Send Orders for Reprints to reprints@benthamscience.ae

The Open Automation and Control Systems Journal, 2015, *7*, 1-7

Open Access

Spatio-temporal Feature Classifier

Yun Wang[1,*] and Suxing Liu[2]

[1]*School of Mechanical and Electrical Engineering, Xingxiang University. Henan, Xinxiang, 4530003, China*

[2]*Henan Electronic Commerce Association, Henan Province, Zhengzhou, 450004, China*

Abstract: Different from current foreground and background segmentation methods, we did not utilize the low level image representation method (such as boundaries and textures) to extract the feature of the videos, instead we proposed a spatio-temporal feature classifier to obtain the union region of object from natural videos as the interest points. We slide the temporal chunk along time axis to obtain samples from videos, and train the Support Vector Machine (SVM) with feature vectors. Then we built a spatio-temporal feature classifier and tested our algorithm on the most popular benchmark dataset. Experiment results showed the effectiveness and robustness of the algorithm.

Keywords: Classifier, Feature extraction, Spatio-temporal.

1. INTRODUCTION

Most methods in feature extraction research were based on fame segmentation, appearance cues and motion constraints were extracted [1, 2]. Lee *et al.* proposed to use spectral graph clustering to detect and segment the primary object based on multiple binary inlier/outlier partitions [3]. However, clustering process cannot model the evolution of object's shape and spatial–temporal arrangement. Ma and Latecki's method attempts to solve this issue by building a model of object region as a constrained Maximum Weight Cliques [4]. They tried to locate the object region from all the video frames simultaneously using this model. However, the model belongs to NP-hard issue and an approximate optimization technique is needed.

Many traditional approaches for behavior recognition are based on 2D/3D tracking models, which heavily rely on tracked features. One class of work uses a tracked feature or object, and its time series can be used as a descriptor in a recognition task. Another class of approaches utilizes a number of spatial features. Spatial arrangements of tracked points and view invariant aspects of the trajectory were used. The most vital part of this framework lies in tracked contours. While it is a practical difficult problem of feature and contour tracking, frame-by-frame recognition using a hand labeled dataset was introduced. While the assumptions of edge detection are less restrictive than those of feature or contour tracking, it is still unreliable when the object of interest has poor contrast.

To extract features of object regions from videos, many methods apply motion analysis of point trajectories as a reasonably robust tool. These approaches are based on the fact that motion estimation requires structures matching. However, there are no such structures in homogeneous areas of the image. It is reasonable to obtain dense trajectories from dense optical flow, but trajectories in homogenous areas are less, which resulting in sparse point trajectories.

Among the state-of the art methods for extracting feature points, Cuboids is a spatio-temporal interest point detector [5]. It is designed to detect dense features [6]. It improves the efficiency of a robust behavior descriptor [7]. By using the spatio-temporal feature points, Cuboids can characterize behavior. A spatiotemporal feature is derived from a short local video sequence. A behavior is then fully represented in terms of the types and locations of feature points. The complexity of distinguishing two behaviors is determined by the detection and description of a rich set of features.

Although interest points extracted by Cuboids are a number of sample behaviors, we noticed that layout of the features represented by the detected interest points is sparse, which will cause the insufficient information of actions were compiled into features. In addition, if the features extracted were used for action recognition and machine learning, interest points extracted by Cuboids usually cover the area of background content which has no contribution to action recognition of the object [8-11]. Furthermore, interest points extracted by Cuboids are also unreliable where the region has poor contrast. As a result, it will lead the performance decrease of action recognition.

2. ALGORITHM

We propose a spatio-temporal feature classifier to obtain the union region of object from natural videos as the interest points. Our aim is to recognize the object from natural scenes and define the interest points as the pixels of object. Our method is different from current methods for detecting features from videos. We build an explicit model of

*Address correspondence to this author at the School of Mechanical and Electrical Engineering, Xingxiang University. Henan, Xinxiang, 4530003, China Tel: (+86-0373)3682637; E-mail: xw430@126.com

1874-4443/15

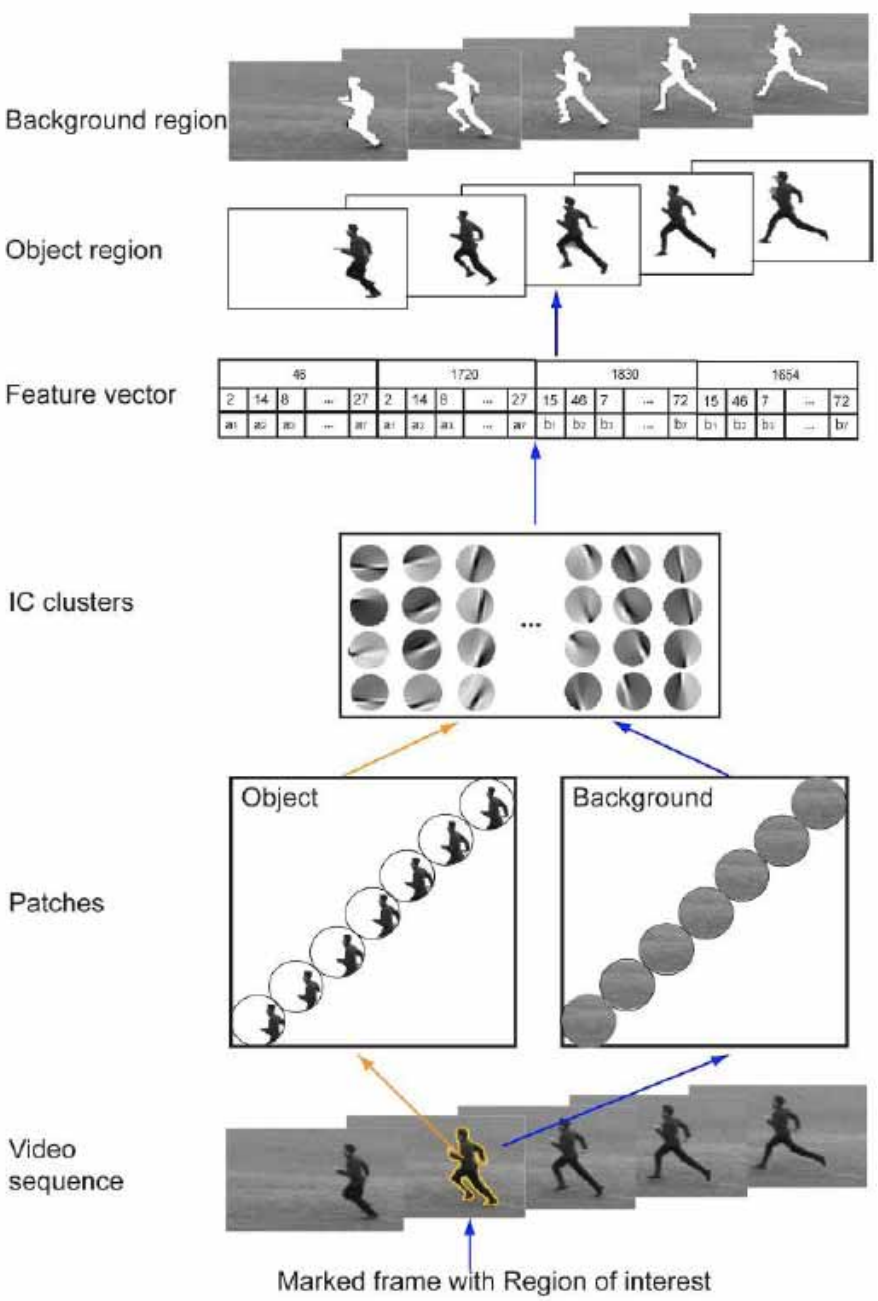

Fig. (1). Spatio-temporal classifier algorithm.

describing moving object in the videos, and therefore the segments usually correspond to a particular object that exhibit coherent appearance or motion. Our model also utilize sliding window through all the frames in the video, which will cover the global features in the video. The detailed steps are illustrated in Fig. (**1**).

Step 1: Divide all the testing videos into chunks of sequences same length in time.

Step 2: In each chunk of sequence, uniformly sample some key frames.

Step 3: Mark the sampled key frames with regions of interest points as pixels cover the objects content by image segmentation tools.

Step 4: Slide the chunk and perform the 3rd step until it reaches the end of the testing videos. Collect and compute the union region of interest points.

Step 5: Densely sample circular patches at the union region of interest points. And sample the same amount of circular patches besides the union region of interest points, which can be deemed as background regions.

Step 6: Perform Independent Component Analysis (ICA) on the sampled patch sequences of the object and obtain the Independent Components (ICs).

Step 7: Map both the patch sequences from region of interest points and background regions to the IC clusters and obtain the corresponding feature vectors.

Step 8: Train the Support Vector Machine (SVM) with feature vectors obtained above. Split the feature vectors into training and testing data, and feed the SVM using training data and using testing data to examine its recognition accuracy. Adopt the cross-validation strategy to optimize the parameters of model. And then build the model for object recognition. The model is able to recognize the object from natural scenes.

Step 9: In the remaining frames of each chunk, uniformly sample circular patches for each frame at regions of object and background. Map the IC clusters and obtain the corresponding feature vectors as data for recognition.

Step 10: Feed the SVM model with these data to classify them into object and background. Automatically mark the regions of object and background.

Step 11: For all the testing videos including every action category, collect the union region of interest points as the final output.

2.1. Support Vector Machines

Multiclass support vector machines (SVMs) are used to classify actions. SVMs are supervised learning models by analyzing data and recognize patterns. SVMs are inherently two-class classifiers. Since the test dataset usually contains more than two classes, multiclass SVM is adopted.

By using support vector machines, SVM aims to assign labels drawn from a finite set of several elements. SVM uses the multi-class formulation, but optimizes it with an algorithm that is very fast in the linear case.

We adopt the one-versus-rest classifiers. A single classifier is trained per class to distinguish that class from all other classes. Prediction is then performed by using each binary classifier, and choosing the prediction with the highest confidence score.

In generally, SVM uses an algorithm based on Structural SVMs. We use the LIBSVM implementation of SVM.

2.2. Cross-validation: Evaluating Classifier Performance

In the process of building SVM models, optimal parameters should be set. Over-fitting from an algorithm has inferred too much from the available training samples. In practice, the reason that SVMs tend to be resistant to over-fitting, it uses regularization. The key to avoid over-fitting lies in careful tuning of the regularization parameter, and in the case of non-linear SVMs, careful choice of kernel and tuning of the kernel parameters. This is best guarded against empirically by using a measure of the generalization ability of the model. Cross validation is one such popular method.

We use k-fold cross-validation to solve above issue. In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data.

3. IMPLEMENTATION

All the experiments are conducted using the implementation on MATLAB (8.2.0.701 (R2013b)) running on a Dell Optiplex 980 desktop.

We build a model by utilizing sliding window through all the frames in the video, which will cover the global features in the video. Slide the chunk in the testing videos. Collect and compute the union region of interest points. Then densely sample circular patches both at the union region of interest points and background. Perform ICA on all the sampled patches and obtain ICs. Map the circular patches to the IC clusters; compute the features of the sampled patches. Train the Support Vector Machine (SVM) with feature vectors obtained above. And feed the SVM model with these data to classify them into object and background. Automatically mark the regions of object and background.

We randomly selected 70% sample data for training and used the rest 30% sample data for testing. The classification accuracies are the averages of the accuracies obtained in 5 training-testing runs. In this procedure, we separated the training data into five folds, tested the model on a single fold using the remaining 4 folds to train the model, and repeated this procedure on each folds.

4. EXPERIMENTS

The model is tested on KTH human action dataset and Weizmann human motion dataset, as shown in Fig. (**2**) and Fig. (**3**) respectively.

4.1. Datasets of natural scenes

The KTH dataset contains 6 types of human actions are captured in four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes, and indoors. It is performed several times by 25 subjects, containing 6 kinds of actions (walking, jogging, running, boxing, hand waving, and hand clapping). The video sequences have a resolution of 160 × 120 pixels.

Fig. (**4**) shows some sample results of interest points for each frame in KTH. For each action in every dataset, the first row shows the original video clips. The second row shows the results containing both region of interest points and background, which are marked by green and yellow color respectively. Red points stand for errors. The third row shows the detected region of interest points corresponding to the frames at the first row.

The Weizmann human action dataset contains 81 low resolution (180 × 144 pixels) video sequences. It contains 83 video sequences performing 9 different actions: bending, jumping jack, jumping forward on two legs, jumping in place on two legs, running, galloping sideways, walking, waving one hand, and waving two hands. It was performed by nine different people. The figures were tracked and stabilized. Sample frames are shown in Fig. (**5**).

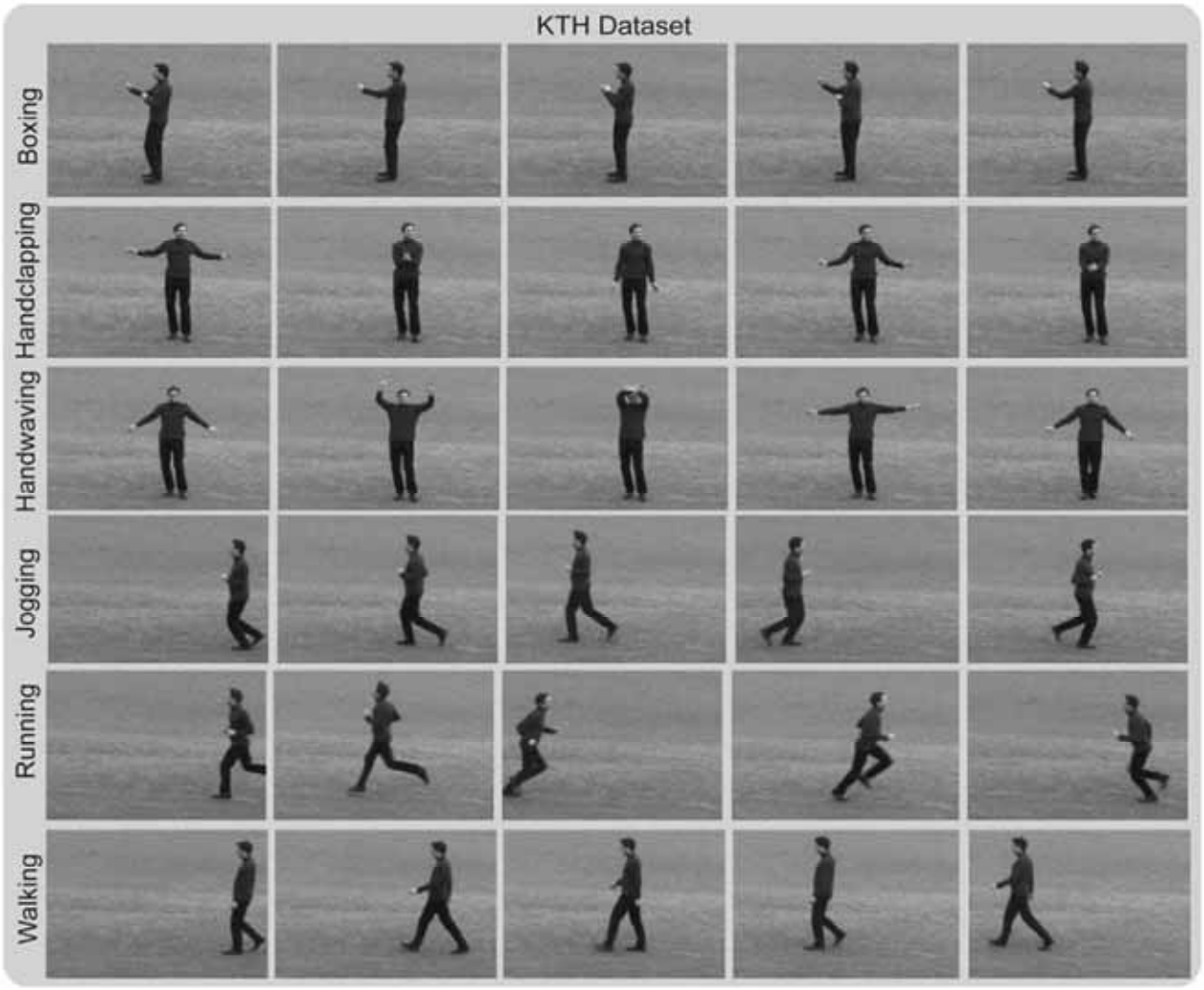

Fig. (2). KTH dataset.

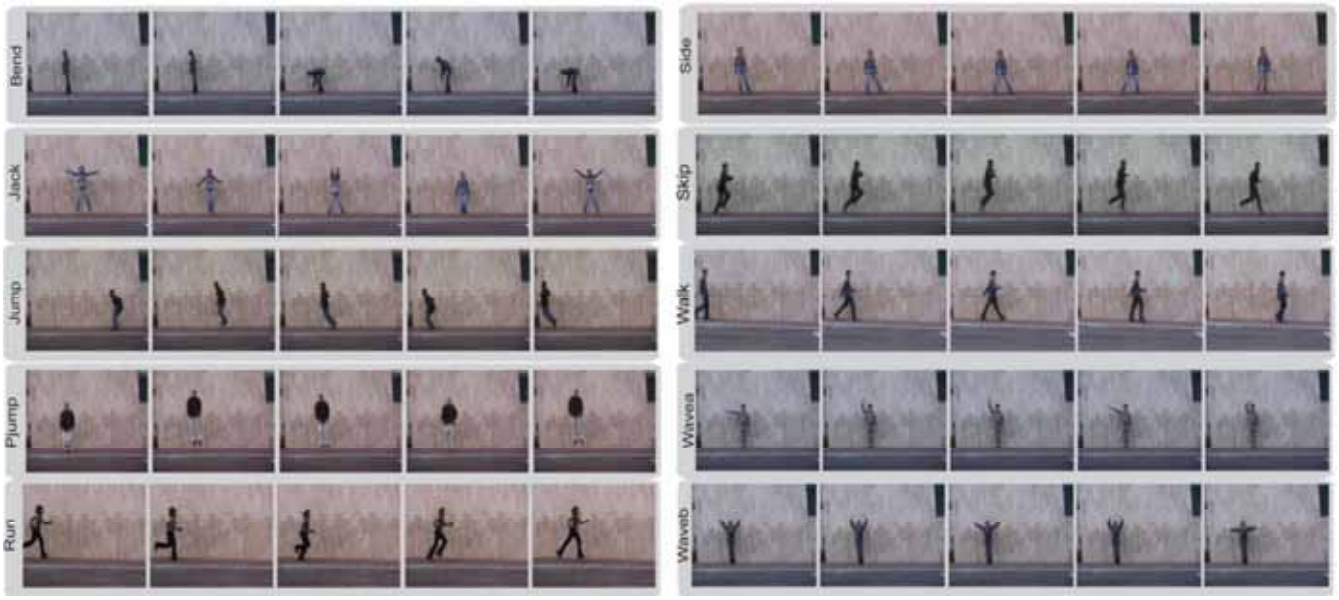

Fig. (3). Weizmann human action dataset.

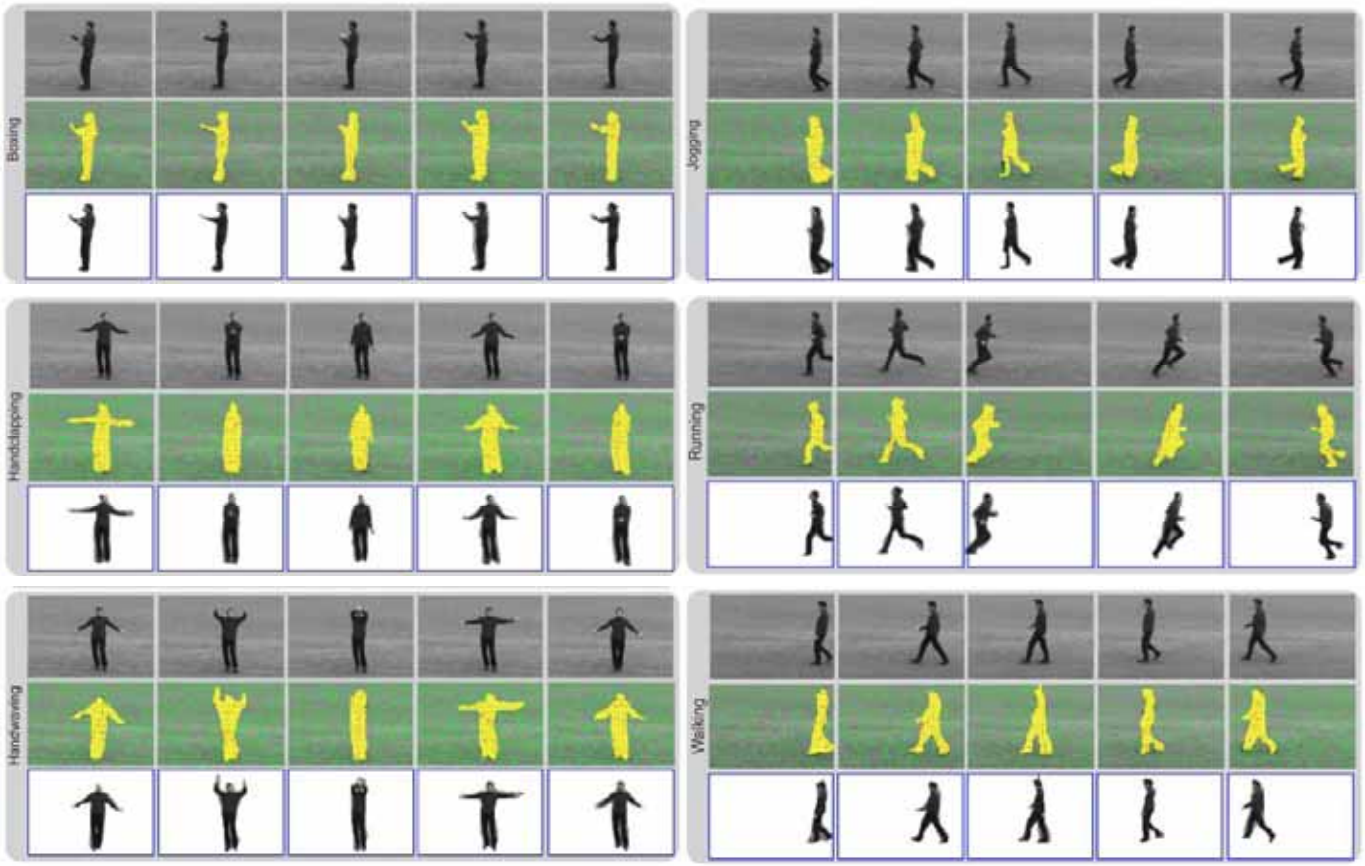

Fig. (4). Feature classification result of KTH dataset.

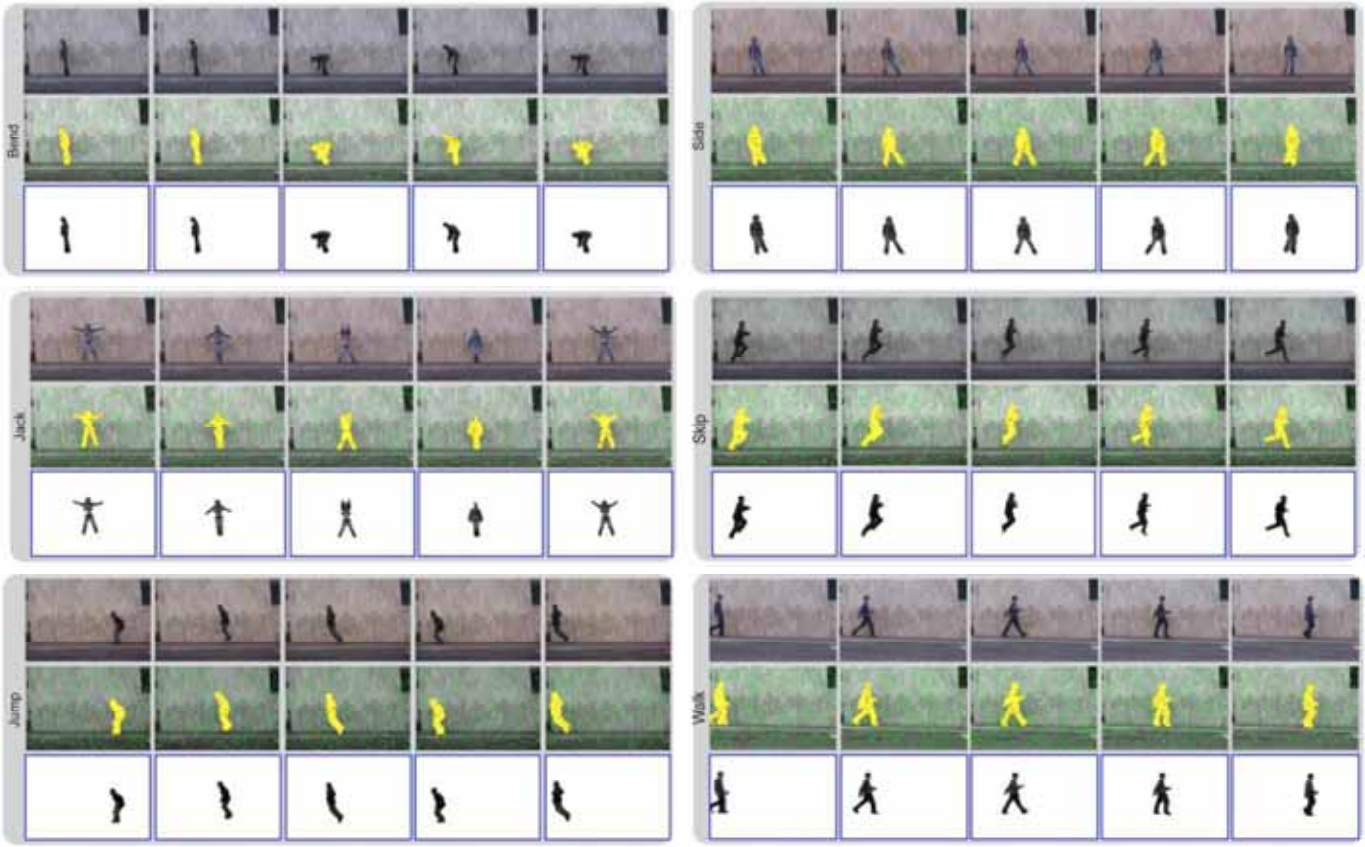

Fig. (**5**). Contd….

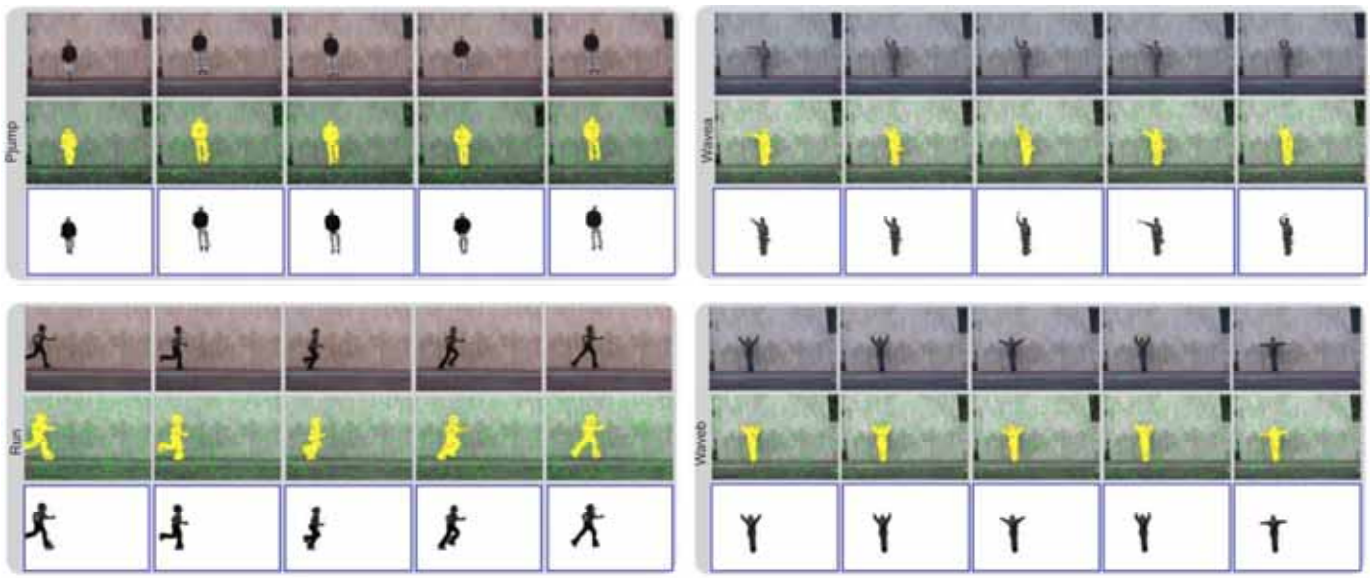

Fig. (5). Feature classification result of Weizmann dataset.

4.2. Effect of Parameter Variation

We examined the effect of parameter variation of our model. There are several vital parameters in our model. Patch size and scale length are direct parameters which can affect the performance of our model. Patch size is the radius of the circular patch in our model, the unit is pixel. Scale length is the number of frames in our spatio-temporal structure.

We test the different combinations of the two parameters. The results indicate that if the patch size is smaller and scale length is longer, the performance will be improved. However, the patch size should be an appropriate value which is related with the resolution of the test data, while scale length should cover the whole process of the action in local range.

It is worth noting that, using smaller patch size and longer scale length requires more memory and computational time. So there exists a trade-off between smaller these parameters and the performance of our model, our experiments showed that using relatively small patch size and reasonable scale length achieves acceptable results for action recognition.

4.3. Testing on KTH Dataset

We first build a spatio-temporal feature classifier to obtain the union region of object from natural videos as the interest points. For all the testing videos, we uniformly choose five adjacent key frames from chunks of sequences. Then we mark the chosen key frames with regions of interest points, and slide the chunk and perform the same process in all the testing videos to collect and compute the union region of interest points. We randomly sample 10×10^5 circular patches (radius is 5 pixels) at the union region of interest points, meanwhile sample the same amount of circular patches at background regions.

In the following, we perform ICA on the patch sequences of the union region and obtain 800 ICs. Finally, we build a Support Vector Machine (SVM) model with feature vectors by mapping both the patch sequences from region of interest points and background regions to the IC clusters. We use this SVM model to mark the regions of object and background automatically, and collect the union region of interest points as the final output. For the parameters of SVM classifier, we select 6,715 NASs, set Lc=3 and Nc =3, and use the 1-χ2 kernel in the SVM classifier (C = 0.125).

4.4. Testing on Weizmann Dataset

We perform similar process as the testing on KTH dataset. We first build a spatio-temporal feature classifier to obtain the union region of object from natural videos as the interest points. We also randomly sample 9×10^5 circular patches (radius is 6) at the union region of interest points, meanwhile sample the same amount of circular patches at background regions. In the following, we perform ICA on the patch sequences of the union region and obtain 1000 ICs. Finally, we use trained SVM model to mark the regions of object and background automatically, and collect the union region of interest points as the final output.

CONCLUSION

In this study, we proposed a spatio-temporal feature classifier to obtain the union region of object from natural videos as the interest points. We built a spatio-temporal feature classifier and tested our model on the KTH human action dataset and Weizmann dataset. Experiment results showed the effectiveness and robustness of the algorithm. We also discussed that the effect of parameters on our algorithm, we found that if the patch size is smaller and scale length is longer, the performance will be improved.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

This work was supported by the key scientific and technological project of Henan province technology department (Grant No. 142102210253).

REFERENCES

[1] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition *via* sparse spatio-temporal features", In: *2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2005, pp. 65-72.

[2] D.G. Lowe, "Distinctive image features from scale-invariant keypoints", *International journal of computer vision*, vol. 60, no. 2, pp. 91-110, 2004.

[3] A. Vazquez-Reina, S. Avidan, H. Pfister, and E. Miller, "Multiple hypothesis video segmentation from superpixel flows", In: *11th European Conference on Computer Vision*, 2010, pp. 268-281.

[4] Z. Dong, O. Javed, and M. Shah, "Video object segmentation through spatially accurate and temporally dense extraction of primary object regions", In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 628-635, 2013.

[5] Y.J. Lee, J. Kim, and K. Grauman, "Key-segments for video object segmentation", In: *IEEE International Conference on Computer Vision*, pp. 1995-2002, 2011.

[6] T. Ma, and L.J. Latecki, "Maximum weight cliques with mutex constraints for video object segmentation", In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 670-677, 2012.

[7] O. Barnichm and M. Van Droogenbroeck, "ViBe: A universal background subtraction algorithm for video sequences", *IEEE Trans. Image Processing*, vol. 20, no. 6, pp. 1709-1724, 2011.

[8] T. Wang and J. Collomosse, "Probabilistic motion diffusion of labeling priors for coherent video segmentation", *IEEE Trans. Multimedia*, vol. 14, no. 2, pp. 389-400, 2012.

[9] A. Levinshtein, A. Stere, K.N. Kutulakos, D.J. Fleet, S.J. Dickinson, and K. Siddiqi, "TurboPixels: Fast Superpixels Using Geometric Flows", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2290-2297, 2009.

[10] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1337-1342, 2003.

[11] L. Grady, "Random walks for image segmentation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1768-1783, 2006.

Received: June 18, 2014 Revised: November 05, 2014 Accepted: November 05, 2014