A Convolutional Treelets Binary Feature Approach to Fast Keypoint Recognition

Chenxia Wu, Jianke Zhu, Jiemi Zhang, Chun Chen, and Deng Cai

College of Computer Science, Zhejiang University
chenxiawu@hotmail.com, jkzhu@zju.edu.cn, jmzhang10@gmail.com,
chenc@zju.edu.cn, dengcai@cad.zju.edu.cn

Abstract. Fast keypoint recognition is essential to many vision tasks. In contrast to the classification-based approaches [1,2], we directly formulate the keypoint recognition as an image patch retrieval problem, which enjoys the merit of finding the matched keypoint and its pose simultaneously. A novel convolutional treelets approach is proposed to effectively extract the binary features from the patches. A corresponding sub-signature-based locality sensitive hashing scheme is employed for the fast approximate nearest neighbor search in patch retrieval. Experiments on both synthetic data and real-world images have shown that our method performs better than state-of-the-art descriptor-based and classification-based approaches.

1 Introduction

Recognizing feature point is essential to many vision tasks including motion analysis, image-based visual servoing, object tracking and recognition [3]. As the object changes its appearance with different views and illumination conditions, it is challenging to achieve effective keypoint recognition performance. Moreover, there are some large deformations for the nonrigid objects [4,5,6].

To tackle the above problem, a common approach is to build the affine-invariant descriptors of local image patches, i.e., SIFT [7] and SURF [8], which usually incur heavy computational burden limiting their capability in the real-time applications. Although keypoint matching can be speeded up by compressing the descriptors with the dimensionality reduction techniques [9,10], such approaches still depend on the extracted feature descriptors. Recently, directly extracting the binary string features [11,12] from small patches surrounding the keypoints has been introduced to achieve realtime keypoint recognition.

Keypoint recognition can also be treated as a classification task [13], in which each keypoint to be matched in a model image corresponds to a class label. Realtime object detection has been achieved by taking advantage of the efficient classifiers like randomized trees [1] and ferns [2]. However, the pose information of each keypoint patch in these methods is lost in the training process. It is important to several tasks, such as pose estimation [4], nonrigid object detection [5], robot localization [14] and object recognition [15]. Additionally, these classification-based approaches are typically quite memory demanding.

In this paper, we address the above limitations by formulating the keypoint recognition as an image patch retrieval problem. Motivated from ferns [2], a large number of patches for each keypoint in the model image are generated under multiple views. Instead of resorting to a classifier, we extract features from those generated patches and record their pose information by homography, and then build a database of features and pose information. By retrieving the nearest neighbor in the database, the keypoint label and the pose for a query patch can be estimated simultaneously.

To reduce the computational cost and memory requirement, we generalize the *treelets* [16] with a convolutional architecture [17] to effectively extract the binary features for realtime application. Furthermore, we employ an efficient Locality Sensitive Hashing (LSH) [18] based on the sub-signatures of the binary feature to speed up the retrieval. We have conducted extensive experiments on both synthetic data and real-world images to compare our proposed Convolutional Treelets Binary Feature Retrieval (CT-BFR) based keypoint recognition approach against the state-of-the-art techniques, which not only shows leading recognition performance but also obtains desirable pose information.

In summary, the main contributions of this paper are: (1) a novel binary feature using the effective treelets transform; (2) a convolutional scheme to reduce the computational cost; (3) a fast binary feature retrieval scheme based on the sub-signature-based LSH to keypoint recognition that matches the keypoints and estimates their poses simultaneously.

2 Related Work

Keypoint matching has been received intensively attentions in computer vision. A typical approach to this problem is to build affine-invariant descriptors for local image patches and compare them across images. SIFT [7] and SURF [8] are the most popular techniques among them. However, the drawback of high computational cost limits their applied fields especially for realtime applications.

Recently, many methods have been presented to speed up the feature matching and reduce the memory consumption, which can be roughly divided into two categories. One kind of approach is to reduce the long descriptors into the short ones, which can be achieved by employing the dimensionality reduction techniques, such as PCA [3] and LDA [9]. An even more drastic dimensionality reduction can be achieved by hash functions that reduce SIFT descriptors to binary strings [10,19,20]. These approaches require to compute the full descriptors beforehand for further processing. Therefore, the efficacy is mainly subject to their original descriptors. To deal with this problem, CARD[21] extracts descriptors based on lookup tables and employs a learning-based sparse hashing to convert the extracted descriptors to short binary codes. Moreover, BRIEF[11] and ORB[12] try to directly extract the binary strings from image patches.

Another group of research is based on the fast classifiers [13,1,2]. By synthesizing a large number of small patches around the keypoints, the keypoint recognition is treated as a classification problem, in which each class contains the set of generated patches. These generated patches are used to train a classifier, such as random trees [1], ferns [2] and generic trees [22]. However, they usually require large memories and do not provide keypoint pose information.

Our work is also related to patch rectification method [4], which focuses on perspective rectification after matching the keypoints. Although having achieved the promising object tracking results, such method is limited to relatively few big patches for a model image while our approach can obtain hundreds of matched patches along with their poses in realtime.

3 Keypoint Recognition by Retrieving Binary Features

In this paper, we formulate the keypoint recognition as an image patch retrieval problem. Given a model image containing the object, a subset of the keypoints is selected by deforming the images many times, applying the keypoint detector and keeping track of the number of times the same keypoint is detected. The keypoints that are found most often are assumed to be the most stable and retained [2]. Each stable keypoint is assigned a unique class label. Our task is to correctly map the query patch to its corresponding keypoint in the model image and estimate its pose, which is represented by homography between the retrieved patch in the model image and the query patch in the test image.

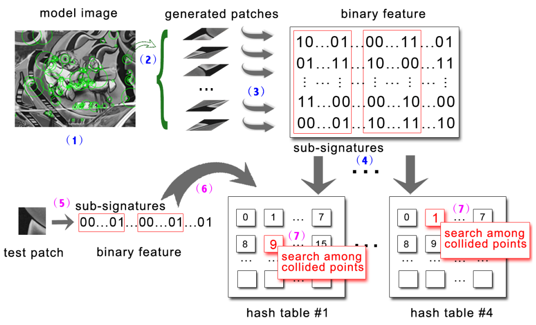

Fig. 1. Illustration of the proposed BFR-based keypoint recognition. Build the patch database: 1) detect the most stable keypoints; 2) generate sample images and get the patches; 3) extract the binary feature for each patch; 4) create hashtables using the sub-signatures of features. Match a detected keypoint patch: 5) extract the binary feature for the query patch; 6) hash the extracted feature to the corresponding hashtables using its sub-signatures; 7) find its nearest neighbor within the collided points to retrieve the patch label and pose.

Similar to [1,2], a database $\chi = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ of keypoint patches for each class is built by synthesizing thousands of example images using the homographic matrices with randomly picked affine deformations by sampling the deformation parameters from a uniform distribution, adding Gaussian noise to each sample image, and smoothing with a Gaussian filter of size 7×7. Let a 2-tuple (c, H)

denote the class label of a keypoint patch and the homographic matrices to generate it. Formally, we aim to retrieve the most similar patch of the query patch $\mathbf{t}$ in the database in order to estimate its class label and pose:

$$\begin{aligned}(\hat{c}, \hat{H})_{\mathbf{t}} &= (c, H)_{N(\mathbf{t})} \\ N(\mathbf{t}) &= \arg\min_{\mathbf{x}_i} d(\mathbf{t}, \mathbf{x}_i), \mathbf{x}_i \in \chi\end{aligned} \tag{1}$$

where $N(\mathbf{t})$ represents $\mathbf{t}$'s nearest neighbor, and $d(\mathbf{t}, \mathbf{x}_i)$ measures the distance between $\mathbf{t}$ and $\mathbf{x}_i$ in some metric space.

To cover a sufficient number of views, the total number of generated example images is typically quite large. Therefore, it is a challenging task to achieve real-time performance with reasonable memory consumption. To reduce the computational cost and storage requirement, we suggest a scheme to directly extract the binary features from image patches. Therefore, Hamming distance can be used to measure the similarity among these binary features, which can be computed extremely fast on modern CPUs that often provide an optimized instruction set to perform a XOR or bit count operation in parallel. Moreover, we found that 256-bit binary feature or even less is sufficient to achieve very good results.

Note that the sub-signature of binary features can be used as the hash function for LSH [18,12]. The power of such technique lies in its capability of retrieving nearest neighbors with a high probability with sufficient number of hash tables. We store the patches with a common sub-signature of binary features in the same bucket of a hash table. Several hash tables are built according to the different sub-signatures. Given a query patch, we first hash it into a very small subset with the corresponding sub-signatures in common, and then compare it with these collided patches through brute-force search based on hamming distances. Taking advantage of the binary feature representation, the whole search process is essentially fast even the size of the database is remarkably large.

We name the presented method as Binary Feature Retrieval (BFR) based keypoint recognition. Fig. 1 summarizes the whole procedure of our approach. In practice, BFR estimates a class label and the pose information to each query patch. For each class of the detected keypoints in the model image, there should be a unique matching point for a certain test image or frame. We simply choose the nearest one among those test patches with the same class label.

4 Convolutional Treelets Binary Feature

The key of our proposed approach is to extract the effective binary features from each patch. To this end, we introduce treelets transform [16] and a thresholding scheme to obtain the discriminative binary embeddings. We also propose an efficient convolutional treelets scheme to reduce the projection time.

4.1 Treelets

Treelets [16] is a multi-resolution analysis tool, which provides an orthogonal basis to reflect the low intrinsic dimensionality of the noise-free data. In this paper, each patch is simply represented by packing the raw intensities into a p dimensional vector $\mathbf{x} = [x_1, \cdots, x_p]$. The main task of treelets is to provide an orthogonal basis by constructing a tree using agglomerative clustering on dimensions of the data. To this end, the index of each dimension is recorded in a set $S = 1, 2, \cdots, p$. At each level of the tree, the two most similar dimensions are merged together and replaced by two uncorrelated new dimensions calculated from local Principal Component Analysis (PCA). After this, the dimension with lower variance is removed from S. Such process is repeated recursively on the dimensions in S until the root node at level $L = p - 1$ is reached. Here, the similarity score M_{ij} between dimensions are measured by the correlation coefficient: $M_{ij} = C_{ij}/\sqrt{C_{ii}C_{jj}}$, where $C_{ij} = E[(x_i - Ex_i)(x_j - Ex_j)^\top]$ is the covariance of pixel intensities of the patch. The treelets algorithm is summarized as below:

- At Level $L = 0$
  Set the basis $B_0 \in R^{p\times p}$ to identity matrix, associate to original coordinates of the patches, the Dirac basis.
  Compute the initial covariance matrix $C^{(0)} \in R^{p\times p}$ and $M^{(0)} \in R^{p\times p}$.
  Initialize the set $S = 1, 2, \cdots, p$.
- Repeat for $L = 1, \cdots, p - 1$
  1. Find the two most similar dimensions (α, β) with respect to $M^{(L-1)}$.
  2. Perform a local PCA on the pair (x_α, x_β) to find a Jacobi rotation matrix J that decorrelates x_α and x_β, such that $C^{(L)}_{\alpha\beta} = C^{(L)}_{\beta\alpha} = 0$ and $C^{(L)}_{\alpha\alpha} > C^{(L)}_{\beta\beta}$, where $C^{(L)} = JC^{(L-1)}J^\top$.
  3. Update the basis $B_L = JB_{L-1}$ and the similarity matrix $M^{(L)}$ accordingly. Actually, $\mathbf{x}^{(L)} = J\mathbf{x}^{(L-1)}$;
  4. Remove the dimension with lower variance from the set S: $S = S\backslash\{\beta\}$.

In the treelets transform, it can be clearly seen that the covariance structure of the data is explored by PCA and such analysis is performed locally. Instead of the global representation, treelets merges the two most similar pixels in each layer to detect internal highly correlated localized structures in patches. Additionally, it is able to find the basis of underlying noiseless data while PCA constructs an optimal linear representation of noisy observations [16].

4.2 Treelets Binary Feature

In practice, one can select the 'best m-basis' from the top level basis B_{p-1} of the treelets to project the data to low-dimensional coordinates. In this case, we can treat the basis B_{p-1} as p projection vectors $(\mathbf{w}_1, \cdots, \mathbf{w}_p)$. They are then sorted by the energy (variance) of each projected training patches $\mathbf{var}(\mathbf{w}_i^\top \mathbf{x})$. The top m ones with the highest energy are selected.

Typically, we use the following function to obtain the m-bit binary feature

$$\mathbf{y} = \mathbf{sgn}(W^\top \mathbf{x} - \mathbf{b}) \tag{2}$$

where $\mathbf{y}$ is the m-bit binary feature. $W \in R^{p\times m}$ is a projection matrix, which is formed by selecting the 'best m-basis' with the highest energy from B_{p-1}.

$\mathbf{b} \in R^{m \times 1}$ is a thresholding vector, which is set to the mean of all the projected training data $W^\top X, X = [\mathbf{x}_1, \cdots, \mathbf{x}_n]$. Obviously, the computational cost of projecting a patch is $O(pm)$, where m is the bit length of the binary feature.

Since the analysis is local, the variances of the projections by treelets are more balanced than those by PCA, which preserves a low quantization error [19]. As the basis are selected in energy descending order, the lower bits have more variances than the higher ones. It means the quality of the bits is in descending order because the representation power for each bit decreases along with the energy. This leads to good performance when the bit length is short. Moreover, we can choose the sub-signatures in bit energy order as the LSH hash functions to obtain good results.

4.3 Convolutional Treelets

Although the treelets method captures the underlying geometric structure of the data, it still requires lots of computational power to project the patch onto the embedded space. Motivated from convolutional neural networks [17], we propose an efficient two-layer convolutional scheme to facilitate the realtime keypoint recognition, which employs treelets as the basic element for unsupervised learning. Similar method has been used to learn a hierarchical representation for action recognition [23].

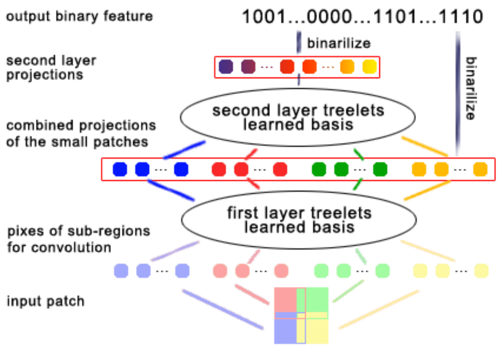
output binary feature
1001...0000...1101...1110
binarilize
second layer projections
second layer treelets learned basis
binarilize
combined projections of the small patches
first layer treelets learned basis
pixes of sub-regions for convolution
input patch

Fig. 2. Extract binary feature from each image patch using two-layer convolutional treelets. The input patches are convolved with the basis learned in first-layer. Then the combined projections are served as the input for the next layer. The combined outputs of the two-layer treelets are thresholded to output the final binary code.

The key ideas of this approach are as follows. We first learn the first-layer treelets on small patches sampled from the sub-regions of collected image patches. The 'best m_s basis' W_1 with the highest energy are selected as the projection matrix from the first-layer treelets. We take the learned first-layer and convolve with the whole region of the image patches. Suppose n_s small patches are used for the first-layer, then $m_1 = m_s \times n_s$ combined projections of the convolution step are then given as input to the next layer which is also implemented by another treelets algorithm. The 'best m_2 basis' W_2 with the highest energy of the second layer treelets are used as the projection matrix for the second-layer. Finally, the projections of two layers are thresholded by the function in Eqn. 2 and then concatenated as the final binary feature. We illustrate the architecture of the presented two layer convolutional treelets in Fig. 2.

The presented two-layer convolutional scheme is able to extract the binary features locally and globally. The first layer treelets aims to acquire the local information of sub-regions, which is effective for capturing the pose variations. Moreover, it provides a compact input for the second layer treelets, which is smoothed by the convolution. The second layer treelets manages to obtain the binary features for the whole patch, which combines the local information of each sub-region. Benefit from the convolution, the output of the second layer treelets is robust to the noise. In our empirical study, the proposed two-layer convolutional architecture is more effective than the single layer treelets method.

Note that the two-layer convolutional treelets also enjoys the merits as the single layer treelets that the quality of the bits is in descending order. Importantly, the two-layer convolutional architecture is more efficient than the single layer treelets algorithm, as it employs two low dimensional treelets instead of one high dimensional treelets. The computational time on patch projection is reduced to $O(p_s m_1 + m_1 m_2)$, which depends on the small patch with size p_s rather than the original large patch size p. m_1, m_2 are the output bit lengths for each treelets layer, and $m_1 + m_2 = m$. Practically, the small patch size p_s is much smaller than the original patch size p, which leads to an efficient implementation compared with the computational cost of the single treelets layer $O(pm)$.

5 Experiments

5.1 Comparison Schemes and Setup

We employ the first image in 'wall' and 'graffiti' in Oxford Dataset [1] as the model image to synthesize the data for our numerical evaluation. We warp the model images by repeatedly applying random affine deformations and detect corners in the deformed images. The 400 most stable keypoints are selected by counting how many times they are detected. The patch database is built by sampling the randomly deformed model image. The size of each patch is set to 32×32 in our experiments. To perform comparisons, we represent the affine image deformations as 2×2 matrices: $R_\theta R_\phi diag(\lambda_1, \lambda_2) R_\phi$, where $diag(\lambda_1, \lambda_2)$ is a diagonal 2×2 matrix and R_γ represents a rotation of γ. We warped the original images using such deformations computed by randomly choosing θ and ϕ in $[0 : 2\pi]$ and λ_1 and λ_2 in $[0.5 : 1.5]$. Gaussian noise is added to these warped images to increase the robustness. The testing set is obtained by generating a separate set in the same affine deformation range.

[1] `http://www.robots.ox.ac.uk/~vgg/data/data-aff.html`

To facilitate fair comparison, we evaluate four binary features for retrieval based keypoint recognition on the synthetic data: the proposed convolutional treelets binary feature (CT), single layer treelets binary feature (ST), fast binary feature BRIEF [11], and spectral hashing (SH) [24]. Note that the rotation or scale invariant binary features are not considered in our framework since they are not discriminable of pose information. We also compare the proposed BFR-based keypoint recognition with these binary features against FERNS [2].

Typically, a set of patches are needed to learn the functions that extract the binary feature in CT, ST and SH. To this end, we randomly select 50K patches from the database. For our proposed CT method, 50K small patches with the size of 12×12 are randomly sampled from the sub-regions in the original patches to train the first layer treelets. For $m = 256$-bit, m_1 and m_2 are set to 150 and 106, respectively. Also, we retain the similar ratio of m_1 and m_2 for other bit-length of the binary feature.

For BFR, we employ the 16-bit sub-signature of each binary feature as the hash function for LSH. The patches having a common sub-signature are stored in the same bucket of a hash table. Several hash tables are built using different sub-signatures, and a query patch is hashed to the collided points sharing with the same sub-signatures. We choose the sub-signature in bit order for three binary features: CT, ST, SH, as the quality of the bits is in descending order. More specifically, the first 16-bit is used to build the first hashtable, the following 16-bit to build the second, and so on. For BRIEF, we just select the sub-signature for hashing randomly since it adopts the uniform distribution and treats each bit equally. FERNS [2] is trained with the whole database.

In our experiments, the correct recognition rate is employed as the evaluation metric for keypoint matching, and the accuracy of pose estimation is measured by the root mean square error (RMSE) between the estimated homography matrix and ground truth. All of our experiments were carried out on a PC with Intel 2.8GHz CPU and 4GB RAM.

5.2 Evaluation on Synthetic Data

We now study the performance of the proposed approach on synthetic data from three aspects: binary feature, hashing method and training examples.

Binary Feature. To make it clear, we directly perform exhaustive brute-force search on the whole database. In Fig. 3, we plot the correct recognition rate and RMSE of homography matrices vary along with the bit length of binary features. It can be observed that the keypoint recognition and pose estimation accuracies increase with the bit length, and become saturate at 256-bit. Thus, the bit length of all binary features is set to 256 in the following. The proposed treelets-based binary features outperform the BRIEF and spectral hashing, while the convolutional treelets binary feature performs better than the single layer treelets method. Furthermore, we find that BRIEF obtains good results with longer bit while our proposed method performs especially well with fewer bits. This is mainly because the bits for BRIEF are randomly picked and the discriminative power are uniformly distributed to each bit.

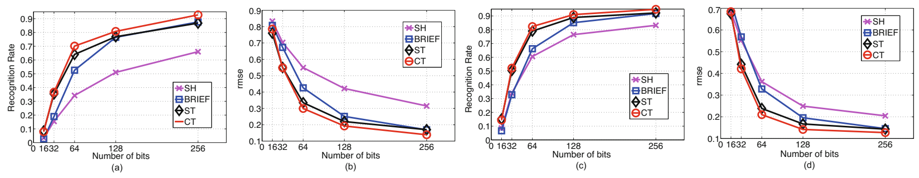

Fig. 3. Comparison of keypoint recognition and pose estimation performance for the binary features varied with bit length. (a), (c) Recognition rate for ‘wall’ and ‘graffiti’. (b), (d) RMSE between estimated homography and ground truth for ‘wall’ and ‘graffiti’.

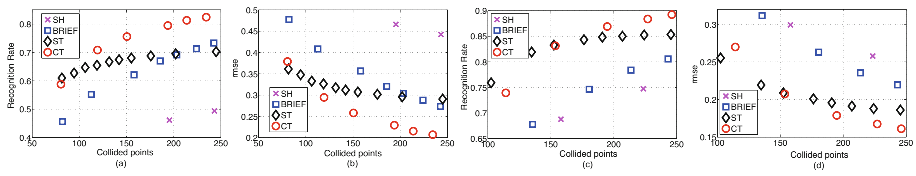

Fig. 4. Comparison of various binary features using retrieval-based keypoint recognition with the different number of collided points. For ‘wall’, 1 to 2 hashtables for SH, 2 to 7 hashtables for BRIEF, 5 to 14 hashtables for ST, 1 to 6 hashtables for CT. For ‘graffiti’, 1 to 2 hashtables for SH, 2 to 5 hashtables for BRIEF, 4 to 11 hashtables for ST, 1 to 5 hashtables for CT. (a), (c) Recognition rate for ‘wall’ and ‘graffiti’. (b), (d) RMSE between estimated homography and ground truth for ‘wall’ and ‘graffiti’.

Hashing. We study the fast nearest neighbor search on the binary code using LSH. Fig. 4 illustrates the keypoint recognition and pose estimation results with the different number of collided points that are selected for refining the approximate search. It can be seen that the more collided points we use, the better results we can obtain for all methods. As the large number of collided points leads to heavy computational burden, there is a tradeoff between the efficiency and keypoint recognition performance. So, we select 250 collided points in our experiments. In Fig. 4, we can find that our proposed convolutional treelets binary features outperforms BRIEF and spectral hashing at a large margin, which indicates that it is very effective for sub-signature-based LSH. Also, single layer treelets binary feature performs better with fewer collided points. On the other hand, BRIEF obtains the good results with more collided points. Surprisingly, spectral hashing method performs quite poor, which is ineffective for the LSH scheme. In fact, the spectral hashing extracts the binary features from the PCA projections, which cannot effectively capture the local information.

Training Examples. We investigate the keypoint recognition and pose estimation performance on the different number of training examples in the database, which affect the training time and the allocated memory. To make fair comparison, we tune the number of hashtables for each each feature and set the same number of collided points for all methods. Fig. 5 shows the correct recognition rate and homography estimation error varying with the number of training examples. Obviously, the performance of all the methods are greatly improved with

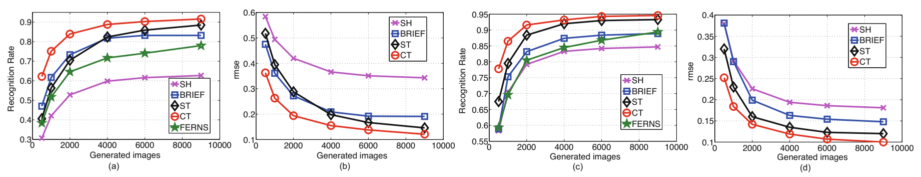

Fig. 5. Comparison of different approaches varied with the number of generated images in database. (a), (c) Recognition rate for 'wall' and 'graffiti'. (b), (d) RMSE between estimated homography and ground truth for 'wall' and 'graffiti'.

Table 1. R500-T500 comparisons of the percentage of inlier matches/the number of detected correct matches across real-world images: 'graffiti' (viewpoint changing), 'boat' (rotation+zooming), 'wall' (viewpoint changing), 'leuven' (lighting), 'bikes' (blur).

	SIFT	SURF	BRIEF	ORB	FERNS	CT-BFR	RMSE
graffiti3	0.660/159	0.552/111	0.268/38	0.580/112	0.303/90	**0.747/115**	0.274
graffiti4	0.359/70	0.366/52	0.003/4	0.245/37	0.141/29	**0.546/65**	0.376
boat3	0.869/252	0.774/151	0.009/1	0.730/130	0.531/178	**0.876/212**	0.142
boat4	0.664/146	0.647/101	0.000/0	0.176/28	0.361/115	**0.748/178**	0.200
wall3	0.822/222	0.797/173	0.920/242	0.639/145	0.619/205	**0.953/222**	0.265
wall4	0.266/55	0.299/50	**0.641/125**	0.110/22	0.172/50	0.418/61	0.226
leuven4	0.568/183	0.595/153	**0.948/250**	0.494/87	0.465/139	0.724/110	0.151
bikes4	0.824/255	0.879/218	0.945/242	0.801/153	0.802/283	**0.964/239**	0.168
time	1209.4ms	414.3ms	235.4ms	278.3ms	50.9ms	157.7ms	

more training examples. Moreover, the proposed convolutional treelets feature performs the best. We also compare with the Naive Bayesian classifier-based keypoint recognition method FERNS[2], which actually employs the BRIEF binary features with $20 \times 14 = 280$ bits. It can be observed that 256-bit BRIEF feature using BFR outperforms FERNS with 280-bit binary feature. This reveals that the proposed BFR-based keypoint recognition scheme is more effective than the Naive Bayesian classifier approach. More importantly, BFR method is capable of retrieving pose information of the query patch. As shown in Fig. 5, the promising homography estimation results with very low RMSE are achieved through our proposed convolutional treelets feature using BFR.

5.3 Evaluation on Oxford Dataset

We investigate the presented methods on Oxford Dataset, in which five image datasets under different conditions are selected, including 'graffiti' (viewpoint changing), 'boat' (rotation+zooming), 'wall' (viewpoint changing), 'leuven' (lighting), 'bikes' (blur). The first image in each category is treated as the model image, and the homography matrix is used to compute the ground truth for both keypoint recognition and pose estimation. The percentage of inlier matches and the number of correct matches are employed as the performance

Table 2. R2000-T2000 comparisons of the percentage of inlier matches/the number of detected correct matches: 'graffiti' (viewpoint changing), 'boat' (rotation+zooming), 'wall' (viewpoint changing), 'leuven' (lighting), 'bikes' (blur).

	SIFT	SURF	BRIEF	CARD	FERNS	CT-BFR	RMSE
graffiti3	0.536/437	0.497/305	0.190/95	0.588/387	0.304/355	**0.618/433**	0.302
graffiti4	0.268/173	0.342/175	0.133/6	0.195/87	0.112/131	**0.508/263**	0.376
boat3	0.775/752	0.714/471	0.000/0	0.883/820	0.506/647	**0.910/769**	0.103
boat4	0.476/340	0.551/271	0.000/0	**0.736/531**	0.303/362	0.707/541	0.281
wall3	0.816/902	0.771/598	0.901/883	0.730/640	0.623/810	**0.929/845**	0.282
wall4	0.300/274	0.293/174	**0.671/491**	0.211/152	0.189/231	0.433/196	0.220
leuven4	0.557/670	0.548/574	**0.961/1109**	0.650/641	0.452/560	0.606/470	0.173
bikes4	0.791/785	0.819/711	0.936/952	0.898/770	0.793/889	**0.945/852**	0.202
time	2687.5ms	1376.5ms	958.5ms	724.5ms	512.5ms	610.3ms	

metrics. We compare our proposed convolutional treelets binary feature retrieval (CT-BFR) based method [2] against state-of-the-art approaches, i.e., SIFT [7], SURF [8], SURF-BRIEF [11], ORB[12], CARD[21] and FERNS [2]. In contrast to the BRIEF with BFR scheme for the evaluations on synthetic data, we directly employ its original implementation in the following.

For the fair comparison with different approaches, the number of detected keypoints for one image is set identically and two cases are compared: reference image with 500 keypoints—test image with 500 keypoints (R500-T500), reference image with 2000 keypoints—test image with 2000 keypoints (R2000-T2000). Since the ORB program only provides 500 keypoints setting and CARD program only provides 2000 keypoints setting, we test them in their own provided cases.

We use the original implementations for all compared methods. For binary feature based methods BRIEF, ORB, CARD and CT-BFR, 256 bit length are used. For FERNS, 32 ferns of size 8 which amounts to 256 bit binary feature are used to establish matches. As in [11], SURF detector is used to detect the stable keypoints for BRIEF. For FERNS and CT-BFR, the keypoint detector that considers extrema of the Laplacian over 4 octaves is used as described in [2], 5K images are generated to build the database. We first assign the most confident label for each detected keypoint in the test image, and then fuse those keypoints of the same label by selecting the one with the smallest Hamming distance for CT-BFR or the largest posterior probability for FERNS. To find matched pairs, no thresholding is used for FERNS as did in [2] and a simple thresholding (no farther than 50) is used for CT-BFR. For the other descriptor-based methods, a one-to-one symmetric nearest neighbor matching scheme [25], two keypoints which are the nearest neighbors to each other are treated as a matched pair, is used to find matched pairs.

We illustrate the overall evaluation results for two cases in Table 1 and Table 2. These results reveals a number of interesting points as follows:

[2] Our implementation is available at `http://www.cse.cuhk.edu.hk/~jkzhu/ctree`

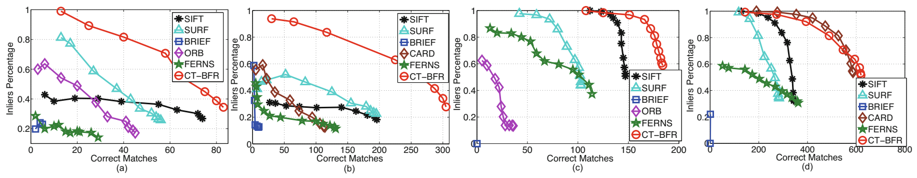

Fig. 6. Inliers percentage vs. Correct matches curves: (a)R500-T500 on 'graf4'. (b)R2000-T2000 on 'graf4'. (c)R500-T500 on 'boat4'. (d)R2000-T2000 on 'boat4'.

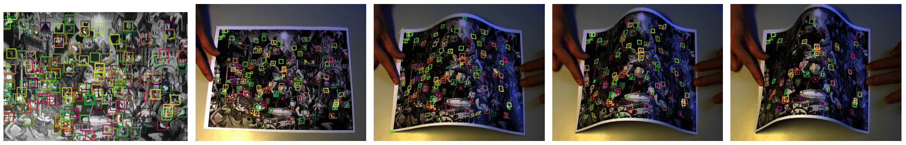

Fig. 7. Keypoints recognition and pose estimation on the video with severe bending deformations. The first image is the model image and the rest are example frames.

- The proposed CT-BFR achieves the best results in most cases especially on the challenging image pairs with large rotation, scale changes and addictive noise. We also report the average time spent on keypoint recognition including keypoint detection, feature extraction and matching. It can be seen that the proposed method is comparable with FERNS and more efficient than other methods. The binary code-based methods are more efficient than the traditional methods SURF and SIFT. Additionally, it is important to note that our approach requires much less memory than the tree-based method FERNS. In our experiments, FERNS with 256-bit feature consumes more than 650MB memory while our 256-bit feature only takes 150MB. We also show the promising homography estimation results using CT-BFR by RMSE.
- Comparing the results of two tables, the performance of classification-based methods FERNS and CT-BFR do not degrade when the number of detected keypoints increases. 1-NN classifier used in our method performs better and can provide the pose information.
- BRIEF is not invariant to the in-plane rotation, which fails on 'graffiti' and 'boat'. However, it performs well in the cases of lighting variations and perspective distortions. ORB was developed to make up the shortage of BRIEF and it does perform better than BRIEF in the cases with large rotation changes. However, it is no better than BRIEF in the cases of lighting variations and perspective distortions. SIFT and SURF can deal with most conditions while cannot perform well on some tough cases like 'graffiti4', 'boat4', 'wall4'. CARD performs better than SIFT, SURF and costs less time.

To further study the performance of each compared method, we also plot the inliers percentage versus correct matches curves on 'graf4' and 'boat4' in Fig. 6 by thresholding the accepted distance with different values. Here we query the keypoints in the test image for the nearest neighbor in the reference image. We

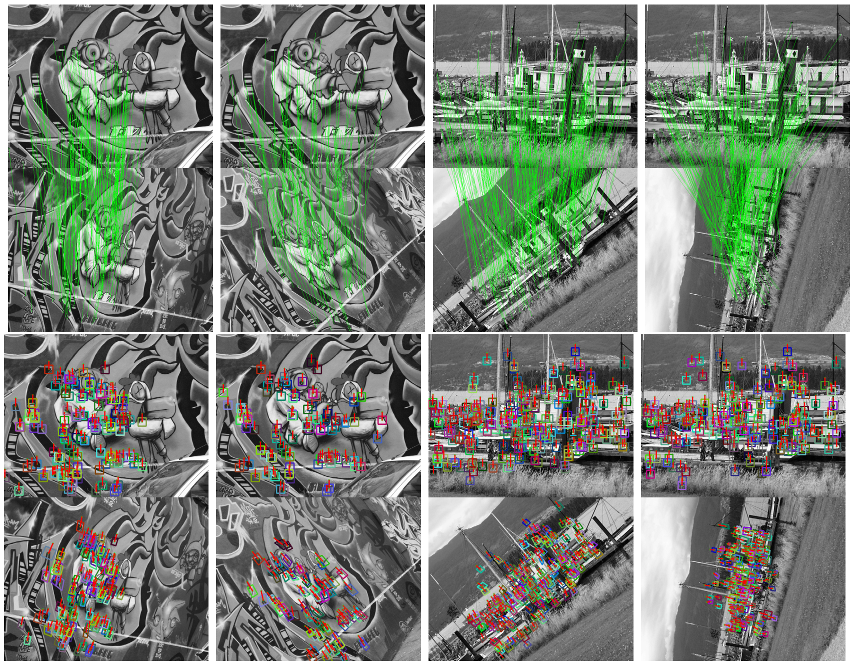

Fig. 8. Examples of keypoint matching and pose estimation using CT-BFR. The first row shows model images, and the second row plots test images.

find that the results accords with those above using the one-to-one symmetric nearest neighbor matching scheme. Our proposed method performs better than other methods at a large margin on 'graf4'. On 'boat4', CT-BFR and CARD outperform other methods. In Fig 8, we also show the example results of CT-BFR on pose estimation. It can be clearly observed that the proposed CT-BFR method not only obtains the correct keypoint matching but also estimates the desirable homography transformation for each keypoint patch.

5.4 Evaluation on Videos

To further demonstrate the effectiveness of our proposed CT-BFR approach, we perform the keypoint matching and pose estimation on a real video containing the planar objects with severe bending deformations. Fig. 7 plots the model image and the example results. It can be seen that the presented method estimates the accurate affine transformation for each patch in the input frame.

6 Conclusion and Future Work

This paper proposed an efficient image patch retrieval-based approach to solve the keypoint matching and pose estimation simultaneously. Moreover, a novel convolutional treelets method was presented to effectively extract the binary features from the patch surrounding the keypoint. An efficient sub-signature-based locality sensitive hashing scheme was employed for the fast approximate nearest neighbor search in patch retrieval. We have conducted extensive evaluations on both synthetic data and real-world images. The encouraging results showed that our method performs better than the state-of-the-art approaches. Despite of these promising results, the major limitation of our method is the dependence of offline training. For future work, we will address this issue by studying the idea of online learning and extend our technique to nonrigid object tracking [5].

Acknowledgments. The work was supported by National Natural Science Foundation of China under Grants (61103105 and 91120302), National Basic Research Program of China (973 Program) under Grant 2011CB302206 and Fundamental Research Funds for the Central Universities.

References

1. Lepetit, V., Fua, P.: Keypoint recognition using randomized trees. PAMI 28 (2006)
2. Ozuysal, M., Calonder, M., Lepetit, V., Fua, P.: Fast keypoint recognition using random ferns. PAMI 32 (2010)
3. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. PAMI 27 (2005)
4. Hinterstoisser, S., Lepetit, V., Benhimane, S., Fua, P., Navab, N.: Learning real-time perspective patch rectification. IJCV 91 (2011)
5. Zhu, J., Lyu, M.R.: Progressive finite newton approach to real-time nonrigid surface detection. In: Proc. Conf. Computer Vision and Pattern Recognition (2007)
6. Zhu, J., Lyu, M.R., Huang, T.S.: A fast 2d shape recovery approach by fusing features and appearance. IEEE Trans. Pattern Anal. Mach. Intell. 31, 1210–1224 (2009)
7. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV 60 (2004)
8. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (surf). CVIU 110 (2008)
9. Hua, G., Brown, M., Winder, S.: Discriminant embedding for local image descriptors. In: ICCV (2007)
10. Strecha, C., Bronstein, A., Bronstein, M., Fua, P.: Ldahash: Improved matching with smaller descriptors. PAMI 34 (2011)
11. Calonder, M., Lepetit, V., Ozuysal, M., Trzinski, T., Strecha, C., Fua, P.: BRIEF: Computing a Local Binary Descriptor Very Fast. PAMI 34 (2011)
12. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: Orb: An efficient alternative to sift or surf. In: ICCV (2011)
13. Lepetit, V., Pilet, J., Fua, P.: Point matching as a classification problem for fast and robust object pose estimation. In: CVPR (2004)
14. Goedeme, T., Tuytelaars, T., Van Gool, L.: Fast wide baseline matching for visual navigation. In: CVPR (2004)
15. Rothganger, F., Lazebnik, S., Schmid, C., Ponce, J.: 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. IJCV 66 (2006)
16. Lee, A.B., Nadler, B., Wasserman, L.: Treelets—an adaptive multi-scale basis for sparse unordered data. Annals of Applied Statistics 2 (2008)

17. LeCun, Y., Bengio, Y.: The handbook of brain theory and neural networks (1998)
18. Gionis, A., Indyk, P., Motwani, R.: Similarity search in high dimensions via hashing. In: VLDB (1999)
19. Wu, C., Zhu, J., Cai, D., Chen, C., Bu, J.: Semi-supervised nonlinear hashing using bootstrap sequential projection learning. IEEE Transactions on Knowledge and Data Engineering 99 (2012)
20. Zhang, D., Wang, J., Cai, D., Lu, J.: Self-taught hashing for fast similarity search. In: SIGIR (2010)
21. Ambai, M., Yoshida, Y.: Card: Compact and real-time descriptors. In: ICCV (2011)
22. Calonder, M., Lepetit, V., Fua, P.: Keypoint Signatures for Fast Learning and Recognition. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 58–71. Springer, Heidelberg (2008)
23. Le, Q., Zou, W., Yeung, S., Ng, A.: Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: CVPR (2011)
24. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: NIPS (2008)
25. Zhao, W.L., Ngo, C.W., Tan, H.K., Wu, X.: Near-duplicate keyframe identification with interest point matching and pattern learning. IEEE Transactions on Multimedia 9 (2007)