A Tree-based Approach to Integrated Action Localization, Recognition and Segmentation

Zhuolin Jiang[1], Zhe Lin[2], and Larry S. Davis[1]

[1]University of Maryland, College Park, MD, 20742
[2]Adobe Systems Incorporated, San Jose, CA, 95110
{zhuolin,lsd}@umiacs.umd.edu, zlin@adobe.com

Abstract. A tree-based approach to integrated action segmentation, localization and recognition is proposed. An action is represented as a sequence of joint hog-flow descriptors extracted independently from each frame. During training, a set of action prototypes is first learned based on a k-means clustering, and then a binary tree model is constructed from the set of action prototypes based on hierarchical k-means clustering. Each tree node is characterized by a shape-motion descriptor and a rejection threshold, and an action segmentation mask is defined for leaf nodes (corresponding to a prototype). During testing, an action is localized by mapping each test frame to a nearest neighbor prototype using a fast matching method to search the learned tree, followed by global filtering refinement. An action is recognized by maximizing the sum of the joint probabilities of the action category and action prototype over test frames. Our approach does not explicitly rely on human tracking and background subtraction, and enables action localization and recognition in realistic and challenging conditions (such as crowded backgrounds). Experimental results show that our approach can achieve recognition rates of 100% on the CMU action dataset and 100% on the Weizmann dataset.

1 Introduction

Action recognition has become an active research topic in computer vision. In this paper, we propose a simultaneous approach to localize and recognize multiple action classes based on a unified tree-based framework.

Realistic actions often occur against a cluttered, dynamic background and are subject to large variations in people's posture and clothing, illumination variations, camera motions and occlusion. Figure 1 shows examples of action frames in realistic environments (with cluttered backgrounds and moving objects). In these cases, it is not easy to detect and segment the actors from the backgrounds. Consequently, they pose a significant challenge for those action recognition approaches which perform simple preprocessing such as background subtraction [1–4]. Even though many previous works have been done for action recognition [5–9], robustly localizing and recognizing actions viewed against a cluttered and dynamic background is still important to explore.

(a) (b) (c)

Fig. 1. Examples of action localization and recognition. (a) The jumping-jacks action is recognized even though large background motion flows created by crowds; (b) The two-handed-wave action is recognized correctly while the actor is partially occluded; (c) The one-handed-wave action is recognized while there is a significant motion interruption from vehicles in the background.

Localizing an action in a video is more computationally complex than searching for an object in a 2D image. For example, a video sequence with size $120 \times 160 \times 1000$ frames can produce more than 10^{14} subvolumes with variable spatial and temporal dimensions, which is approximately 10^6 times larger than the number of subwindows produced when searching for an object in a 120×160 image. Although there are some recent approaches for efficiently searching subwindows in a 2D image for object detection [10], they cannot be easily extended to volume search in 3D spatial and temporal spaces. Most prior work employs the sliding window scheme for detecting actions. Actions are usually described by adopting spatial-temporal features and combined shape and motion information. The classifiers are based on cascade classifiers [11, 12], vocabulary trees [7] and branch-and-bound schemes [10, 13]. However, most previous approaches for action detection and retrieval built action detectors independently for each action class, which may not scale well for detecting a large number of action types.

We introduce an efficient, tree-based approach to localize and recognize an action simultaneously. This approach extends our previous work [9]. Compared to [9], the differences between two approaches include: (1) the shape feature in this paper is based on HOG while the shape feature in [9] is based on binary silhouette or appearance-based likelihood; (2) the prototype tree in this paper is used to localize and recognize actions while the tree in [9] is used to speed up the process of prototype matching and actor location refinement; (3) the probabilistic framework in this paper is constructed to determine action category labels and action prototypes, while the probabilistic framework in [9] is built to determine actor location and action prototype.

The block diagram of our approach is shown in Figure 2. In the training phase, an action interest region [1] is specified in each frame of a set of training videos. Action prototypes are learned via k-means clustering on the entire set of the computed hog-flow descriptors. Next, a binary tree model is constructed using the set of learned action prototypes. In this tree model, each leaf node cor-

[1] Its center is on the vertical central axis of human bounding box, and its side length is proportional to the height of the bounding box.

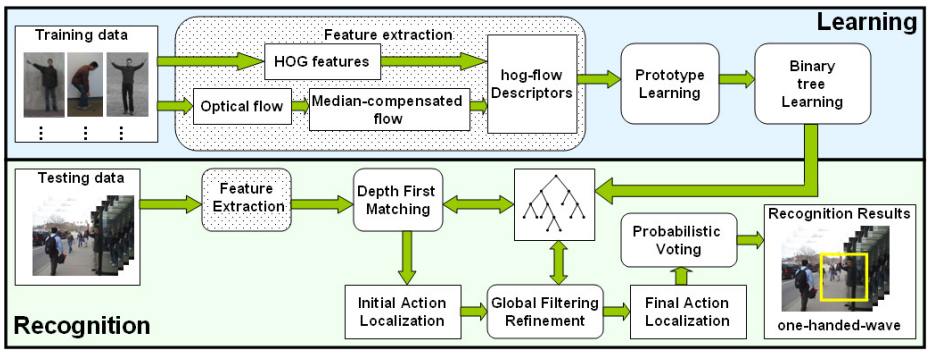


Fig. 2. Overview of our approach.

responds to a learned action prototype and contains a list of parameters including the probability of the prototype belonging to an action category, frame indices of all the training descriptors which matched to this prototype, and a rejection threshold. These parameters allows us to integrate action localization, recognition and segmentation in a video. In the testing phrase, we use the conventional sliding-window-based scheme to generate a sequence of initial bounding boxes of an action. These initial bounding boxes are rapidly obtained by traversing the learned binary tree using a fast matching approach (see sec. 2.2) to find the nearest neighbor prototypes. This is followed by a Kalman filtering-based refinement on their scales and positions. Finally, given the refined bounding boxes at each frame, the action category is identified by maximizing a sum of joint probability method. Our main contributions are three fold:

- A HOG-based shape feature is adopted for modeling shape information to enhance the joint shape-motion descriptors proposed in Lin et al. [9].
- A binary-tree-based approach is introduced to efficiently localize and recognize multiple action classes against cluttered, dynamic backgrounds and under partial occlusions.
- Action recognition is modeled as a maximum probability estimation problem of the joint probabilities of action category labels and action prototypes.

1.1 Related Work

Numerous approaches have been proposed for action detection and recognition recently. Some approaches use motion trajectories of human bodies or body interest points (or landmarks) to recognize actions, mainly based on visual tracking [14–17]. Other approaches represent an action as a space-time volume or use human silhouettes as shape models for action recognition [1–4, 18]. But the latter typically requires background subtraction, which faces severe difficulties in real world scenarios (*i.e. cluttered, dynamic backgrounds*). Recently, space-time interest points, or local feature-based approaches, have been applied to action detection and recognition [6, 19–21, 8]. Many approaches combine multiple features to improve action recognition performance [22, 7, 23, 12, 24]. Commonly

used classifiers for action recognition include k-NN classifiers [25, 26], support vector machine (SVM) [21, 22, 6, 27, 28], boosting-based classifiers [16, 12, 20, 11], hidden markov model (HMM) [29], dynamic time warping (DTW)-based classifiers [24, 9, 30] and Hough voting schemes [31].

In analogy to object detection, sliding window-based schemes have been employed for action detection. Laptev and Rerez [12] combined histograms of oriented gradients (HOG) [32] and histograms of optical flow for action detection. Thurau and Hlavac [25] extended the standard HOG-based pose descriptor to cope with background clutter and used n-Gram models for sequence matching. Mikolajczyk and Uemura [7] used a large number of local motion-appearance features and represented them in a vocabulary forest. Features extracted from a test sequence are matched to the trees in the forest to vote for the action categories and locations.

However, most of the action detection approaches [5, 13, 12] built a detector for each action class independently. Detection is then very expensive when the number of action classes is large. For multi-class action recognition, sharing feature computation and basic classification steps between action categories is desirable. This has been investigated in [33] for multi-class object detection. Motivated by this work, we introduce a simultaneous multi-class action localization and recognition approach by sharing information (feature computations) to reduce redundancy and improve efficiency.

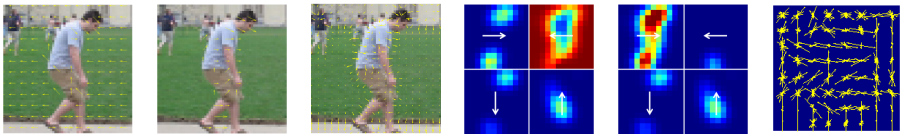
2 Action Representation and Learning

We use hog-flow descriptors for representing actions. Here we regard a sliding-window region as a potential action interest region. A hog-flow descriptor is extracted from each potential action interest region in an image for determining whether the action of interest exists. An action interest region[2] is defined as a square region around the human body.

2.1 Hog-flow Descriptor

We compute a shape descriptor based on the histogram of oriented gradients introduced in [32]. A simplified HOG descriptor for an action interest region is represented as a feature vector $D_h = (h_1...h_{n_h}) \in \mathcal{R}^{n_h}$ by dividing the action interest region into n_h non-overlapping square grids (or sub-regions) $R_1...R_{n_h}$. Unlike [32] which computes HOG descriptors from a dense overlapping grid, we compute them on a non-overlapping square grid. More specifically, we compute the descriptor as follows: The input image I is first smoothed by a Gaussian filter with standard deviation $\sigma_g = 2$; then we use a simple 1-D centered mask [-1, 0, 1] to compute the image spatial gradient x component $g_x(x, y) =$

[2] For the training data, we compute the action interest region from background subtraction. The action interest region is defined as a square region around the localized bounding box. For the test data, the action interest region is obtained by global filtering refinement (see sec. 3.2).



(a) (b) (c) (d) (e) (f)

Fig. 3. An example of computing the hog-flow descriptor of an action frame against a moving camera and a dynamic background. (a) Raw optical flow field; (b) Motion-compensated optical flow field; (c) Image spatial gradient, (d) Flow descriptor D_f computed from the raw optical flow field. The flow descriptor is visualized by placing its four channels in a 2×2 grid. (e) Flow descriptor D_f computed from the compensated optical flow field, (f) A visualization of HOG descriptor D_h with 8×8 grid and 9 orientation bins.

$I(x+1,y) - I(x-1,y)$ and y component $g_y(x,y) = I(x,y+1) - I(x,y-1)$ along x direction and y direction respectively[3]. The magnitude ρ and orientation θ are computed by $\rho(x,y) = \sqrt{g_x(x,y)^2 + g_y(x,y)^2}$, $\theta = \arctan g_y(x,y)/g_x(x,y)$. Next, we divide the action interest region into n_h non-overlapping square grids. For each grid, we accumulate the votes over all the pixels (x,y) into n_o orientation bins weighted by their magnitude $\rho(x,y)$ to compute the histograms of oriented gradients. Finally, the feature vector D_h is obtained by concatenating all the histogram entries and L_2-normalization. The image spatial gradients and the HOG descriptor computed from an action interest region are visualized in Figure 3(c) and 3(f), respectively.

A flow descriptor for an action interest region is represented as a n_f-dimensional feature vector $D_f = (qbmf_x^+, qbmf_x^-, qbmf_y^+, qbmf_y^-) \in \mathcal{R}^{n_f}$, where '$qbmf$' refers to quantized, blurred, motion-compensated flow. The flow descriptor D_f is computed by using the approach introduced in [9]. A median flow-based motion compensation scheme is used for handling the influences of moving cameras and dynamic backgrounds in [9]. Figure 3(a) and 3(b) show an example of motion flow compensation for an action frame against a moving camera and a dynamic background. From these two figures, we know that this approach not only effectively removes background flows but also corrects foreground flows, so the extracted flow descriptors are robust against cluttered, dynamic backgrounds. A flow descriptor for an example action interest region with and without motion compensation, are visualized in Figure 3(d) and 3(e) respectively.

We concatenate the shape and motion descriptors D_h and D_f to form a joint hog-flow descriptor. The distance between two hog-flow descriptors is computed using the Euclidean distance metric.

2.2 Tree Model Construction and Matching

We represent an action as a set of representative action prototypes [29, 25, 9], $\Lambda = (\lambda_1, \lambda_2 ... \lambda_k)$. For compressing the redundant information from the training

[3] For computing the gradient vector at pixel position (x,y) of color images, we compute the image gradients for each color channel and take the one with largest norm.

descriptors in the presence of small inter-class or large intra-class variability, and learning representative action prototypes Λ, we perform clustering on the entire set of hog-flow descriptors. Next, a binary tree model is constructed from these learned prototypes.

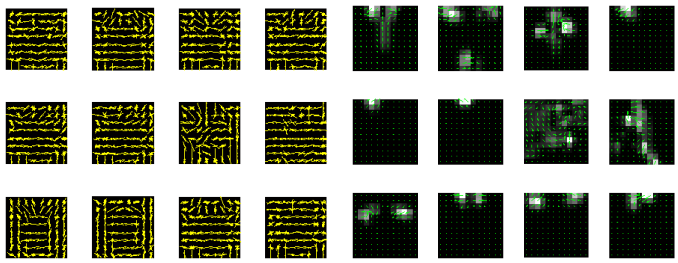
Prototype Learning Given the set of descriptors extracted from all frames of the training videos, we perform k-means clustering using the Euclidean distance measure for learning the action prototypes. Since our HOG descriptors and flow descriptors are obtained by L_2 normalization, the Euclidean distance metric is reasonable for clustering the joint hog-flow descriptors. The resulting cluster centers are used as the learned action prototypes. Figures 4(a) and 4(b) show typical HOG components and flow components of action prototypes.

Tree Learning In order to rapidly localize and recognize an action occurring in a video, we build a binary tree over the set of prototypes based on the hierarchical k-means clustering algorithm [34].

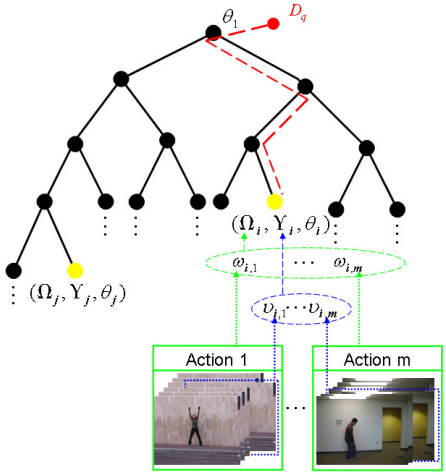
For maximizing the efficiency of the tree search algorithm, we estimate a matching rejection threshold for each tree node. The thresholds are denoted as $\Theta = (\theta_1, \theta_2, ..., \theta_{n_t})$, where n_t is the number of tree nodes. These thresholds are learned from the training data as follows: For the i-th tree node, let D_{leaf_i} denote the maximum Euclidean distance between the descriptor corresponding to the i-th node and all of the training descriptors corresponding to its children leaf nodes. The threshold θ_i is set to τD_{leaf_i}, where the factor τ is chosen to reduce the influence of the noisy samples from the HOG or flow components of training descriptors. τ is estimated via cross-validation.

In addition to the rejection threshold associated with each tree node, each leaf node λ_i contains a list of parameters $\Upsilon_i = (\upsilon_{i,1}, ..., \upsilon_{i,m}), \Omega_i = (\omega_{i,1}, ..., \omega_{i,m})$. Υ_i lists the frame indices of training descriptors that best match with the prototype (leaf node) λ_i. Ω_i denotes the probability distribution of each prototype (leaf node) belonging to an action class$\{\alpha_i\}_{i=1...m}$. We first compute an observation vector $\hat{\Omega}_i = (\hat{\omega}_{i,1}, ..., \hat{\omega}_{i,m})$. $\hat{\omega}_{i,m}$ is defined as $\hat{\omega}_{i,m} = \frac{F_{i,m}}{F_m}$, where $F_{i,m}$ denotes the number of training features from class m that are assigned to leaf node λ_i and F_m denotes the number of training features in class m. $\hat{\Omega}_i$ is L_1 normalized to generate a class distribution vector Ω_i. These parameters are estimated by matching the set of hog-flow descriptors from the training data to the learned tree, which is very important in estimating actor's position, scale, and action category label for a testing video (explained in sec. 3). An example of the binary tree model is visualized in Figure 4(c). The yellow leaf nodes in the figure represent action prototypes.

Fast tree matching Given a query subwindow from a frame of video, a query hog-flow descriptor is extracted and first compared to the descriptor corresponding to the top node (see Figure 4(c)). For each non-leaf node, if the distance between the query feature descriptor D_q and the current tree node descriptor



(a) HOG components (b) Flow components



(c) Learned binary tree model

Fig. 4. An example of tree learning. (a)(b) Visualization of HOG and flow components of learned prototypes for $k = 12$. The HOG component is represented by 8×8 grids and 9 orientation bins. The flow component is represented by four (orientation channels) 12×12 grids. In the flow component, grid intensity indicates motion strength and 'arrow' indicates the dominant motion orientation at that grid, (c) The learned binary tree model. Yellow tree nodes are prototypes , Query feature D_q and its searching path (red color dashed line).

is less than its rejection threshold, this descriptor is accepted and we continue traversing the tree; the child node most similar to the query descriptor is selected as the next node. This continues until the query feature reaches a leaf node of the tree. On the other hand, if the distance ever exceeds the rejection threshold of the tree node, then this query feature is rejected and no longer compared with its children nodes.

3 Action Recognition

3.1 Problem Formulation

Let random variable V be an observation from an image frame, λ be a prototype random variable chosen from the set of k learned hog-flow prototypes

$\Lambda = (\lambda_1, \lambda_2 ... \lambda_k)$, α be an action category random variable chosen from the set of m action categories $A = (\alpha_1, \alpha_2 ... \alpha_m)$, and $\beta = (x, y, s)$ denote random variables representing actor location comprising image location (x, y) and scale s. Then, the problem of action localization and recognition in a test video is equivalent to maximizing the joint probability distribution $p(\alpha, \lambda, \beta, V)$ in each frame of the video. We assume that i) the observation V is given; ii) action category α is independent of actor location β; and iii) the probability of actor location β given observation V is uniform. The joint probability distribution can now be decomposed into an action category mapping term and a prototype matching term:

$$p(\alpha, \lambda, \beta, V) \propto p(\alpha, \lambda, \beta | V) \propto p(\alpha | V, \lambda) p(\lambda | V, \beta) \tag{1}$$

The action category mapping term $p(\alpha|V, \lambda)$ is estimated by α-th element $\omega_{i,\alpha}$ of the class distribution vector $\Omega_i = (\omega_{i,1}, ..., \omega_{i,m})$. We model the prototype matching term $p(\lambda|V, \beta)$ as:

$$p(\lambda | V, \beta) = e^{-d(D(V,\beta), D(\lambda))}, \tag{2}$$

where d represents the Euclidean distance between the descriptor $D(V, \beta)$ determined by observation V at location β, and the descriptor $D(\lambda)$ corresponding to the prototype λ.

3.2 Refinement by Global Filtering

An initial estimate of position (x, y) and scale s of actor location is first obtained by a conventional sliding-window-based searching scheme over the video frames using the fast tree matching method. For reducing the influence of noisy optical flow and image gradient from a test frame, we perform Kalman filtering to refine the positions and scales for the initial estimate of position (x, y) and scale s of actor locations over all test frames. The refined position and scale of each frame is used as an action interest region. The dynamic equation and measurement equation are:

$$\begin{bmatrix} \beta_{t+1} \\ \dot{\beta}_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \beta_t \\ \dot{\beta}_t \end{bmatrix} + Q_t, \tag{3}$$

$$\hat{\beta}_{t+1} = \begin{bmatrix} 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \beta_{t+1} \\ \dot{\beta}_{t+1} \end{bmatrix} + R_t, \tag{4}$$

where $\beta_t = (x_t, y_t, s_t)$ denotes the location and scale for frame t, $\dot{\beta}_t$ is the velocity and $\hat{\beta}_t$ is the measured location and scale for frame t, which has the minimum distance between the extracted descriptor extracted from the query subwindow specified by $\hat{\beta}_t$, and any leaf nodes of the tree. $Q_t \backsim N(\mathbf{0}, diag(\sigma_{\beta}^2, \sigma_{\dot{\beta}}^2))$ is the process noise and $R_t \backsim N(\mathbf{0}, diag(\sigma_{\hat{\beta}}^2))$ is the measurement noise.

3.3 Action Recognition

After global filtering, the actor's refined locations and scales $\{\beta_t\}_{t=1...T}$ are known, where T represents the number of frames in the test video. Action recognition proceeds by re-computing hog-flow features from these image windows. Given a location and scale β_t at frame t, we compute the joint probability of action category label α_i and prototype λ_i at frame t as follows:

$$J_t(\alpha_i) = \omega_{i,\alpha_i} e^{-d(D(V,\beta_t),D(\lambda_i(\beta_t)))}, \tag{5}$$

Each descriptor extracted from the refined locations and scales $\{\beta\}_{t=1...T}$ and each frame contribute to the probability of action categories $\{\alpha_i\}_{i=1...m}$ occurring in the test video. This is equivalent to maximizing the sum of joint-probabilities. Finally, the maximum joint-probability action label is given as:

$$\alpha_i^* = argmax_{\{\alpha_i\}_{i=1...m}} \sum_{t=l_{start}}^{l_{end}} J_t(\alpha_i), \tag{6}$$

where l_{start} and l_{end} are the start frame and end frame of the test sequence, respectively. This is equivalent to probabilistic (soft) voting where each frame contributes a probability of belonging to a category.

3.4 Action Segmentation

In addition to action localization and recognition, our approach also segments the actor's pose in each frame of a test action sequence. After the process of tree construction, each action category in i-th leaf node (prototype) has its own set of representative training descriptors, which is stored in $\Upsilon_i = (\upsilon_{i,1}, ..., \upsilon_{i,m})$ by their indices during training.

Given the probability parameters $\Omega_i = (\omega_{i,1}, ..., \omega_{i,m})$ for the i-th leaf node (prototype), we define a segmentation mask for i-th leaf node as $B_i = \sum_{j=1}^{m} \omega_{i,j} b_j$, where $\{b_j\}_{j=1..m}$ are the binary silhouettes from the training data corresponding to each action category and identified by $\Upsilon_i = (\upsilon_{i,1}, ..., \upsilon_{i,m})$. If a test frame corresponds to the i-th leaf node, it can use its corresponding averaged segmentation mask to segment the actions. Example results of action segmentations are shown in Figure 7 and 8. A more precise action segmentation may resort to body pose fitting or other useful observations, but our results can be used as initial segmentations and input to a high level segmentation program.

4 Experiments

We evaluated our approach on two public action datasets: CMU action dataset [5] and Weizmann action dataset [1], in terms of recognition rate and average computation time. The average computation time is computed as the average time required to localize and recognize an actor in a test frame. Our experiment focused on the CMU action dataset, since this dataset is much more difficult due

(a) CMU dataset (b) Weizmann dataset

Fig. 5. Evaluation datasets.

to significant background clutter and motion. For computing simplified HOG descriptors, we divide an action interest region into $n_h = 8 \times 8$ non-overlapping square cells and accumulate the votes over all pixels into $n_o = 9$ orientation bins for each cell. The hog-flow descriptor is 1152-dimensions which consists of a $8 \times 8 \times 9 = 576$-dimensional HOG descriptor and a $12 \times 12 \times 4 = 576$-dimensional flow descriptor.

4.1 Evaluation on the CMU Action Dataset

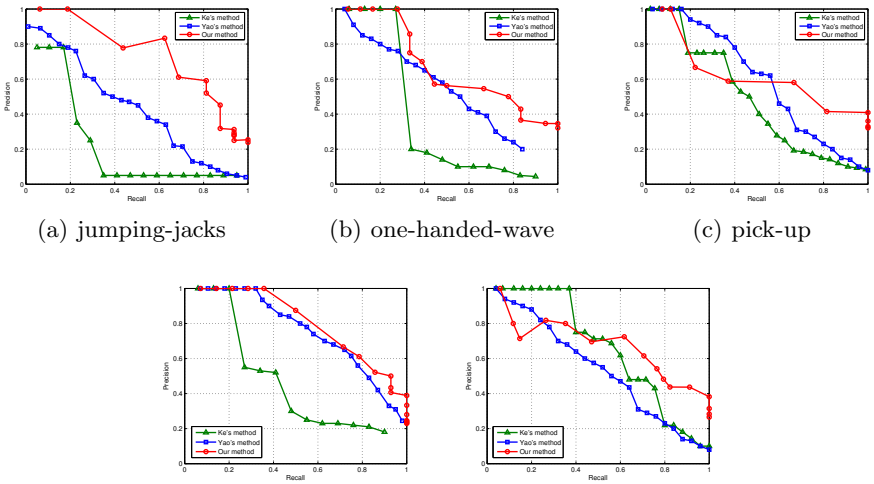
This dataset consists of five action classes: 'jumping-jacks', 'one-handed-wave', 'pick-up', 'push-button' and 'two-handed-wave'. Totally there are 48 video sequences for the training data. Figure 5(a) shows sample training frames from the dataset. The testing data contains 110 videos (events) which are down-scaled to 160×120 in resolution. The dataset is known to be very challenging, because it was captured by using a hand-held camera in environments with moving persons or vehicles in the background. The experiment results reported from the dataset are only the action detection results [5, 24]. We evaluated our approach on both action localization (detection) and action recognition.

We first detect the actions occurring in the test videos. We used the decision criterion from [5, 24], where a detection is correct if the intersection of the detected and the ground truth bounding boxes is larger than 50% and the classification label is correct. We generated the precision-recall (P-R) curve for each action: $Precision = TP/(TP + FP)$ and $Recall = TP/NP$, where TP is the number of true positives, FP is the number of false positives and NP is the total number of positives. Figure 6 shows P-R curves of each action. The red curves correspond to our approach, while the green and the blue P-R curves are the basic results reported in [5] and [24] respectively. In general, our approach achieved better performance compared to the results reported in [5, 24].

In addition to integrated detection and recognition performance shown in the P-R curves, we also evaluated isolated recognition performance given ground truth detections in order to measure the quality of our action recognition component. We used the ground truth actor location in the first frame, and then searched for the actor in its local neighborhood in later frames. We evaluated our approach with respect to the number of prototypes k from 500 to 2789. As shown in Table 1, the recognition rate reached 100% at $k = 2789$.

Table 1. Prototype-based recognition results using joint hog-flow features on the CMU dataset (cluttered, dynamic background).

method	recog. rate (%)	avg. time (s)
500 proto.	84.55	0.86
1000 proto.	89.09	0.91
1700 proto.	89.09	0.88
2300 proto.	90	0.92
2789 proto.	100	0.89



(a) jumping-jacks (b) one-handed-wave (c) pick-up

(d) push-button (e) two-handed-wave

Fig. 6. Precision-Recall curves of action detection on the CMU dataset. The green curve labeled as 'Ke's method' is the results reported in [5]; the blue curve labeled as 'Yao's method' is the results reported in [24].

Figure 7 shows some qualitative results of action localization and recognition on this dataset.

4.2 Evaluation on the Weizmann Action Dataset

The Weizmann dataset contains 90 videos separated into 10 actions performed by 9 persons. Example frames of this dataset are shown in Figure 5(b). We performed leave-one-person-out experiments to evaluate our approach.

We evaluated the performance of our approach using different number of prototypes. Table 2 shows our experimental results. By using all the training descriptors (except the training descriptors from the testing person) as action prototypes, our approach obtained 100% recognition rate. We compared these results to the state of art action recognition approaches [16, 23, 9, 22, 1, 25]. We achieved the same recognition rate as [16, 23, 9], but note that we did not use background subtractions and object tracking to localize and recognize actions. Figure 8 gives some localization and recognition results using our approach.

Fig. 7. Examples of action localization and recognition results on the CMU dataset. Note that our approach effectively handled interruption of moving people and vehicles in the scene. The 1_{st} row: jumping-jacks; The 2_{nd} row: one-handed-wave; The 3_{rd} row: pick-up; The 4_{th} row: push-button; The 5_{th} row:two-handed-wave. The green regions are the segmentation masks.

5 Conclusions

The experimental results show that our approach yields very good results for action localization and recognition in realistic scenarios with cluttered, dynamic backgrounds. Our approach does not rely on background subtraction and human tracking. In the future, we aim to combine local feature voting-based approach [7] with our global scheme to improve our results further. Additionally, we are also exploring scene context [8] as priors to improve our system and to apply it to more complicated scenarios such as movies and TV shows.

Acknowledgement

This research was funded in part by the VIRAT program.

References

1. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes (2005) *ICCV*.

Table 2. Prototype-based recognition results using joint hog-flow features on the Weizmann dataset (static background). 'BS' and 'OT' denote background subtraction and object tracking respectively.

method	recog. rate (%)	avg. time (s)	BS/OT
500 proto.	91.11	0.91	None
1500 proto.	92.22	0.93	None
2500 proto.	88.89	0.94	None
3000 proto.	90.00	0.96	None
4000 proto.	94.44	0.96	None
all descriptors	100	0.94	None
Fathi [16]	100	N/A	OT
Schindler [23]	100	N/A	OT
Lin [9]	100	N/A	BS, OT
Jhuang [22]	98.8	N/A	BS
Blank [1]	99.61	N/A	BS
Thurau [25]	94.40	N/A	None

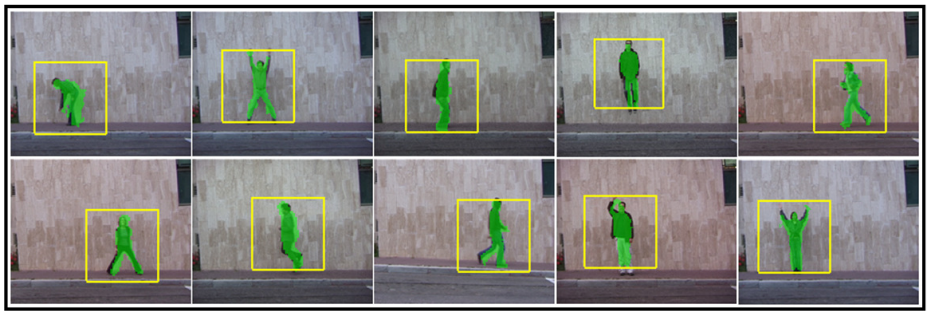


Fig. 8. Examples of action localization and recognition results on the Weizmann dataset. The 1_{st} row: bend, jack, jump, pjump, run; The 2_{nd} row: side, skip, walk, one-hand-wave, two-hand-wave. The green regions are the segmentation masks.

2. Liu, J., Ali, S., Shah, M.: Recognizing human actions using multiple features (2008) *CVPR*.
3. Li, F., Nevatia, R.: Single view human action recognition using key pose matching and viterbi path searching (2007) *CVPR*.
4. Bobick, A., Davis, J.: The recognition of human movement using tempral templates. IEEE Trans. PAMI **23** (2001) 257–267
5. Ke, Y., Sukthankar, R., Hebert, M.: Event detection in crowded videos (2007) *ICCV*.
6. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies (2008) *CVPR*.
7. Mikolajczyk, K., Uemura, H.: Action recognition with motion-appearance vocabulary forest (2008) *CVPR*.
8. Marszalek, M., Laptev, I., Schmid, C.: Actions in context (2009) *CVPR*.
9. Lin, Z., Jiang, Z., Davis, L.S.: Recognizing actions by shape-motion prototype trees (2009) *ICCV*.
10. Lampert, C., Blaschko, M., Hofmann, T.: Beyond sliding windows: Object localization by efficient subwindow search (2008) *CVPR*.

11. Ke, Y., Sukthankar, R., Hebert, M.: Efficient visual event detection using volumetric features (2005) *ICCV*.
12. Laptev, I., Perez, P.: Retrieving actions in movies (2007) *ICCV*.
13. Yuan, J., Liu, Z., Wu, Y.: Discriminative subvolume search for efficient action detection (2009) *CVPR*.
14. Ali, S., Basharat, A., Shah, M.: Chaotic invariants for human action recognition (2007) *ICCV*.
15. Efros, A.A., Berg, A.C., Mori, G., Malik, J.: Recognizing action at a distance (2003) *ICCV*.
16. Fathi, A., Mori, G.: Action recognition by learning mid-level motion features (2008) *CVPR*.
17. Sheikh, Y., Sheikh, M., Shah, M.: Exploring the space of a human action (2005) *ICCV*.
18. Li, W., Zhang, Z., Liu, Z.: Expandable data-driven graphical modeling of human actions based on salient postures. IEEE Trans. Circuits and Systems for Video Technology **18** (2008) 1499–1510
19. Niebles, J.C., Wang, H., Fei-Fei, L.: Unsupervised learning of human action categories using spatial-temporal words. Int'l J. Computer Vision **79** (2008) 299–318
20. Nowozin, S., Bakir, G., Tsuda, K.: Discriminative subsequence mining for action classification (2007) *ICCV*.
21. Dollar, P., Rabaud, V., Cottrell, G., Belongie, S.: Behavior recognition via sparse spatio-temporal features (2005) *VS-PETS*.
22. Jhuang, H., Serre, T., Wolf, L., Poggio, T.: A biologically inspired system for action recognition (2007) *ICCV*.
23. Schindler, K., Gool, L.V.: Action snippets: How many frames does human action recognition require? (2008) *CVPR*.
24. Yao, B., Zhu, S.: Learning deformable action templates from cluttered videos (2009) *ICCV*.
25. Thurau, C., Hlavac, V.: Pose primitive based human action recognition in videos or still images (2008) *CVPR*.
26. Weinland, D., Boyer, E.: Action recognition using exemplar-based embedding (2008) *CVPR*.
27. Liu, J., Shah, M.: Learning human actions via information maximization (2008) *CVPR*.
28. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: A local svm approach (2004) *ICPR*.
29. Elgammal, A., Shet, V., Yacoob, Y., Davis, L.S.: Learning dynamics for exemplar-based gesture recognition (2003) *CVPR*.
30. Veeraraghavan, A., Chellappa, R., Roy-Chowdhury, A.K.: The function space of an activity (2006) *CVPR*.
31. Yao, A., Gall, J., Gool, L.V.: A hough transform-based voting framework for action recognition (2010) *CVPR*.
32. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection (2005) *CVPR*.
33. Torralba, A., Murphy, K.P., Freeman, W.T.: Sharing visual features for multiclass and multiview object detection. IEEE Trans. PAMI **29** (2007) 854–869
34. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree (2006) *CVPR*.