual features such as color histogram [1], PCA features [2], Fourier signature features [3], Integral Invariants [4], etc., local visual features have better discriminative power, and are more robust with respect to occlusion. Furthermore, good local visual features can be invariant to image

*Corresponding author. Tel:86-731-84576455

Email addresses: lhmnew@nudt.edu.cn (Huimin Lu), zqzheng@nudt.edu.cn (Zhiqiang Zheng)

rotation, image translation, image scale, changes of view, and even illumination changes. Thus local visual features have become increasingly popular in recent years, and they have been applied very well in many computer/robot vision problems, such as image retrieval [5], image stitching [6], wide baseline matching [7], object recognition [8], place recognition [9], texture recognition [10], robot localization [11], and robot navigation [12]. A local visual feature algorithm consists of a feature detector and a feature descriptor. The former answers the question "where is the feature?", and the later the question "what is the feature?". Many algorithms have been proposed to solve these two problems. With respect to feature detectors, the following have been designed: Harris [13], Susan [14], DOG [8], MSER [15], Salient regions [16], IBR [7], EBR [7], Harris-Laplace [17], Harris-Affine [17], Hessian-Affine [17], Features from Accelerated Segment Test (FAST) [18][19], etc.. With respect to feature descriptors, the following have been used to compute the feature vectors in the feature region: SIFT [8], PCA-SIFT [20], CSIFT [21], SURF [22], GLOH [23], Local Binary Pattern (LBP) [24], Center-Symmetric Local Binary Pattern (CS-LBP) [24], etc.. Many researchers have done a great deal of work on the evaluation and comparison of these algorithms [23][25][26][27], while corresponding source codes/binaries, and many image databases have been released on their websites for further research and evaluation with new algorithms.

Although local visual features have so many advantages, a common deficiency for most of the existing algorithms is that their computation costs are usually high. This deficiency limits the actual application of local visual features, especially in those situations with high real-time requirements, such as robot navigation, self-localization, and object recognition. Therefore several improved versions of the above algorithms have been proposed to accelerate feature detection and/or description. Fast approximated SIFT is presented in Ref. [28]. Compared to the standard SIFT, it uses a box filter to compute the DoM (Difference-of-Mean) images efficiently based on integral image. The key-points can then be detected, and the descriptor is also accelerated by using an integral orientation histogram. The experiments show speed increases by a

factor of eight while the performance is only slightly decreased. In the iterative SIFT [29], the number of features can be defined in advance, so the process of searching the key-points continues iteratively without the need for sequentially going through the whole scale space. When it is applied in the robot localization problem, the computation load of the feature extraction and matching process can be reduced as much as possible while high localization accuracy can be maintained. SURF [22] also takes the advantage of integral images. In feature detector, SURF approximates second order Gaussian derivatives with box filters, and image convolutions with these box filters can be computed rapidly by using integral images. In feature descriptor, Haar wavelet responses, which can also be quickly computed via integral images, are used to construct the descriptor vector. The experiments for camera calibration and object recognition show that SURF outperforms its competitors.

The omnidirectional vision system can provide a 360° view of the robot's surrounding environment in a single image, and the robot can use it to realize object recognition [30], tracking [31] and self-localization [32, 33]. It has become more and more popular as a visual sensor for robots, providing perception information about the environment for robot control and planning. The original algorithms of local visual features should be modified when they are applied to omnidirectional vision because of its special imaging character, especially in determining the feature regions. In Ref. [34], the standard SIFT is simplified, and used for robot localization. The features are detected only in one resolution of the panoramic images without considering scale invariance, and then each feature region is rotated to the same global orientation to ensure rotation invariance. The authors found that the localization performance was improved by omitting the normalization step in constructing the SIFT descriptor. Ref. [35] explains why the rectangular regions are no longer appropriate for omnidirectional vision, and then proposes the active feature regions depending on the positions of key-points. The authors assume that the displacement of the omnidirectional vision is significantly smaller than the depth of the scene, so the feature point's surroundings are seen under the same spatial angle approx-

imately. The optimal feature regions bounded by four conics can be computed according to this character. However, the assumption may not be valid in real application, especially for indoor environments. Ref. [36] discusses the difficulty of matching the local visual features in panoramic images for the varying resolution, and then proposes a multi-angular aperture technique which computes multiple feature regions with different angular apertures. The matching result is improved at the cost of increasing the feature extraction and matching time greatly.

In this paper, we will propose two novel real-time local visual features for omnidirectional vision, so the local visual features can be applied in actual engineering problems with high real-time requirements without a high computation burden. Features from Accelerated Segment Test (FAST) [18][19] will be used as the feature detector, and Local Binary Pattern (LBP) [24] and Center-Symmetric Local Binary Pattern (CS-LBP) [24] as feature descriptors, so two algorithms named FAST+LBP and FAST+CSLBP will be designed. The panoramic images in the COLD database [37] will be used to test our algorithms. The following sections are organized as follows: FAST, LBP and CS-LBP are introduced in section 2; our FAST+LBP and FAST+CSLBP are proposed in section 3; the best parameters of the algorithms are determined by experiments, and the performance of FAST+LBP and FAST+CSLBP is evaluated and compared with the standard SIFT in section 4; section 5 is the conclusion of this paper.

2. FAST, LBP and CS-LBP

In this section, we give a brief introduction of FAST, LBP, and CS-LBP. All three algorithms are computationally simple, so they can be the basis of our real-time local visual features.

2.1. FAST

The corner feature is defined in FAST detector [18][19] by the following Segment-Test algorithm: If more than N contiguous pixels in a Bresenham circle

of radius r around a center pixel p are all brighter than p by some threshold or all darker than p by some threshold, there is a corner feature at p. Then machine learning is utilized to speed up this corner detection process. Every pixel has 16 attributes corresponding to the 16 pixels in the Bresenham circle (for $r = 3$), and each attribute can be 0, 1 or -1. If a pixel with position x on the circle of p is brighter(darker) than p, the corresponding attribute is 1(-1). Otherwise, the attribute is 0. A decision tree can be learned by using ID3 to select the pixels in the circle which yield the most information about whether the center pixel is a corner. Therefore a pixel can be classified as a corner feature or not more efficiently, which means the Segment-Test algorithm is accelerated. This decision tree is then converted into C-code, creating a long string of nested if-then-else statements which is compiled and used as a corner detector. Finally non-maximal suppression is applied to remove corners which have an adjacent corner with higher value of the sum of the absolute difference between the pixels in the circle and the center pixel.

FAST algorithms are named according to different values of N. Thus for N values of 9, 10, 11, and 12, the corresponding algorithms are FAST 9, FAST 10, FAST 11, and FAST 12. According to the experiments in [19], FAST 9 seems to be the best FAST detector, and it is over five times faster than the quickest non-FAST detector. The FAST algorithm also significantly outperforms Harris, DoG, Harris-Laplace, SUSAN, etc. in repeatability, except in cases with large amounts of added image noise.

2.2. LBP operator

The LBP is firstly proposed as a texture operator [38], and it has been highly successful for various computer vision problems such as texture classification [39], face recognition [40], background subtraction [41], and recognition of 3D textured surfaces [42].

The LBP is a powerful illumination invariant texture primitive. The operator describes each pixel by the relative gray values of its neighboring pixels. An example with eight neighbors is shown in Fig. 1. If the gray value of the

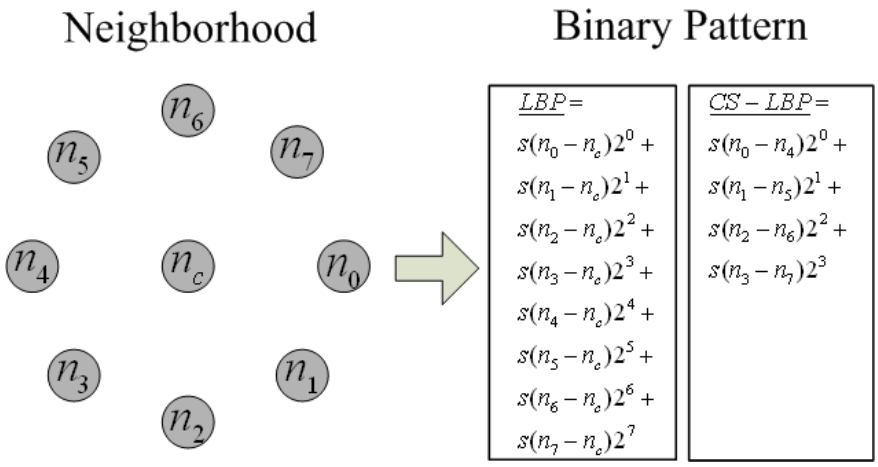


Figure 1: The LBP and CS-LBP for a neighborhood of eight pixels. This figure is from Ref. [24].

neighboring pixel is higher or equal to that of the center pixel, the binary value is set to be one. Otherwise it is set to be zero. The LBP value of a center pixel in (x, y) position is computed over the neighborhood as follows:

$$LBP_{R,N}(x,y) = \sum_{i=0}^{N-1} s(n_i - n_c)2^i, \quad s(t) = \begin{cases} 1, & t \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where n_c is the gray value of the center pixel, and n_i the gray value of N equally spaced pixels on a circle of radius R. According to Eq.(1), the $LBP_{R,N}$ value may be any integer between 0 and $2^N - 1$. The histogram of the $LBP_{R,N}$ values computed over an image region (the histogram dimension will be 2^N) can be used for texture description, and it has been proven to be robust against illumination changes. It is also very fast to compute, and do not require many parameters to be set [38].

Several modified versions of LBP operator have been described in Ref. [39] for achieving rotation invariance and reducing the histogram dimension of the LBP. When the image is rotated, the gray value n_i will correspondingly move along the perimeter of the circle, so different $LBP_{R,N}$ may be computed. To remove the effect of rotation, the first modified version with rotation invariance

is defined as follows:

$$LBP^{ri}_{R,N}(x,y) = min\{ROR(LBP_{R,N}, i) \quad | \quad i = 0, 1, ..., N-1\} \tag{2}$$

where $ROR(LBP_{R,N}, i)$ performs a circular bit-wise right shift on the N-bit number $LBP_{R,N}$ i times. $LBP^{ri}_{R,N}$ can have 36 different values when $N = 8$, and the histogram dimension of $LBP^{ri}_{R,N}$ over an image region is 36.

In the second version named *uniform* LBP, at most two one-to-zero or zero-to-one transitions in the circular binary code are allowed, so whether an LBP is uniform can be judged by the following definition:

$$U(LBP_{R,N}) = |s(n_{N-1} - n_c) - s(n_0 - n_c)| + \sum_{i=1}^{N-1} |s(n_i - n_c) - s(n_{i-1} - n_c)| \tag{3}$$

If $U(LBP_{R,N}) \leq 2$, the LBP is uniform. The *uniform* LBP, expressed as $LBP^{u2}_{R,N}$, can have $N(N-1)+2$ different values, so the histogram dimension of $LBP^{u2}_{R,N}$ over an image region is $N(N-1)+2+1$ (the final 1 corresponds to those *non-uniform* LBP).

The third version is the *uniform* LBP with rotation invariance which combines the above two modifications. Therefore $LBP^{riu2}_{R,N}$ value is computed as follows:

$$LBP^{riu2}_{R,N}(x,y) = \begin{cases} \sum_{i=0}^{N-1} s(n_i - n_c), & U(LBP_{R,N}) \leq 2 \\ N+1, & \text{otherwise} \end{cases} \tag{4}$$

$LBP^{riu2}_{R,N}$ value can have $N+1+1$ different values, so the histogram dimension of $LBP^{riu2}_{R,N}$ over an image region is $N+1+1$.

All three modified LBP versions can be considered to be a mapping from the original LBP with high value range to the corresponding modified LBP with low value range. Thus the histogram dimension can be reduced to varying extents. In practice, the mapping process is implemented by a look-up table which can be created in advance according to the different mapping mode: ri, u2, or riu2.

2.3. CS-LBP operator

Instead of comparing each neighboring pixel with the center pixel, the CS-LBP [24] compares the center-symmetric pairs of pixels, as shown in Fig. 1.

This halves the number of comparisons for the same number of neighbors - N. The CS-LBP value of a center pixel in (x, y) position is computed as follows:

$$CS-LBP_{R,N,T}(x,y) = \sum_{i=0}^{(N/2)-1} s(n_i - n_{i+(N/2)})2^i, \quad s(t) = \begin{cases} 1, & t > T \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where n_i and $n_{i+(N/2)}$ are the gray values of center-symmetric pairs of pixels of N equally spaced pixels on a circle with radius R, and the threshold T is a small value.

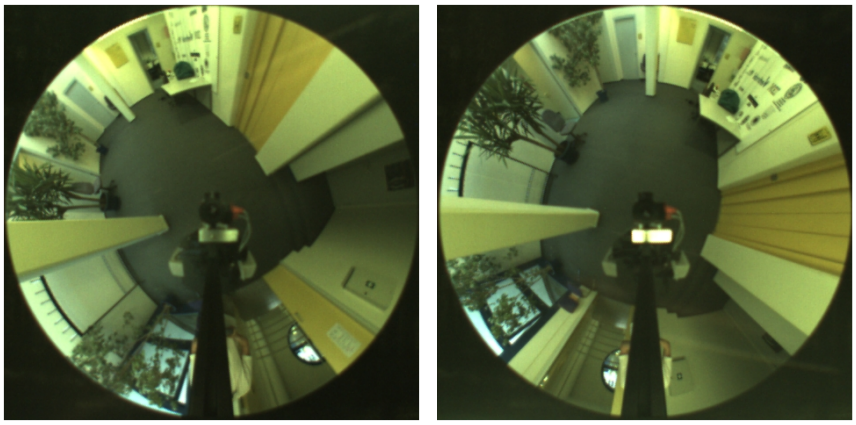
$CS-LBP_{R,N,T}$ value can have $2^{N/2}$ different values, so the histogram dimension of $CS-LBP_{R,N,T}$ over an image region is $2^{N/2}$. Compared to the original LBP, the histogram dimension of the CS-LBP is greatly reduced.

3. Our Novel Real-Time Local Visual Features

In this section, we present our two novel real-time local visual features, namely FAST+LBP and FAST+CSLBP, for omnidirectional vision in detail. The algorithms are divided into three steps: feature detector, feature region determination, and feature descriptor. Both of the feature detectors are FAST, and the feature region determining methods are the same for both. The LBP and CS-LBP operator will be used as the feature descriptor in the two algorithms respectively.

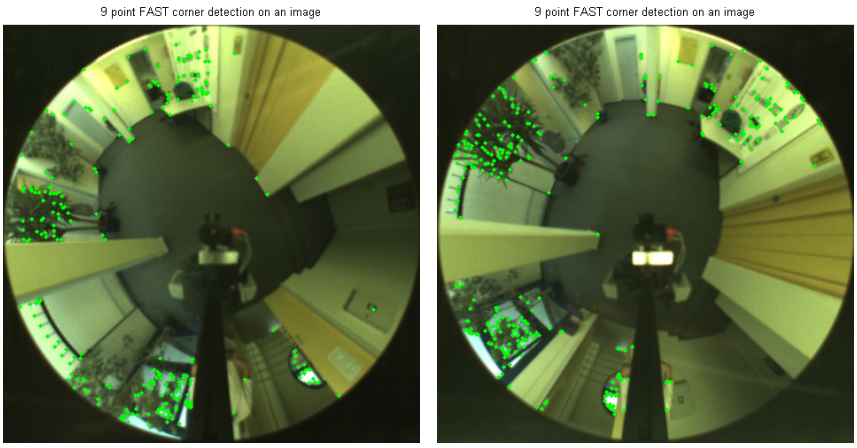
3.1. FAST feature detector

Because the FAST 9 algorithm has a low computation cost and excellent performance in repeatability, it was chosen as the feature detector for our real-time local visual features. The typical panoramic images and the corner features detected by FAST 9 are demonstrated in Fig. 2 and Fig. 3 respectively. The images in Fig. 2 are from the COLD database [37], and the database will be used in all of the experiments described in this paper. The two images are acquired by the robot's omnidirectional vision in two different positions. The robot's translation between these two positions is 0.7561 m, and the rotation is 0.9053 rad.



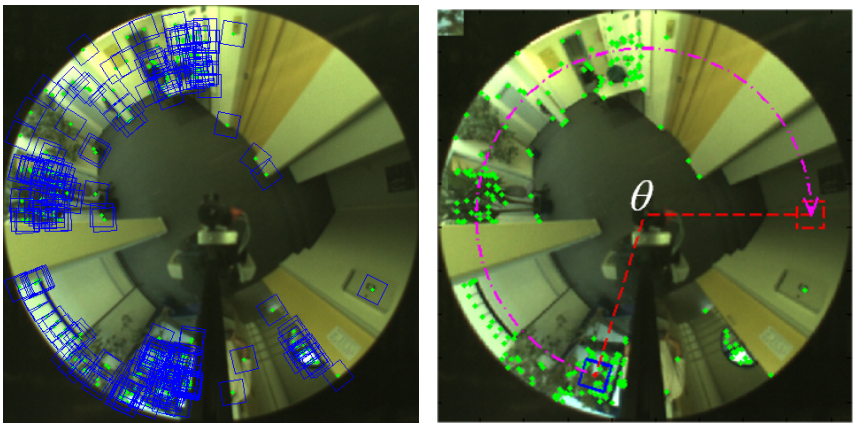
(a) (b)

Figure 2: The typical panoramic images from the COLD database. (a) and (b) are acquired by the robot's omnidirectional vision in two different positions. The robot's translation is 0.7561 m, and the rotation is 0.9053 rad.



(a) (b)

Figure 3: The feature detecting results of the panoramic images in Fig. 2 by FAST 9. The green points are the detected corner features.



(a) (b)

Figure 4: (a) The blue rectangles are the feature regions for the panoramic image in Fig. 2(a). (b) A feature region is rotated by angle θ to a fixed orientation. The small region on the top left of the image is the rotated feature region.

3.2. Feature region determination

After a corner feature has been detected, a surrounding image region should be determined, and then a descriptor can be extracted from the image region. Some affine invariant feature detectors [17] have been proposed to adapt the feature region to affine transformations by iterative algorithms. Although they provide better performance, the computation complexity increases significantly [17]. Therefore we do not consider affine invariance for our real-time local visual feature algorithms. We adopt the feature region determining method proposed in Ref. [34] to achieve rotation invariance. Rectangular image regions surrounding corner features are firstly determined in the radial direction, and then rotated to a fixed orientation, as shown in Fig. 4. Fig. 4(a) shows the determined feature regions for the panoramic image in Fig. 2(a), and Fig. 4(b) shows how a feature region is rotated to the fixed orientation. During the rotation process, bilinear interpolation is used.

In the next section, we will compare this feature region determining method with the one which determines the feature regions directly in horizontal and vertical directions through experimentation. The image size of each feature region is also an important parameter, and the best size will be determined by

experiments in the next section.

3.3. Feature descriptor with LBP and CS-LBP

The final step of the local visual feature algorithm is to describe the features by computing vectors according to the information of feature regions. Recently, the LBP and CS-LBP have been used as feature descriptors in Ref. [24], and the strength of the SIFT descriptor is also combined. The SIFT-like grid is used, but SIFT gradient features are replaced by LBP-based features and CS-LBP-based features. The experimental results in Ref. [24] show that the proposed LBP descriptor and CS-LBP descriptor outperform the SIFT descriptor. In this paper, we use the same approach to extract descriptors for the detected features by FAST in section 3.1 and 3.2.

3.3.1. Feature descriptor with LBP

An LBP value for each pixel of the feature region can be computed according to the introduction in section 2.2. In order to incorporate spatial information into the descriptor, the feature region can be divided into different grids such as 1×1 (1 cell), 2×2 (4 cells), 3×3 (9 cells), and 4×4 (16 cells), as shown in Fig. 5. For each cell, the histogram of LBP values is created, and then all the histograms are concatenated into a vector as the descriptor. Finally, the descriptor is normalized to unit length. The descriptor dimension is $M \times M \times$*the histogram dimension* for $M \times M$ cells. Therefore, the resulting descriptor is a 3D histogram of LBP feature locations and LBP values. In computing the histogram, the LBP values can be weighted with a Gaussian window overlaid over the whole feature region, or with uniform weights over the whole region. The latter means that the feature weighting is omitted.

The performance and dimension of the LBP descriptor will be affected greatly by different algorithm parameters such as the number of cells, different R and N, Gaussian or uniform weighting, the LBP mode including the original LBP, LBP^{ri}, LBP^{u2}, and LBP^{riu2} as introduced in section 2.2. The best parameters will be determined by experiments in the next section.

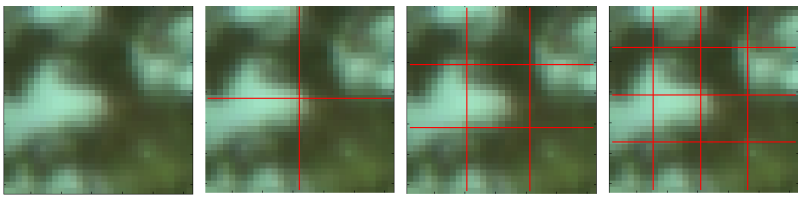


Figure 5: The different grids that the feature region can be divided into. From left to right: 1×1 cell, 2×2 cells, 3×3 cells, 4×4 cells.

3.3.2. Feature descriptor with CS-LBP

A CS-LBP value for each pixel of the feature region can be computed according to the introduction in section 2.3. The histogram of CS-LBP values is created to construct the CS-LBP descriptor in the same way as that presented in section 3.3.1. The performance and dimension of the CS-LBP descriptor will also be greatly affected by different algorithm parameters such as the number of cells, different R and N, different threshold T, Gaussian or uniform weighting. The best parameters will also be determined by experiments in the next section.

4. Experimental Evaluation and Discussion

In this section, a series of experiments will be done to test our two local visual feature algorithms. Firstly, we will introduce the experimental setup such as image database, the feature matching criterion and the criterion for performance evaluation. Then the best parameters for FAST+LBP and FAST+CSLBP will be determined by experiments. After the best parameters have been determined, the performance and the needed computation time of our algorithms will be compared with SIFT. Finally the discussions will be presented according to the experimental results.

4.1. Experimental setup

COLD [37] is a freely available database which provides a large-scale, flexible testing environment for vision-based topological localization. COLD contains 76 image sequences acquired in three different indoor environments across Europe. The images are acquired by the same perspective and omnidirectional vision in

different rooms and under various lighting conditions. We will use the typical panoramic images and image series to perform our experiments.

When local visual features are applied in robot localization, robot SLAM, etc., the features should be matched between the image pairs acquired in different imaging conditions, such as different robot positions and various lighting conditions. Therefore we evaluate the performance of local visual features according to the feature matching results. For each feature descriptor in an image, we compute its Euclidean distances with all the feature descriptors in another image needing to be matched. We consider that a match is found between the feature pair with the closest distance if the ratio of the closest to second closest distance is smaller than threshold T_{ratio} [8] as follows:

$$ratio = \frac{the\ closest\ distance}{the\ second\ closest\ distance} \leq T_{ratio} \tag{6}$$

The FAST detector was compared with several well known detectors in Ref. [19], and the LBP and CS-LBP descriptors were also compared with SIFT in Ref. [24], so their performances have been tested independently. In this paper, we will evaluate the overall performance of local visual features as a whole, but not evaluate the detector and descriptor independently as in Ref. [17][19][23][24]. Therefore we use $matching\ score\ versus\ 1-precision$ as the criterion for performance evaluation, instead of $recall\ versus\ 1-precision$ which is used to evaluate the descriptor's performance in Ref. [23][24]. We define *matching score* in the same way as Ref. [25]:

$$matching\ score = \frac{the\ number\ of\ correct\ matches}{the\ smaller\ number\ of\ features\ in\ the\ pair\ of\ images} \tag{7}$$

We define $1-precision$ as follows in the same way as Ref. [23][24]:

$$1-precision = \frac{the\ number\ of\ false\ matches}{the\ number\ of\ correct\ matches + the\ number\ of\ false\ matches} \tag{8}$$

After the feature matching is finished, an 18 bin histogram is created from $\triangle\theta_i = normalize(\theta_i - \theta_i^{'})$ using all the matched features, where θ_i and $\theta_i^{'}$ are the rotated angles of the ith pair of matched features relative to the fixed orien-

tation in section 3.2, and $normalize(.)$ means normalizing an angle to $[0, 2\pi)$. According to the character of omnidirectional vision, when the robot is just rotated or the robot's translation is small comparing to the depth of the scene, the relative angle of each pair of correctly matched features, namely φ, should be almost the same, so it can be estimated by computing the mean value of those $\triangle\theta_i$ falling into the highest bin, and φ is approximately the rotation angle of the robot. If $|\triangle\theta_i - \varphi| < T_{angle}$, where T_{angle} is the threshold determined by experiments, the match related to $\triangle\theta_i$ is a correct match. Otherwise, it is a false match.

As we change the threshold T_{ratio}, the curve of *matching score versus* $1 -$ *precision* can be acquired to evaluate the performance of the algorithms.

4.2. Parameter evaluation for FAST+LBP

The evaluation of different parameter settings for FAST+LBP is carried out in this experiment to determine the best parameters. As presented in section 3, six parameters will affect the performance of FAST+LBP. We will test their different settings as follows:

The size of the feature region: 15×15, 19×19, 23×23, 27×27, 31×31, 35×35, 39×39, 43×43 pixels;

The feature region determining method: method 1–determining the feature's rectangular region directly in horizontal and vertical directions, method 2–determining the rectangular region in the radial direction and then rotating it to the fixed orientation as proposed in section 3.2;

The number of grids: 1×1 cell, 2×2 cells, 3×3 cells, 4×4 cells;

The N and R: $N = 8$ and $R = 1$, $N = 16$ and $R = 2$, $N = 24$ and $R = 3$;

The LBP mode: the original LBP, LBP^{ri}, LBP^{u2}, LBP^{riu2};

The weighting strategy: Gaussian weighting, uniform weighting.

Because of a huge amount of different combinations of the above parameters, only one parameter is varied at a time while the others are kept fixed. The pair of images in Fig. 2 are used to perform the feature matching, and the curves of *matching score versus* $1 -$ *precision* with different parameters are shown in

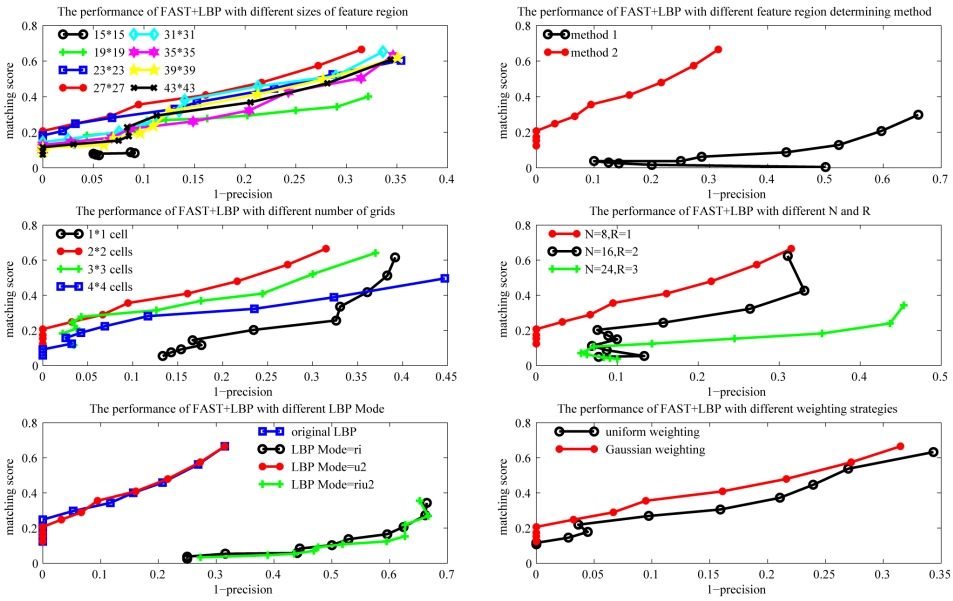


Figure 6: Parameter evaluation results for FAST+LBP. Only one parameter is varied at a time while the others are kept fixed with the best parameters.

Fig. 6. The red curves in Fig. 6 represent the performance achieved by using the best parameters. From the matching results, we see that 27×27 pixels for the feature region, region determining method 2, 2×2 cells, $N = 8$, $R = 1$, LBP^{u2}, and Gaussian weighting provide the best performance for FAST+LBP. The descriptor dimension of our final FAST+LBP is $2\times2\times(8\times7+2+1)$=236, as shown in Fig. 7.

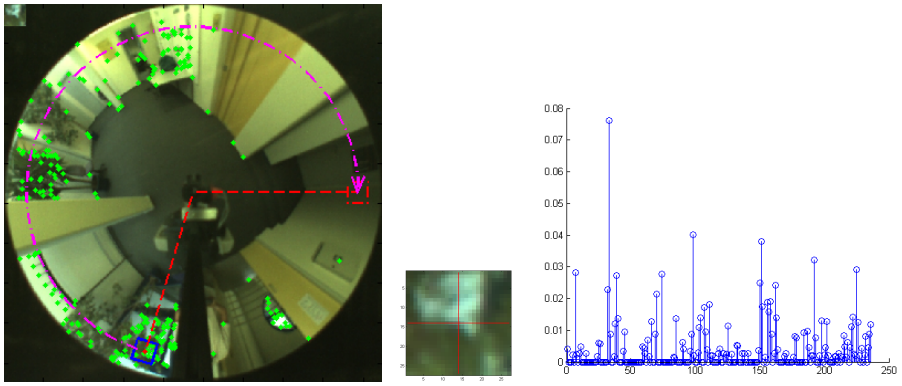
4.3. Parameter evaluation for FAST+CSLBP

The evaluation of different parameter settings for FAST+CSLBP is carried out in this experiment to determine the best parameters. As presented in section 3, there are also six parameters affecting the performance of FAST+CSLBP. We will test their different settings as follows:

The size of the feature region: 23×23, 27×27, 31×31, 35×35, 39×39, 43×43, 47×47, 51×51 pixels;

The feature region determining method: the same as those in FAST+LBP;

The number of grids: the same as those in FAST+LBP;



(a) (b) (c)

Figure 7: Our final FAST+LBP algorithm. (a) A feature region on the panoramic image. (b) The scale-up feature region. The region is divided into 2×2 cells. (c) The resulting feature descriptor.

The N and R: $N = 8$ and $R = 2$, $N = 6$ and $R = 2$;

The T: $T = 0$, $T = 5$, $T = 10$;

The weighting strategy: the same as those in FAST+LBP.

We perform the feature matching in the same way as FAST+LBP, and the same pair of images are used. The curves of *matching score versus* $1-$*precision* with different parameters are shown in Fig. 8. The red curves in Fig. 8 represent the performance achieved by using the best parameters. From the matching results, we see that 43×43 pixels for the feature region, region determining method 2, 3×3 cells, $N = 6$, $R = 2$, $T = 5$, and Gaussian weighting provide the best performance for FAST+CSLBP. The descriptor dimension of our final FAST+CSLBP is $3\times3\times2^{6/2}$=72, much smaller than that of our final FAST+LBP, as shown in Fig. 9.

4.4. Performance comparison of FAST+LBP, FAST+CSLBP, and SIFT

The performance comparison of FAST+LBP, FAST+CSLBP and SIFT is carried out in this experiment. The SIFT we adopt is implemented by Andrea Vedaldi [43]. The criterion of *matching score versus* $1-$*precision* is still used. Because most of the current robot's cameras are color ones, and the images in the COLD database are color images, we also compare the color version of

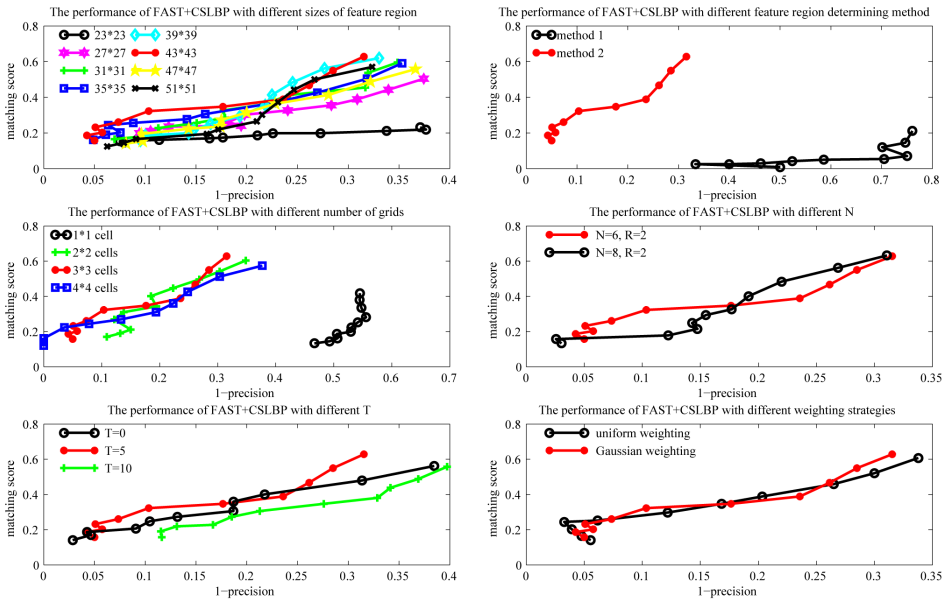
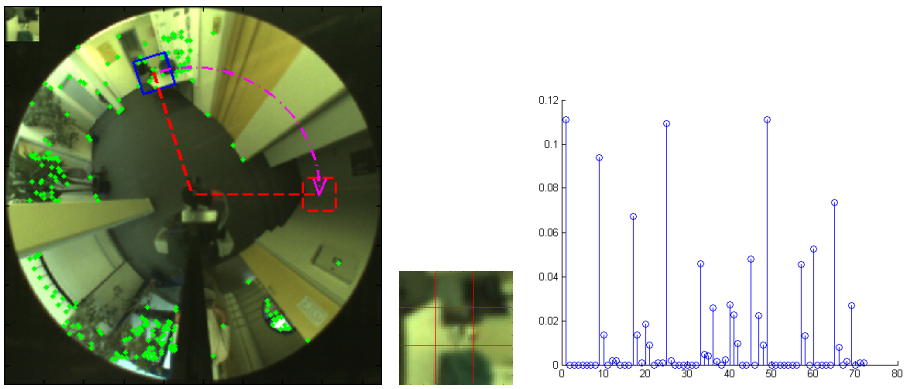


Figure 8: Parameter evaluation results for FAST+CSLBP. Only one parameter is varied at a time while the others are kept fixed with the best parameters.



(a) (b) (c)

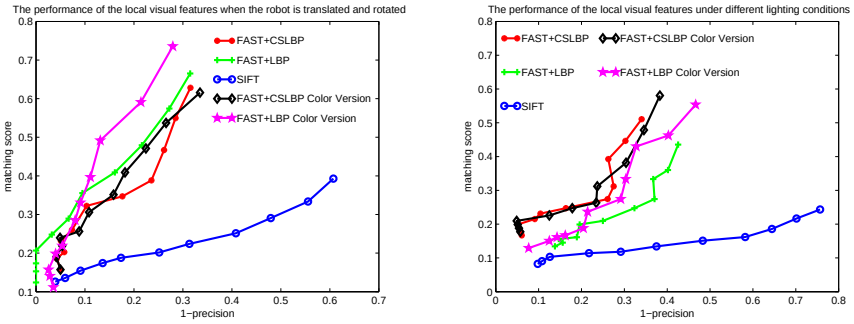
Figure 9: Our final FAST+CSLBP algorithm. (a) A feature region on the panoramic image. (b) The scale-up feature region. The region is divided into 3×3 cells. (c) The resulting feature descriptor.

(a) (b)

Figure 10: The typical panoramic images from the COLD database acquired by the robot's omnidirectional vision in the same position but under different lighting conditions. (a) At night. (b) In daytime and cloudy weather.

FAST+LBP and FAST+CSLBP together. In our color version of FAST+LBP and FAST+CSLBP, the feature detector still uses the gray values of images, but the descriptor is computed in all of the R, G, B color channels, so its dimension is three times of that of the gray version. Two pairs of images are used. The first one is that in Fig. 2, and they are acquired when the robot is translated and rotated. The second pair of images are acquired when the robot is in the same position but under different lighting conditions, as shown in Fig. 10. The matching results of these two pairs of images are depicted in Fig. 11(a) and (b) respectively.

We fix the threshold T_{ratio} as 0.95 after making a compromise between *matching score* and *precision*. The matching results of the pair of images in Fig. 2 by FAST+LBP and FAST+CSLBP with this threshold are shown in Fig. 12. Then we can evaluate how *matching score* changes with the different imaging conditions of omnidirectional vision caused by the robot's translation, rotation, and different lighting conditions. Three image series are used in this evaluation. The first one includes 30 images, and they are acquired as the robot is only translated. The translation increases with the image number, and the maximal translation is 1.7975 m. The second one includes 17 images, and they



(a) (b)

Figure 11: The performance comparison of FAST+LBP, FAST+CSLBP, the color version of FAST+LBP, the color version of FAST+CSLBP, and SIFT. (a) The robot is translated and rotated. (b) Under different lighting conditions.

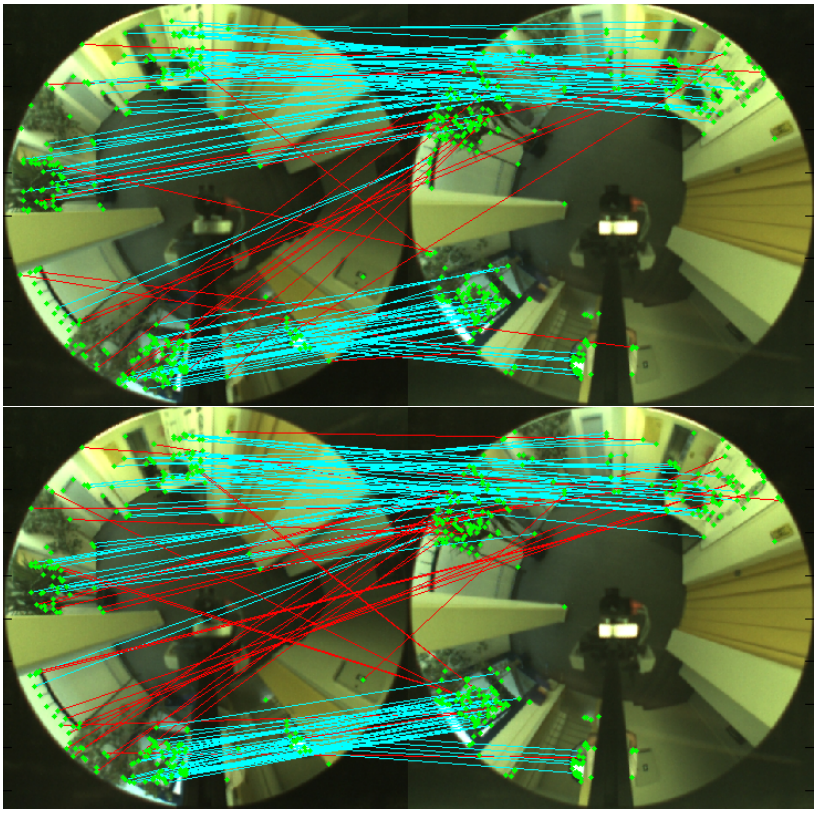


Figure 12: The matching results of the pair of images in Fig. 2 by FAST+LBP (top) and FAST+CSLBP (bottom). The green points are the detected corner features. The cyan lines represent the correct matches, and the red lines represent the false matches.

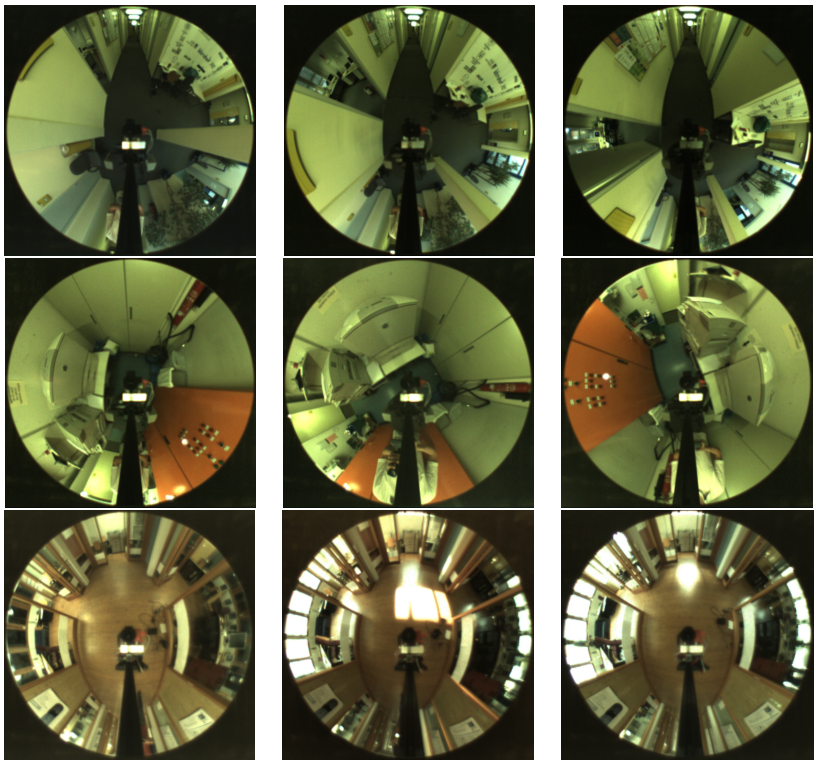
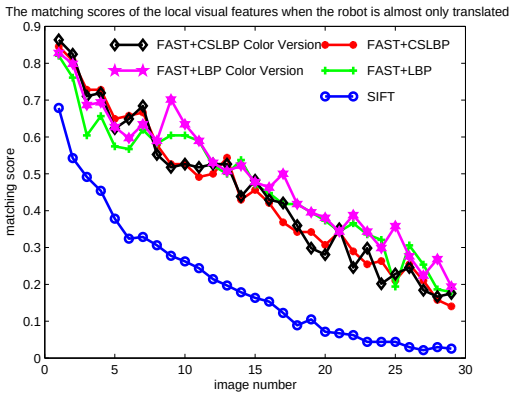


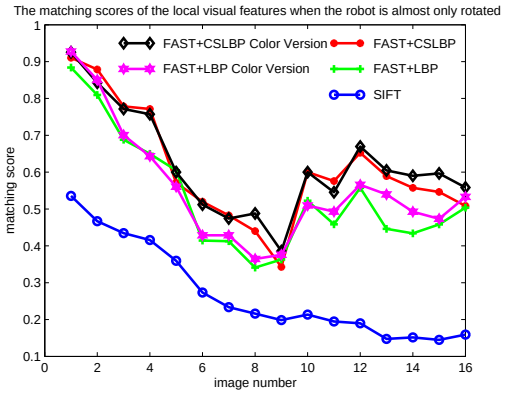
Figure 13: The typical images belonging to different series. (top) The first series. (middle) The second series. (bottom) The third series.

are acquired as the robot is only rotated. The rotation increases with the image number, and the maximal rotation is π. The third one includes 5 images, and they are acquired in the same position and under different lighting conditions. Some typical images belonging to each series are shown in Fig. 13. We perform the feature matching between the first image and all the other images in each series, so how *matching score* changes with different imaging conditions is acquired, as shown in Fig. 14.

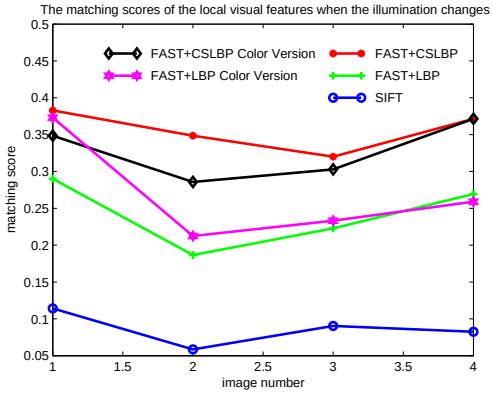
From the above experimental results, we clearly see that FAST+LBP and FAST+CSLBP provide better performance than SIFT in image matching, and they are excellent local visual features for omnidirectional vision. The matching results are not bad even when the robot is translated and rotated greatly and the lighting conditions are very different. The color version seems a little



(a)



(b)



(c)

Figure 14: The *matching score* with different imaging conditions by FAST+LBP, FAST+CSLBP, the color version of FAST+LBP, the color version of FAST+CSLBP, and SIFT. (a) The robot is only translated. (b) The robot is only rotated. (c) Under different lighting conditions.

better than the gray version. However, its computation cost is much higher, because the descriptor of the color version is computed in each of the three color channels. Furthermore, it takes much more time to match features for the color version because of the larger descriptor dimension. So we prefer the gray version rather than the color version. Regarding the comparison of FAST+LBP and FAST+CSLBP, several conclusions can be summarized as follows: FAST+LBP seems better than FAST+CSLBP when the robot is translated and rotated, as shown in Fig. 11(a) and Fig. 14(a); FAST+CSLBP seems better than FAST+LBP when the robot is only rotated, as shown in Fig. 14(b); FAST+CSLBP seems better than FAST+LBP when the illumination changes, as shown in Fig. 11(b) and Fig. 14(c); the descriptor dimension of our final FAST+CSLBP is much smaller than that of our final FAST+LBP, which is also an important factor that should be considered when choosing local visual feature in actual applications.

4.5. Comparison of the needed computation time

In this experiment, we collect 125 panoramic images from the COLD database, and then extract local visual features from these images using FAST+LBP, FAST+CSLBP, and SIFT respectively. Our FAST+LBP and FAST+CSLBP are implemented by C++, and the SIFT we use is implemented by C++ and Matlab using C-Mex technique [43]. The computer is equipped with 2.26GHz Duo CPU and 1.0G memory. The number of features, the time needed to extract all the features in an image, and the average time needed to extract one feature are demonstrated in Fig. 15. The time needed in the three steps of FAST+LBP is also shown (the result of FAST+CSLBP is almost the same, so we do not demonstrate it in this figure). We see that our FAST+LBP and FAST+CSLBP extract about 150∼350 features on an image, less than SIFT. Actually, according to the researches in Ref. [29][44], the large number of local visual features is beyond what robot localization or image retrieval really needs, and the number can be reduced greatly. So the number of the FAST+LBP and FAST+CSLBP features is enough for the applications, which has also been verified by the above

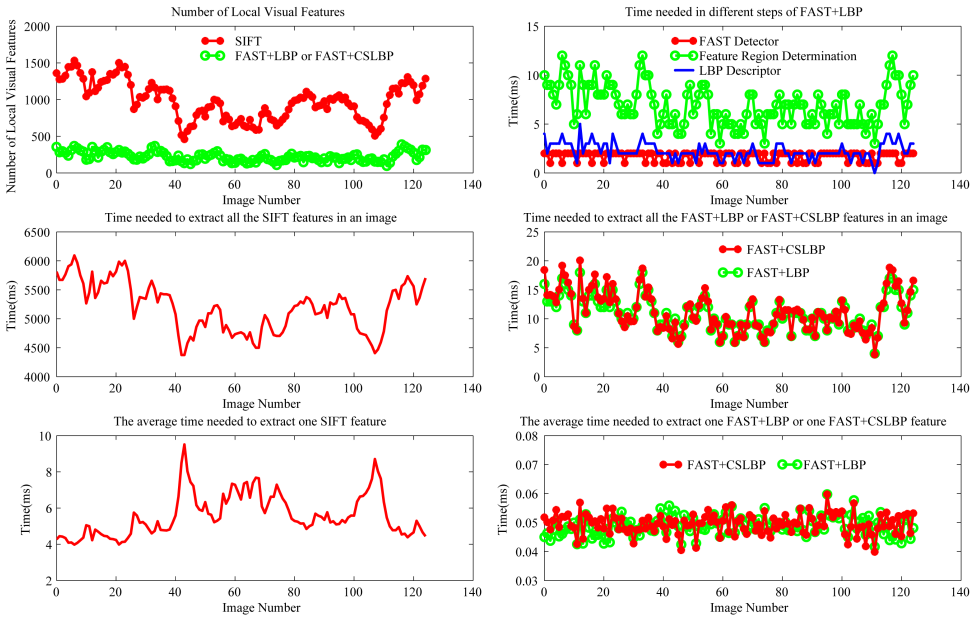


Figure 15: The comparison of the needed computation time by FAST+LBP, FAST+CSLBP and SIFT.

image matching experiments. Our FAST+LBP and FAST+CSLBP can be performed much faster, and the computation times needed in FAST+LBP and FAST+CSLBP are almost the same. After doing statistics on the computation time, we find that the time needed to extract all the features by SIFT in an image is about 508 times that of FAST+LBP or FAST+CSLBP; the average time needed to extract one feature by SIFT is about 115 times that of FAST+LBP or FAST+CSLBP. The computation time needed to extract all the features in an image by FAST+LBP or FAST+CSLBP is from 5ms to 20ms, so they can be performed in real-time.

4.6. Discussions

Our FAST+LBP and FAST+CSLBP have the following good features:

- They are computationally simple, and can be used in the actual robot localization, visual SLAM, etc. with real-time requirement;

- Better matching results can be achieved compared to SIFT, which means that they have better discriminative power;

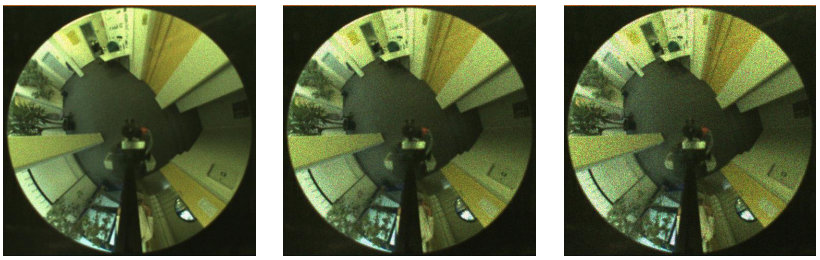


Figure 16: The typical images when multiplicative noise with different variances is added. The variances are 0.01 (left), 0.03 (middle), and 0.05 (right) respectively.

- They are robust with respect to rotation, different lighting conditions, and the robot's certain translation;

- They can also be used in perspective cameras, besides omnidirectional vision.

Because the FAST detector is sensitive to image noise [19], we also compare the performances of FAST+LBP, FAST+CSLBP, the color version of FAST+LBP, the color version of FAST+CSLBP, and SIFT when different image noise is added. We add uniformly distributed random noise with mean 0 and different variances to the panoramic image in Fig. 2(a). The noise is multiplicative, and the range of the variance is from 0 to 0.05. Several noisy images are shown in Fig. 16. We perform the feature matching between the original image and the noisy images to see how the noise affects the performance of different local visual features. The experimental results are shown in Fig. 17. We see that although large amounts of image noise have been added to the image and the FAST detector is sensitive to the image noise, good performance of FAST+CSLBP, comparable with SIFT, can be achieved. The performance of FAST+CSLBP is much better than that of FAST+LBP in this experiment, which also means that the CS-LBP descriptor is much more robust to image noise than the LBP descriptor. There is not much difference in the robustness to image noise between the color versions and the gray versions of our local visual features.

In the next work, we will try to improve the robustness of the FAST detector

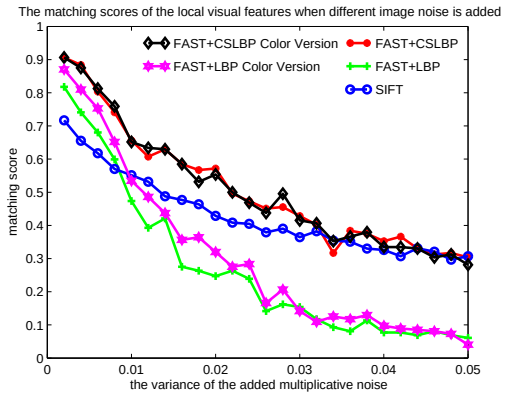


Figure 17: The performance comparison of the local visual features when different image noise is added.

to image noise. We will perform more experiments to evaluate the performance of our FAST+LBP and FAST+CSLBP, and compare them with more local visual features, besides SIFT. We will also try to apply our real-time local visual features to the actual robot topological localization, visual SLAM, and scene/place classification or recognition.

5. Conclusions

Two novel local visual features, namely FAST+LBP and FAST+CSLBP, are proposed for omnidirectional vision in this paper. They combine the advantages of two computationally simple operators by using FAST as the feature detector, and LBP and CS-LBP operators as feature descriptors. The best parameters of the algorithms were determined by experiments. The comparisons between FAST+LBP, FAST+CSLBP, the color version of FAST+LBP, the color version of FAST+CSLBP, and SIFT were performed, and the experimental results show that our algorithms have better performance than SIFT, and features can be extracted in real-time. Furthermore, several conclusions on the comparison of FAST+LBP and FAST+CSLBP are also summarized from the experimental results.

Acknowledgement

We would like to thank Edward Rosten and Tom Drummond for their release of the FAST source code, Marko Heikkil*ä* and Timo Ahonen for their release of the LBP source code, Andrea Vedaldi for his release of the SIFT source code, and Andrzej Pronobis, Barbara Caputo, et al. for providing their COLD database. Without these wonderful open resources, we could not have implemented and evaluated our local visual feature algorithms so conveniently and quickly. We would like to thank the anonymous reviewers for their valuable comments.

References

[1] I. Ulrich, I. Nourbakhsh, Appearance-based place recognition for topological localization, in: 2000 IEEE International Conference on Robotics and Automation, 2000, pp. 1023-1029.

[2] B. J. A. Kröse, N. Vlassis, R. Bunschoten, and Y. Motomura, A probabilistic model for appearance-based robot localization, Image and Vision Computing 19(6)(2001), 381-391.

[3] E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro, Image-based Monte Carlo localisation with omnidirectional images, Robotics and Autonomous Systems, 48(1)(2004), 17-30.

[4] J. Wolf, W. Burgard, H. Burkhardt, Robust vision-based localization by combining an image-retrieval system with Monte Carlo localization, IEEE Transactions on Robotics, 21(2)(2005), 208-216.

[5] K. Mikolajczyk, C. Schmid, Indexing based on scale invariant interest points, in: 8th IEEE International Conference on Computer Vision, vol.1, 2001, pp. 525-531.

[6] M. Brown and D. G. Lowe, Automatic Panoramic Image Stitching using Invariant Features, Int. J. Comput. Vision 74(1)(2007), 59-73.

[7] T. Tuytelaars, L. V. Gool, Matching widely separated views based on affine invariant regions, Int. J. Comput. Vision 59(1)(2004), 61-85.

[8] D. G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vision 60(2)(2004), 91-110.

[9] M. M. Ullah, A. Pronobis, B. Caputo, J. Luo, P. Jensfelt, H. I. Christensen, Towards robust place recognition for robot localization, in: 2008 IEEE International Conference on Robotics and Automation, 2008, pp. 530-537.

[10] S. Lazebnik, C. Schmid, J. Ponce, A sparse texture representation using local affine regions, IEEE Trans. Pattern Anal. Mach. Intell. 27(8)(2005), 1265-1278.

[11] S. Se, D. G. Lowe, J. Little, Global localization using distinctive visual features, in: IEEE/RSJ International Conference on Intelligent Robots and System, vol.1, 2002, pp. 226-231.

[12] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. V. Gool, Omnidirectional Vision Based Topological Navigation, Int. J. Comput. Vision 74(3)(2007), 219-236.

[13] C. Harris, M. Stephens, A combined corner and edge detector, Alvey Vision Conference, 1988, pp. 147-151.

[14] S.M. Smith, J.M. Brady, SUSAN-a new approach to low level image processing, Int. J. Comput. Vision 23 (1) (1997), 45-78.

[15] J. Matas, O. Chum, M. Urban, T. Pajdla, Robust wide-baseline stereo from maximally stable extremal regions, in: Proceedings of the British Machine Vision Conference, 2002, pp. 384-393.

[16] T. Kadir, A. Zisserman, M. Brady, An affine invariant salient region detector, in Proceedings of the European Conference on Computer Vision, Lecture Notes in Computer Science, vol. 3021, 2004, pp. 228-241.

[17] K. Mikolajczyk, C. Schmid, Scale & affine invariant interest point detectors, Int. J. Comput. Vision 60 (1) (2004) 63-86.

[18] E. Rosten, T. Drummond, Fusing points and lines for high performance tracking, in: IEEE International Conference on Computer Vision, 2005, pp. 1508-1515.

[19] E. Rosten, T. Drummond, Machine learning for high-speed corner detection, in: European Conference on Computer Vision, 2006, pp. 430-443.

[20] Y. Ke, R. Sukthankar, PCA-SIFT: A more distinctive representation for local image descriptors, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, vol. 2, 2004, pp. 506-513.

[21] A.E. Abdel-Hakim, A.A. Farag, CSIFT: a sift descriptor with color invariant characteristics, in: Proceedings of the Computer Vision and Pattern Recognition, 2006, pp. 1978-1983.

[22] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, Speeded-Up Robust Features (SURF), Computer Vision and Image Understanding 110 (3) (2008), 346-359.

[23] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE Trans. Pattern Anal. Mach. Intell. 27 (10) (2005), 1615-1630.

[24] M. Heikkil*ä*, M. Pietik*ä*inen, and C. Schmid, Description of interest regions with local binary patterns, Pattern Recognition 42 (3) (2009), 425-436.

[25] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, L. V. Gool, A comparison of affine region detectors, Int. J. Comput. Vision 65 (1) (2005) 43-72.

[26] C. Schmid, R. Mohr, and C. Bauckhage, Evaluation of interest point detectors, Int. J. Comput. Vision 37(2) (2000), 151-172.

[27] J. Li, and N. M. Allinson, A comprehensive review of current local features for computer vision, Neurocomputing 71 (2008), 1771-1787.

[28] M. Grabner, H. Grabner, and H. Bischof, Fast approximated SIFT, in: 7th Asian Conference on Computer Vision, 2006, pp. 918-927.

[29] H. Tamimi, H. Andreasson, A. Treptow, T. Duckett, A. Zell, Localization of Mobile Robots with Omnidirectional Vision using Particle Filter and Iterative SIFT, Robotics and Autonomous Systems 54 (9) (2006), 758-765.

[30] H. Lu, H. Zhang, J. Xiao, F. Liu, Z. Zheng, Arbitrary Ball Recognition Based on Omni-directional Vision for Soccer Robots, RoboCup 2008: Robot Soccer World Cup XII (2009), 133-144.

[31] A. Bonarini, P. Aliverti, and M. Lucioni, An Omnidirectional Vision Sensor for Fast Tracking for Mobile Robots, IEEE Transactions on Instrumentation and Measurement 49 (3) (2000), 509-512.

[32] E. Menegatti, A. Pretto, A. Scarpa, and E. Pagello, Omnidirectional Vision Scan Matching for Robot Localization in Dynamic Environments, IEEE Transactions on Robotics 22 (3) (2006), 523-535.

[33] M. Lauer, S. Lange, and M. Riedmiller, Calculating the perfect match: An efficient and accurate approach for robot self-localization, RoboCup 2005: Robot Soccer World Cup IX (2006), pp. 142-153.

[34] H. Andreasson, A. Treptow, and T. Duckett, Self-Localization in non-statinary environments using omni-directional vision, Robotics and Autonomous Systems 55 (7) (2007), 541-551.

[35] T. Svoboda, and T. Pajdla, Matching in Catadioptric Images with Appropriate Windows, and Outliers Removal, CAIP 2001, LNCS 2124, pp. 733-740, 2001.

[36] S. H. Ieng, R. Benosman, and J. Devars, An Efficient Dynamic Multi-Angular Feature Points Matcher for Catadioptric Views, in: Proceedings

of 2003 International Workshop on Omnidirectional Vision and Camera Networks, 2003.

[37] A. Pronobis, and B. Caputo, COLD: The Cosy Localization Database, the International Journal of Robotics Research 28 (5)(2009), 588-594.

[38] {http://www.ee.oulu.fi/mvg/page/lbp_bibliography}

[39] T. Ojala, M. Pietik*ä*inen, T. M*ä*enp*ää*, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Trans. Pattern Anal. Mach. Intell. 24 (7)(2002), 971-987.

[40] T. Ahonen, A. Hadid, M. Pietik*ä*inen, Face description with local binary patterns: application to face recognition, IEEE Trans. Pattern Anal. Mach. Intell. 28 (12)(2006), 2037-2041.

[41] M. Heikkil*ä*, M. Pietik*ä*inen, A texture-based method for modeling the background and detecting moving objects, IEEE Trans. Pattern Anal. Mach. Intell. 28 (4)(2006), 657-662.

[42] M. Pietik*ä*inen, T. Nurmela, T. M*ä*enp*ää*, M. Turtinen, View-based recognition of real-world textures, Pattern Recognition 37 (2)(2004), 313-323.

[43] {http://www.vlfeat.org/~vedaldi/code/sift.html}

[44] L. Ledwich, and S. Williams, Reduced SIFT Features for Image Retrieval and Indoor Localisation, in: Proceedings of Australian Conference on Robotics and Automation, 2004.