Brief Papers

Real-Time Keypoint Recognition Using Restricted Boltzmann Machine

Miaolong Yuan, Huajin Tang, and Haizhou Li

Abstract— **Feature point recognition is a key component in many vision-based applications, such as vision-based robot navigation, object recognition and classification, image-based modeling, and augmented reality. Real-time performance and high recognition rates are of crucial importance to these applications. In this brief, we propose a novel method for real-time keypoint recognition using restricted Boltzmann machine (RBM). RBMs are generative models that can learn probability distributions of many different types of data including labeled and unlabeled data sets. Due to the inherent noise of the training data sets, we use an RBM to model statistical distributions of the training data. Furthermore, the learned RBM can be used as a competitive classifier to recognize the keypoints in real-time during the tracking stage, thus making it advantageous to be employed in applications that require real-time performance. Experiments have been conducted under a variety of conditions to demonstrate the effectiveness and generalization of the proposed approach.**

Index Terms— **Classification, deep learning, feature matching, keypoint recognition, real-time tracking, restricted Boltzmann machine (RBM).**

I. Introduction

Recognizing textured patches surrounding salient image features under widely varying camera poses and lighting conditions is a still fundamental issue in computer vision and has a wide range of real applications, such as augmented reality (AR) [1], [2], visual servoing [3], [4], object recognition [5], and visual navigation [6], [7].

Due to the requirements of robustness to partial occlusions and real-time performance, recognition of image patches around interest points is crucial for vision-based applications. Matching keypoints over small baselines has been successfully developed [8]. In small baseline matching, a matching score, such as normalized cross-correlation score, is calculated and compared based on the pixel intensities around the points. However, in most of the applications, such as AR, visual navigation, and 3-D reconstruction from multiple views, point matches need to be established across the images captured under widely varying camera poses. Traditional normalized cross-correlation score-based methods are not invariant to scale and rotation changes.

Manuscript received August 28, 2012; revised October 31, 2013; accepted January 12, 2014. Date of publication February 14, 2014; date of current version October 15, 2014.

M. Yuan is with the Institute for Infocomm Research, Agency for Science Technology and Research, Singapore 138632 (e-mail: myuan@i2r.a-star.edu.sg).

H. Tang is with the Institute for Infocomm Research, Agency for Science Technology and Research, Singapore 138632, and also with College of Computer Science, Sichuan University, Chengdu 610065, China (e-mail: htang@i2r.a-star.edu.sg).

H. Li is with the Institute for Infocomm Research, Agency for Science Technology and Research, Singapore 138632, and also with the School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, NSW 2052, Australia (e-mail: hli@i2r.a-star.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2014.2303478

Although there are many impressive methods reported in the literature, finding keypoint correspondences under large perspectives and varying lighting conditions is still a challenging issue. Most of the wide baseline matching approaches are based on affine invariant regions [4], [9]–[14]. These regions are constructed using affine transformations to adapt to viewpoint changes, such as rotation and scale. They are often described using local descriptors that are invariant to rotation and scale. Among those keypoint matching methods, the scale-invariant feature transform (SIFT) descriptor [14] is a very effective method and has been widely used in many applications. The SIFT descriptor computes a histogram of local oriented gradients around the interest point and stores the bins in a 128-D vector. It tolerates significant local deformations. In [15], principal component analysis (PCA) was applied to refine SIFT descriptors, which are fast for matching. Compared with the 128-D SIFT descriptors, PCA-SIFT is faster for matching, but proved to be less distinctive than SIFT. Meanwhile, the computation time of PCA-SIFT is not alleviated.

To tackle this bottleneck, the speeded up robust features (SURF) method was proposed in [16], which is a robust and efficient descriptor using a Haar wavelet approximation of the determinant of an Hessian Blob detector. Though it is faster than SIFT and more robust against different image transformations, it is still computationally expensive. Recently, [17] proposed an impressive discriminative method using linear discriminant analysis (LDA) and Powell minimization for learning local image descriptors. However, it requires ground truth correspondences between the features, which can be obtained easily using 3-D models estimated from input images. It can also be applied offline to learn descriptors that can be used in the future without any learning.

To achieve real-time performance, Lepetit *et al.* [18], [19] proposed a novel method (referred in this brief as the L&F method) in which matching is formulated as a generic image classification problem using randomized trees. Each keypoint is learned with all the possible appearances that an image feature may take under large perspective and scale variations. The major advantage of the classification-based matching method is that it shifts much of the computation time from the running stage to the learning stage and it is robust to varying viewpoints and lighting conditions, thus making matching reliable and fast. However, it may not be optimal due to the possible complex appearance changes. [20] extended the L&F method by proposing a tracking-by-detection-based online training method. However, a rough 3-D model is required to manually align at least three image features with their corresponding 3-D points to initialize the system. The training process may fail if the image points are not approximately aligned with their 3-D points. Other extensions have also been developed, such as random ferns [21] and online AdaBoost [22] to account for the variations in appearance during tracking.

The training data normally includes inherent noise. It is desirable to model the statistical structures in the noisy training data sets to achieve better recognition performance. In this brief, we take the advantage of the restricted Boltzmann machine (RBM) [23] in

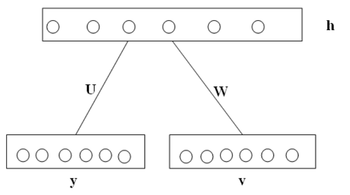

Fig. 1. Undirected bipartite graphical model of an RBM.

modeling training data distributions to develop a real-time keypoint recognition approach. The RBM is a generative model that can learn probability distributions of different types of data including labeled and unlabeled data sets and generate high level features. It is a very powerful tool for dimension reduction and classification [24]. The RBM and its variants have been extensively studied in different applications, such as digital number classification, face classification, modeling motion capture data, and multiobjective optimization [25], [28]–[33]. In this brief, we learn an RBM to model the input training data and the learned RBM is then used as a competitive nonlinear classifier for keypoint recognition. Our experimental results show that the recognition performance using an RBM outperforms that of the L&F method without sacrificing real-time performance.

The remainder of this brief is organized as follows. Section II describes technical details of RBM. Section III describes the proposed real-time keypoint recognition framework using an RBM model. Section IV presents the experimental results and analyzes the performance of the proposed method. Some issues are discussed in Section V. The conclusion is given in Section VI.

II. Restricted Boltzmann Machines

An RBM is an undirected bipartite graphical model, as shown in Fig. 1, where $\mathbf{W}$ are the weights between the hidden and visible units and $\mathbf{U}$ are the weights between the hidden units and the target classes. The visible units $\mathbf{v} = (v_1, v_2, \ldots, v_D)$ and the target units $y \in \{1, 2, \ldots, K\}$ are training data sets. The target units are formatted as 1-of-K scheme and denoted as $\mathbf{y}$. The hidden units $\mathbf{h} = (h_1, h_2, \ldots, h_M)$ are the corresponding learned latent variables, which can be considered as higher level features of the input data. M is the dimension of the hidden units. In an RBM, there are no links between the same layers, but between the visible units $\{\mathbf{v}\}$ and the hidden units $\{\mathbf{h}\}$, and the target units $\{\mathbf{y}\}$ and the hidden units $\{\mathbf{h}\}$, making the conditionals factorize conveniently. RBMs are based on latent variables to learn statistical structures of the input data. The learned RBMs can be used as nonlinear classifiers. It has been reported [25] that an RBM outperforms the standard support vector machines in the classification of the Mixed National Institute of Standards and Technology database [26] handwritten digits.

The joint distribution of the generative model $P(\mathbf{y}, \mathbf{v}, \mathbf{h})$ of an RBM can be represented in the form of an energy-based model, which can be written as follows:

$$P(\mathbf{y}, \mathbf{v}, \mathbf{h}) = \frac{1}{Z}\exp(-E(\mathbf{y}, \mathbf{v}, \mathbf{h})) \tag{1}$$

$$E(\mathbf{y}, \mathbf{v}, \mathbf{h}) = -\mathbf{v}^T\mathbf{W}\mathbf{h} - \mathbf{b}^T\mathbf{h} - \mathbf{c}^T\mathbf{v} - \mathbf{d}^T\mathbf{y} - \mathbf{y}^T\mathbf{U}\mathbf{h} \tag{2}$$

$$Z = \sum_{\mathbf{y}}\sum_{\mathbf{v}}\sum_{\mathbf{h}}\exp(-E(\mathbf{y}, \mathbf{v}, \mathbf{h})) \tag{3}$$

where $\mathbf{b}$, $\mathbf{c}$, and $\mathbf{d}$ are the offsets of the hidden, the visible, and the target units, respectively. Let $\mathbf{v} \in \{0,1\}^D$ denote the observed (visible) binary variables. In our method, they are randomly generated training data sets for keypoints. $y \in \{1, 2, \ldots, K\}$ denotes the keypoint classes extracted from a keyframe, which will be matched to the image feature points extracted from each incoming frame during the online stage. Due to the specific structure of an RBM, as shown in the graphical model in Fig. 1, it is straightforward to have

$$P(\mathbf{h}|\mathbf{v}, \mathbf{y}) = \Pi_j P(h_j|\mathbf{v}, \mathbf{y}) \tag{4}$$

$$P(\mathbf{v}|\mathbf{h}) = \Pi_i P(v_i|\mathbf{h}) \tag{5}$$

$$P(h_j = 1|\mathbf{v}, \mathbf{y}) = \sigma\left(\sum_i W_{ji}v_i + U_{jy} + b_j\right) \tag{6}$$

where $\sigma(a)$ is a logistic sigmoid function defined by $1/(1 + \exp(-a))$. It is noted that $P(h_j = 1|\mathbf{v}, \mathbf{y})$ can be used as features in supervised and semisupervised settings in some classification problems. Similarly, we have

$$P(v_i = 1|\mathbf{h}) = \sigma\left(\sum_j W_{ij}h_j + c_i\right) \tag{7}$$

$$P(y_k|\mathbf{h}) = \frac{e^{d_{y_k} + \sum_j U_{jy_k}h_j}}{\sum_l y_l e^{d_{y_l} + \sum_j U_{jy_l}h_j}} \tag{8}$$

Eq. (8) is a softmax function, which can be used directly to calculate the probability of the input data belonging to class k. In our system, the input data is a binary vector and we use a logistic function as a empirical distribution of the training data. The log-likelihood gradient for the parameters $\theta = \{\mathbf{W}, \mathbf{U}, \mathbf{b}, \mathbf{c}, \mathbf{d}\}$ for any RBMs can be written as

$$\frac{\partial}{\partial\theta}\log P(\mathbf{v}, \mathbf{y}) = E_{\mathbf{h}\sim P_\theta(\mathbf{h}|\mathbf{v}_i, y_i)}\left[-\frac{\partial}{\partial\theta}E(\mathbf{h}, \mathbf{v}_i, y_i)\right] - E_{\mathbf{v},\mathbf{y},\mathbf{h}\sim P_\theta(\mathbf{h},\mathbf{v},\mathbf{y})}\left[-\frac{\partial}{\partial\theta}E(\mathbf{h}, \mathbf{v}, \mathbf{y})\right] \tag{9}$$

In (9), the first and second terms are the expectations under $P_\theta(\mathbf{h}|\mathbf{v}_i, y_i)$ (i.e., the posterior of $\mathbf{h}$ given the training samples $(\mathbf{v}_i, y_i)$) and $P_\theta(\mathbf{h}, \mathbf{v}, \mathbf{y})$, respectively. They are referred as the positive and the negative phases, respectively [36]. In the positive phase, $\mathbf{v}$ is clamped to the observed input data and we can sample the hidden unit $\mathbf{h}$ given $\mathbf{v}$. However, the negative phase is normally intractable. In general, we can use the well-known contrastive divergence (CD)-based one-step Gibbs sampling to estimate the negative phase. CD is an approximation of the log-likelihood gradient that has been found to be a successful update rule for training RBMs [36].

In the classification stage, the target units are the classes of keypoints, which have been extracted from a preselected keyframe. When given an input $\mathbf{v}$, we first use (6) to compute $P(\mathbf{h}|\mathbf{v}, \mathbf{y})$ and then sample $\mathbf{h}$ from this distribution. Next, we use (8) to compute $P(y_k|\mathbf{h})$ that determines the probability of $\mathbf{v}$ belonging to class k. Alternatively, we can compute the marginal $P(\mathbf{y}|\mathbf{v})$ to predict target variables.

III. Real-Time Keypoint Recognition Using RBM

A. Overview

Similar to the method in [18] and [19], the keypoint recognition in this research is also formulated as a classification problem by treating the set of all the possible appearances of each individual keypoint as a point class. It includes two stages, i.e., offline learning stage and online keypoint recognition stage. Fig. 2 shows the block diagrams of the proposed RBM-based method. During the offline stage, keypoints will be first extracted from a preselected keyframe using a fast multiscale extraction method [19]. Similar to the methods in [18]–[21], we also use an image warping technique to generate

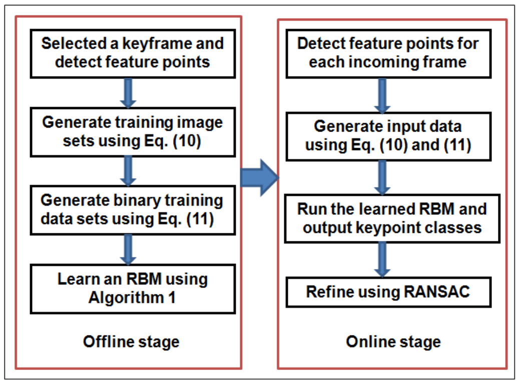
Selected a keyframe and detect feature points
Generate training image sets using Eq. (10)
Generate binary training data sets using Eq. (11)
Learn an RBM using Algorithm 1
Offline stage
Detect feature points for each incoming frame
Generate input data using Eq. (10) and (11)
Run the learned RBM and output keypoint classes
Refine using RANSAC
Online stage

Fig. 2. Block diagrams of the keypoint recognition method.

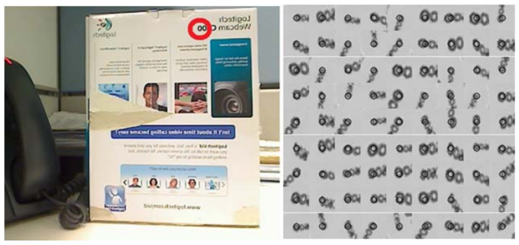

Fig. 3. Example of generating training image patches. Left: selected keyframe. Right: set of image patches generated using (10) corresponding to the feature point extracted from the keyframe, as shown in the red circle.

training image patches for each keypoint and generate the binary training data sets by conducting pixel intensity comparisons of several predefined pixels. The major difference lies in that we use an RBM to learn the training data and to recognize the keypoints in real-time during the tracking stage.

B. Generating Training Image Sets

The first task is to generate training image sets. We approximate image patches surrounding each keypoint as locally planar structures and their distortions under perspective projection as homographies. Given a training image and its K keypoints, denoted as $M = \{m_1, m_2, \ldots, m_K\}$, numerous possible appearances (i.e, represented by 16×16 image patches) of each keypoint m_i are generated using random affine transformations. An affine transformation can be decomposed as

$$\mathbf{T} = \mathbf{R}_\sigma \mathbf{R}_\phi^{-1} \mathbf{S} \mathbf{R}_\phi \tag{10}$$

where $\mathbf{R}_\sigma$ and $\mathbf{R}_\phi$ are the two rotation matrices that are parameterized by the angles σ and ϕ, and $\mathbf{S} = \begin{vmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{vmatrix}$ is a scaling matrix. σ and ϕ are randomly drawn from $[0, 2\pi]$. λ_1 and λ_2 are from $[0.5, 1.5]$. This range of scales and rotations is sufficient to randomly generate various image patches under wide variations of viewpoints and scales. The right image in Fig. 3 shows an image patch set of one interest keypoint extracted from the right image.

C. Generating Training Data Sets

After randomly generating the training image patch set, the real input data (the observed data) used to train an RBM model is defined by a n-dimensional binary vector based on the difference of the intensities of the two pixels $d_1(i)$ and $d_2(i)$ in the training image patch sets. $d_1(i)$ and $d_2(i)$ have been randomly preselected. Given $2 \times n$ pixels that are randomly preselected and denoted as $d_1(n)$ and $d_2(n)$, the input data, i.e., the visible unit is defined as follows:

$$v = \begin{cases} 1, & \text{if } I_p(d_{1i}) \geq I_p(d_{2i}) \\ 0, & \text{otherwise} \end{cases} \quad (i = 1, \ldots, n) \tag{11}$$

where $I_p(\cdot)$ is the intensity of an image patch at a pixel. n is the dimension of the input data. The n-D binary vector obtained using (11) exhibits spatial coherence [18], [19]. Thus, each binary vector is suitable to be represented as an input for learning an RBM where the empirical data distribution can be easily approximated as a logistic distribution.

Algorithm 1 RBM Learning Procedure

```
$\mathbf{v}^1$ and $\mathbf{y}^1$ are the samples from the training distribution,
$\lambda$ is a learning rate for the stochastic gradient descent in CD-1.
The positive phase:
• Compute $P(h_j^1 = 1|\mathbf{v}^1, \mathbf{y}^1)$ using Eq. (6) $(j = 1, \ldots, M)$
• Sample $\mathbf{h}^1$ from $P(\mathbf{h}^1|\mathbf{v}^1, \mathbf{y}^1)$
The negative phase:
• Compute $P(v_i^2 = 1|\mathbf{h}^1)$ using (7) $(i = 1, \ldots, D)$
• Sample $\mathbf{v}^2$ from $P(\mathbf{v}^2|\mathbf{h}^1)$
• Compute $P(y_k^2 = 1|\mathbf{h}^1)$ using (8) $(k = 1, \ldots, K)$
• Sample $\mathbf{y}^2$ from $P(\mathbf{y}^2|\mathbf{h}^1)$
• Compute $P(h_j^2 = 1|\mathbf{v}^2, \mathbf{y}^2)$ using (6) $(j = 1, \ldots, M)$
• Sample $\mathbf{h}^2$ from $P(\mathbf{h}^2|\mathbf{v}^2, \mathbf{y}^2)$
• Update using (9):
$\theta \leftarrow \theta - \lambda(\frac{\partial}{\partial\theta}E(\mathbf{h}^1, \mathbf{v}^1, \mathbf{y}^1) - \frac{\partial}{\partial\theta}E(\mathbf{h}^2, \mathbf{v}^2, \mathbf{y}^2))$
```

D. RBM-Based Keypoint Recognition

After generating training data sets and their corresponding point class labels, we use an RBM to model the statistical structures of these training data sets. Algorithm 1 is used to estimate $\theta = \{\mathbf{W}, \mathbf{U}, \mathbf{b}, \mathbf{c}, \mathbf{d}\}$ of an RBM model. The learned weights represent the statistical distributions of the inputs. By having more hidden units, we can increase the modeling capacity of the RBM.

During the online stage, given a vector of a keypoint obtained using (10) and (11), we first use (6) to derive the corresponding hidden variables $\mathbf{h}$. Next, we apply (8) to compute $P(y_k|\mathbf{h})$. Finally, we obtain the corresponding keypoint class k of this feature point using $k := \arg\max_k p(y_k|\mathbf{h})$. Alternatively, we can compute the marginal $P(\mathbf{y}|\mathbf{v})$ to predict target variables.

IV. Experimental Result

A. Result

We have conducted four experiments on a 2.53-GHz Intel Xeon(R) with 12-GB RAM, including two indoor and two outdoor experiments to validate the effectiveness and the generalization of the proposed method. We set the learning rate to 0.1, momentum to 0.5, epochs to 30, batch size to 100, and weight decay factor to 0.0004, respectively. In our current implementation, the number of the hidden units was set to 100 in the training and tracking stages. The epochs can be large as 300, which will normally result in better matching performance. However, a larger number of epochs requires more training time. The resolution of the image sequences of the two indoor and the first outdoor experiments were 640×480 and the second outdoor experiment was 640×360. In all the experiments, we set the number of keypoints to be extracted as 400 and used the multiscale keypoint extraction method in [19] to extract the keypoints. 800 training image patches were generated for each keypoint using (10).

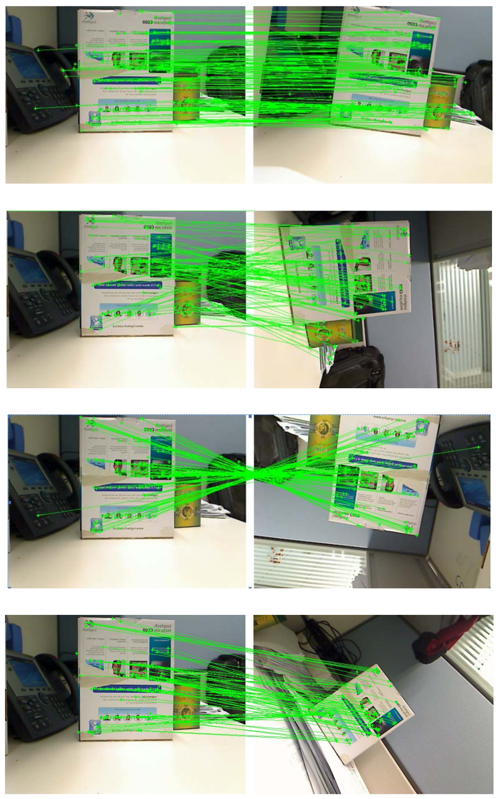
Fig. 4. Matching results of the first indoor experiment.

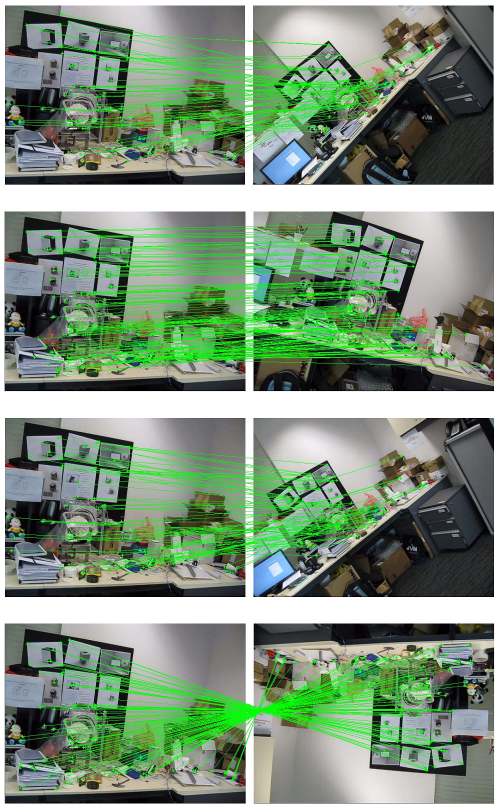
Fig. 5. Matching results of the second indoor experiment.

Eq. (11) was then used to generate binary training data used for learning an RBM. The dimension of the training data sets (i.e., n in (11) was 160. An RBM was learned for each experiment using the

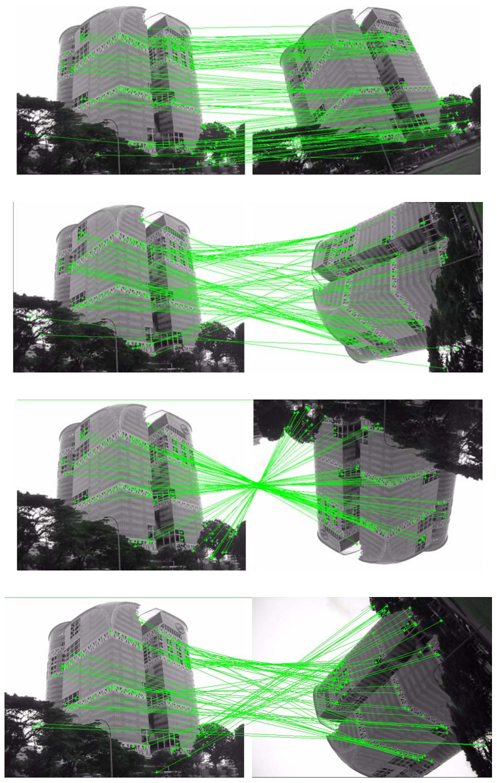
Fig. 6. Matching results of the first outdoor experiment.

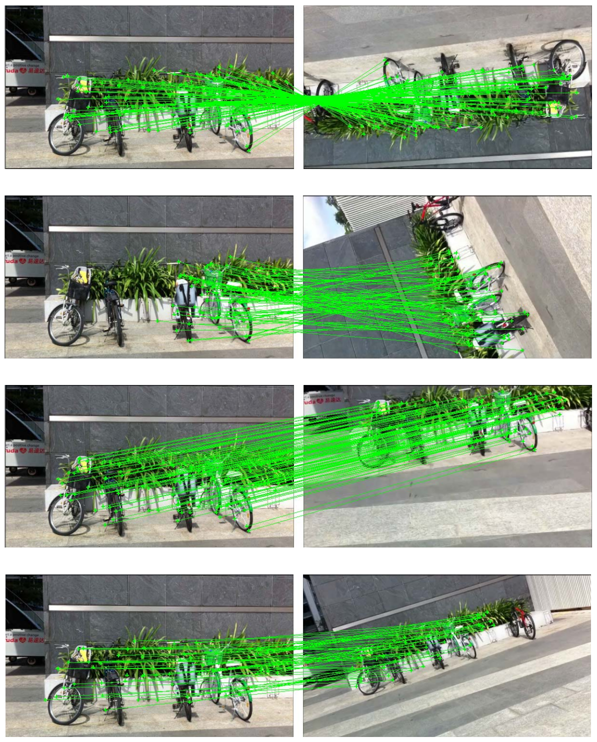
Fig. 7. Matching results of the second outdoor experiment.

binary training data sets and the corresponding class labels obtained from this selected keyframe. During the tracking stage, we used the learned RBM model to establish point correspondences between the keyframe and each incoming frame. Finally, we refined the point correspondences using RANSAC technique. Some of the matching results are shown in Figs. 4–7. In these experiments, the left image in each pair is the training image and the right is the incoming image. These experiments show that point matches can be reliably established under widely varying camera poses.

As mentioned in [5], [7], and [19], the classification-based keypoint recognition method can be used for AR [19], object recognition [5], and vision-based simultaneous localization and mapping [7]. The proposed method can be employed in applications that require real-time performance, such as AR. It can be used for localizing texture objects, where virtual objects can be superimposed without any predefined markers. In the following section, we will analyze the performance of the proposed method.

B. Performance Analysis

To analyze the performance of our method, we first compared our method with the L&F method. We set the same experimental conditions for both methods, as described below.

1) The dimension of the input binary data to learn an RBM was 160 while in the L&F method, we set 16 classifiers each with 10 preselected intensity pixel comparisons.
2) The number of keypoints to be learned for both methods was 400.
3) The same RANSAC method was used to refine the final results for both methods.

Figs. 8 and 9 show the comparison results. For the purpose of demonstration, 103 images have been sampled from the image sequences. The results demonstrate that the recognition rates can be improved around 20% without sacrificing real-time performance. Additionally, the proposed RBM-based method can recognize many more keypoints than the L&F method for some images. For example, in the 45th image in the first outdoor experiment, we could obtain 74 point matches while only 35 matches were established using the L&F method.

The improvement on recognition rates is at the cost of more training time, which depends on the numbers of point classes, epochs and samples. Fig. 10 shows the average training time with the numbers of the point classes for the RBM method. It can be observed that the training time is scaled approximately linearly with respect to the number of the point classes. On the other hand, by increasing the number of point classes, the system can yield more matches. Fig. 11 shows an example of the matching results on the 99th image in the second indoor experiment using 200 and 400 point classes. The training time for 200 and 400 classes are about 203 s and 559 s, respectively. Under the same conditions, the training time of the RBM method is approximately double that of the L&F method. Computation time is often critical to many real-time applications, such as in AR and vision-based robot navigation. The average computation time for each frame is less than 0.06 s. Table I shows the CPU time for each frame including the three time-consuming stages, i.e., feature detection, RBM classifier, and RANSAC refinement. Thus, both the proposed method and the L&F method are able to achieve real-time performance. It should be noted that in our current system, we have only used CPU to implement our RBM-based keypoint recognition. However, because most of the computation time to estimate an RBM model is attributed to matrix multiplication, GPU can be used to further improve the system speed.

Different batch sizes will affect the performance of the proposed algorithm. We have randomly sampled two images (i.e., 69th and 90th) from the second indoor and outdoor experiments to analyze the training time and recognition performance. Figs. 12 and 13 show the average training time and recognized numbers of the keypoints with the corresponding batch sizes. It can be observed that when the batch size is from 40 to 120, the training time significantly decreases and the numbers of the recognized keypoints also slightly decrease. However, when the batch size is larger than a certain value, the batch sizes do not obviously affect the training time and the numbers of the recognized keypoints. The batch size was chosen as 100 in our current system.

To further demonstrate the advantages of the proposed RBM method, we have compared our results with those obtained using the SIFT method (kindly provided in [27]). 400 point classes were learned for the RBM method while the same number of keypoints were extracted and matched using the nearest-neighbor k-D tree search for the SIFT approach. The matches were finally refined using RANSAC technique. As shown in Figs. 14 and 15, both methods yielded similar numbers of matches. However, our method can achieve real-time performance while the SIFT method cannot. Furthermore, we also compared our method with the SURF method. We used OpenCV(version 2.4.3) to test computation time of the SURF method on the same PC where we run our RBM method. The computation time to localize and match 400 feature points for each image (randomly selected from the images used in our experiments) is about 2.0 s. It is hardly able to meet real-time performance on the image sequences used in our experiments.

V. Discussion

There are several factors that will affect the performance of the RBM-based keypoint recognition method. First, the modeling capacity of an RBM can be increased by having more hidden units. It remains an open problem to determine the number of hidden units required for optimal keypoint recognition. Second, the number of training samples will also affect the recognition performance. Practically, these factors depend on the requirements of the real applications. We need to investigate the joint space of the dimension of the hidden units, the learning rate, the number of the training iterations and the number of the training samples to achieve the best recognition performance.

As reported in the literature, RBMs have been shown to be very successful in dimension reduction and classification [24]. Several variants of RBMs have been explored and employed in real applications. For example, a sparse RBM is a sparse distributed representation motivated by the fact that neurons in the cortex are believed to have a distributed and sparse representation. Larochelle and Bengio [25] proposed a discriminative RBM (DRBM) framework, which can be used to derive an efficient classifier. The major difference between a generative RBM and a DRBM is that a generative RBM is learned by optimizing a joint distribution $P(\mathbf{v}, \mathbf{y})$, while a DRBM is learned by optimizing $P(\mathbf{y}|\mathbf{v})$. As investigated in [35], the advantage brought by discriminative or generative training usually depends on the amount of available training data. Smaller training sets tend to favor generative learning while bigger ones favor discriminative learning. For real-time keypoint recognition, it should be further explored, which one is more suitable. However, as suggested in [25] and [35], a hybrid classifier might be a better choice, which can enjoy the best properties of both sides. We would like to argue that RBMs and their variants might be able to be combined with existing learning methods for image descriptors to yield better matching performance. For example, RBMs might be able to replace the LDA and Powell minimization for learning local image descriptors constructed by a set of building blocks in [17].

Furthermore, a new technique called deep learning [36] has been extensively explored and had been found more powerful than a single RBM in real applications, such as information retrieval, face

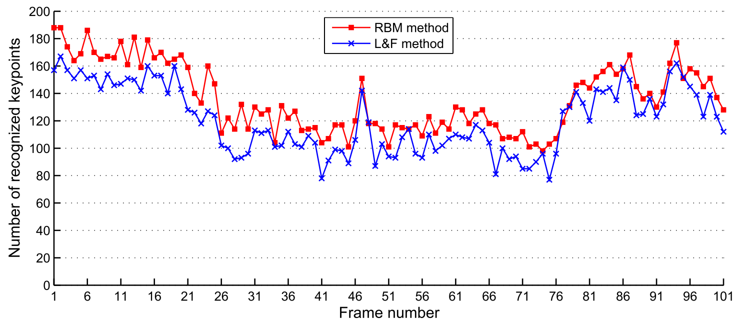

Fig. 8. Comparison results using the RBM and the L&F methods for the first indoor experiment.

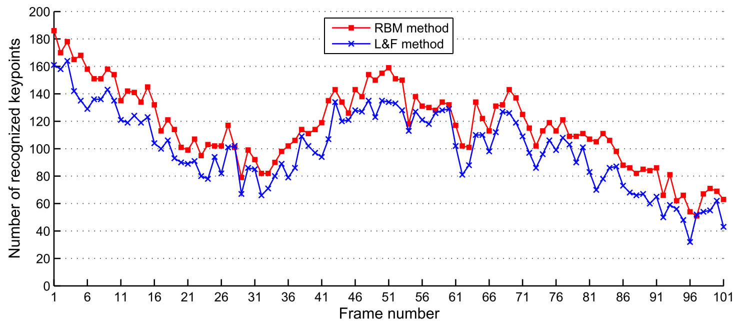

Fig. 9. Comparison results using the RBM and the L&F methods for the second outdoor experiment.

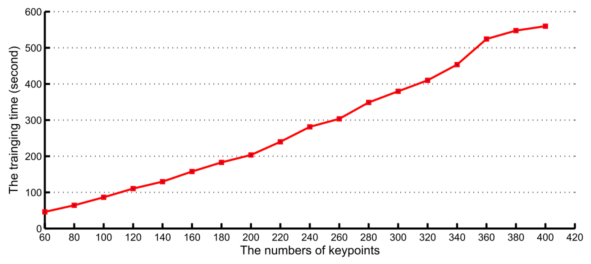

Fig. 10. Average training time with different numbers of keypoint classes.

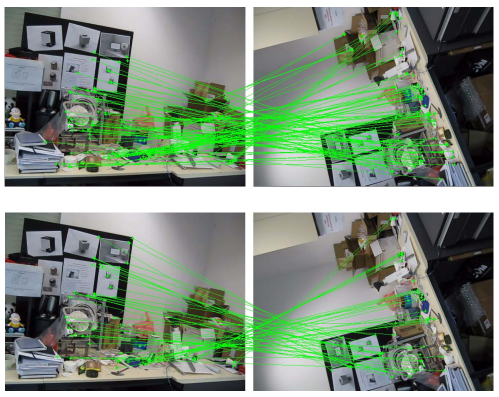

Fig. 11. Matching results using different numbers of point classes: 400 (top) and 200 (bottom).

TABLE I
RUNNING TIME PERFORMANCE (SECOND)

Feature Detection	RBM Classifier	RANSAC
0.022	0.030	0.004

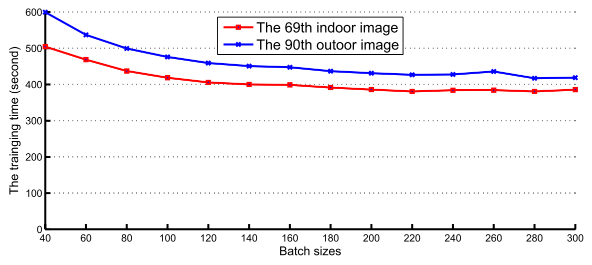

Fig. 12. Average training time with different batch sizes.

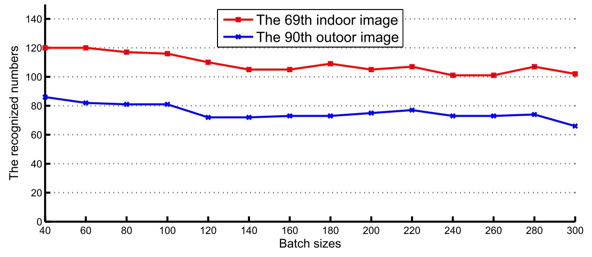

Fig. 13. Recognized keypoints with different batch sizes.

verification, and so on. Deep belief networks (DBNs) are one of the most representative deep learning techniques, which can model the statistical structures of the input data layer by layer [24], [34]. In a DBN, the joint of the top two layers is an RBM and each lower hidden layer is also learned as an RBM. After training the first hidden layer, the hidden units of the first hidden layer (i.e., the learned features) then become the input units for learning the next RBM. This layer-wise learning procedure can be repeated, depending on the

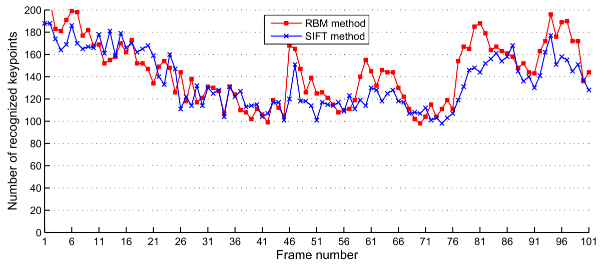

Fig. 14. Comparison results using the RBM and the SIFT methods for the indoor experiment.

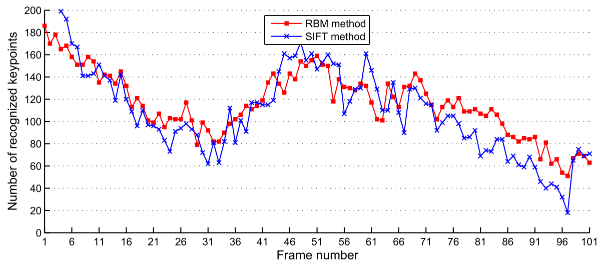

Fig. 15. Comparison results using the RBM and the SIFT methods for the outdoor experiment.

requirements of related applications. Finally, a global fine-tuning strategy can be applied to obtain the optimal weights using back-propagation of error derivatives. In future, we will explore the keypoint recognition problem using a DBN model. The major contribution in this brief is to provide a way that RBM, its variants and deep architecture (such as DBN) can be used to achieve better keypoint recognition rates without sacrificing the real-time performance.

VI. Conclusion

In this brief, we presented a real-time keypoint recognition method using RBM. An RBM is used to model the statistical structures of the training data sets that are able to achieve better keypoint recognition rates without sacrificing the real-time performance. Several experiments have been conducted to demonstrate the effectiveness of the proposed approach, and it can be used for applications that require real-time performance, such as AR and robot navigation. In future, we are interested in exploring fast online RBM learning methods so that more keypoints from more than two keyframes can be recognized in real-time.

References

[1] I. Gordon and D. G. Lowe, "Scene modeling, recognition and tracking with invariant image features," in *Proc. 3rd IEEE Int. Symp. Mixed Augmented Real.*, Nov. 2004, pp. 110–119.

[2] V. Ferrari, T. Tuytelaars, and L. Van Gool, "Markerless, augmented reality with a real-time affine region tracker," in *Proc. IEEE Int. Symp. Mixed Augmented Real.*, Oct. 2001, pp. 87–96.

[3] D. Tell, "Wide baseline matching with applications to visual servoing," Ph.D. dissertation, Dept. Numer. Anal. Comput. Sci., Royal Inst. Technol., Stockholm, Sweden, 2002.

[4] T. Tuytelaars, L. van Gool, L. D'Haene, and R. Koch, "Matching of affinely invariant regions for visual servoing," in *Proc. IEEE Conf. Robot. Autom.*, May 1999, pp. 1601–1606.

[5] J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[6] T. Goedeme, T. Tuytelaars, and L. Van Gool, "Fast wide baseline matching for visual navigation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2004, pp. 24–29.

[7] B. Williams, G. Klein, and I. Reid, "Real-time SLAM relocalisation," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, 2007, pp. 1–8.

[8] Z. Zhang, R. Deriche, O. Faugeras, and Q. T. Luong, "Robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," *Artif. Intell.*, vol. 78, nos. 1–2, pp. 87–119, Oct. 1995.

[9] P. Pritchett and A. Zisserman, "Wide baseline stereo matching," in *Proc. 6th Int. Conf. Comput. Vis.*, 1998, pp. 863–869.

[10] C. Schmid and R. Mohr, "Local grayvalue invariants for image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 5, pp. 530–535, May 1997.

[11] A. Baumberg, "Reliable feature matching across widely separated views," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2000, pp. 774–781.

[12] F. Mindru, T. Moons, and L. van Gool, "Recognizing color patterns irrespective of viewpoint and illumination," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 1999, pp. 368–373.

[13] M. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *Proc. 8th IEEE Int. Conf. Comput. Vis.*, Jul. 2001, pp. 525–531.

[14] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[15] Y. Ke and R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jul. 2004, pp. 506–513.

[16] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *Proc. 9th Eur. Conf. Comput. Vis.*, May 2006, pp. 404–417.

[17] M. Brown, G. Hua, and S. Winder, "Discriminant learning of local image descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 43–57, Jan. 2011.

[18] V. Lepetit, J. Pilet, and P. Fua, "Point matching as a classification problem for fast and robust object pose estimation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jul. 2004, pp. 244–250.

[19] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1465–1497, Sep. 2006.

[20] M. Ozuysal, V. Lepetit, F. Fleuret, and P. Fua, "Feature harvesting for tracking-by-detection," in *Proc. 9th Eur. Conf. Comput. Vis.*, 2006, pp. 592–605.

[21] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua, "Fast keypoint recognition using random ferns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 3, pp. 448–461, Mar. 2010.

[22] G. Helmut, G. Michael, and B. Horst, "Real-time tracking via on-line boosting," in *Proc. Brit. Mach. Vis. Conf.*, 2006, pp. 47–56.

[23] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," in *Parallel Distributed processing*, vol. 1, D. Rumelhart and J. McClelland, Eds. Cambridge, MA, USA: MIT Press, 1986, pp. 194–281, ch. 6.

[24] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[25] H. Larochelle and Y. Bengio, "Classification using discriminitive restricted Boltzmann machines," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 536–543.

[26] Y. LeCun and C. Cortes. (2012, Aug. 9). *The Mnist Database of Handwritten Digits*, New York University, New York, NY, USA [Online]. Available: http://yann.lecun.cm/exdb/mnist

[27] R. Hess. (2012, Aug. 9). *The Scale Invariant Feature Transform Source Codes*, Oregon State University, Corvallis, OR, USA [Online]. Available: http://robwhess.github.com/opensift/

[28] G. B. Huang, H. Lee, and E. L. Miller, "Learning hierarchical representations for face verification with convolutional deep belief networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2518–2525.

[29] M. Welling, M. Rosen-Zvi, and G. E. Hinton, "Exponential family harmoniums with an application to information retrieval," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 1481–1488.

[30] G. Taylor, G. E. Hinton, and S. Roweis, "Modeling human motion using binary latent variables," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1345–1352.

[31] P. V. Gehler, A. D. Holub, and M. Welling, "The rate adapting poisson model for information retrieval and object recognition," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 337–344.

[32] L. L. Daniel and P. Chow, "High-performance reconfigurable hardware architecture for restricted boltzmann machines," *IEEE Trans. Neural Netw.*, vol. 21, no. 11, pp. 1780–1792, Nov. 2010.

[33] H. Tang, V. A. Shim, K. C. Tan, and J. Y. Chia, "Restricted Boltzmann machine based algorithm for multi-objective optimization," in *Proc. IEEE CEC*, Jul. 2010, pp. 3958–3965.

[34] A. Stuhlsatz, J. Lippel, and T. Zielke, "Feature extraction with deep neural networks by a generalized discriminant analysis," *IEEE Trans. Neural Netw.*, vol. 23, no. 4, pp. 596–607, Apr. 2012.

[35] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 841–848.

[36] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–27, 2009.