Learning Deformable Action Templates from Cluttered Videos

Benjamin Yao and Song-Chun Zhu
Department of Statistics, UCLA Lotus Hill Research Institute
Los Angeles, CA Ezhou, China
{zyyao, sczhu}@stat.ucla.edu

Abstract

In this paper, we present a Deformable Action Template (DAT) model that is learnable from cluttered real-world videos with weak supervisions. In our generative model, an action template is a sequence of image templates each of which consists of a set of shape and motion primitives (Gabor wavelets and optical-flow patches) at selected orientations and locations. These primitives are allowed to slightly perturb their locations and orientations to account for spatial deformations. We use a shared pursuit algorithm to automatically discover a best set of primitives and weights by maximizing the likelihood over one or more aligned training examples. Since it is extremely hard to accurately label human actions from real-world videos, we use a three-step semi-supervised learning procedure. 1) For each human action class, a template is initialized from a labeled (one bounding-box per frame) training video. 2) The template is used to detect actions from other training videos of the same class by a dynamic space-time warping algorithm, which searches a best match between the template and target video in 5D space (x, y, scale, $t_{template}$ and t_{target}) using dynamic programming. 3) The template is updated by the shared pursuit algorithm over all aligned videos. The 2nd and 3rd steps iterate several times to arrive at an optimal action template. We tested our algorithm on a cluttered action dataset (the CMU dataset) and achieved favorable performance than [7]. Our classification performance on the KTH dataset is also comparable to state-of-the-arts.

1. Introduction

Real-world actions occur often in cluttered, dynamic environments and actors can not easily be segmented from the background due to distracting motion from other objects in the scene. This poses a challenge for many approaches on action recognition [1, 5, 23, 11] that assume the foreground segmentation (silhouette) is available. Even a milder assumption that requires actors to be tracked and aligned by a bounding box [17, 6] is sometimes unrealistic. On the one hand, it is a very hard task to automatically detect and track humans in highly cluttered scenes (see the last row of Fig.1 for example). On the other hand, it is extremely tedious to manually label human actions from real-world videos for two reasons: 1) One bounding box each frame is required to offset global motion; 2) Frame-to-frame correspondences

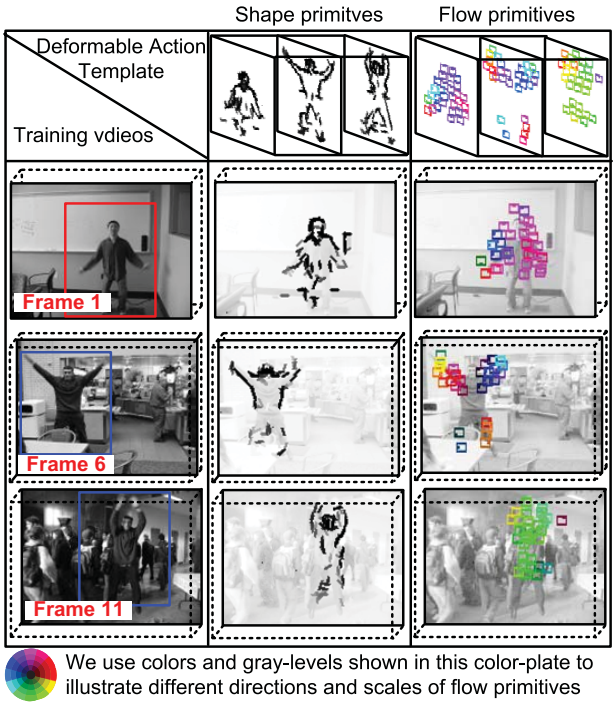


Figure 1. Deformable action template learned for a "jumping jack" action. The 1st row displays 3 frames of the template, each composed of 70 shape primitives and 30 motion primitives. Shape primitives and motion primitives are illustrated by bars and colored rectangles (see the color-plate above for explanation) respectively. The leftmost column shows three training videos. The actor in the 1st video is labeled with a bounding box (red rectangle), the blue rectangles in the other two videos are automatically detected. The 2nd and 3rd columns shows locally shifted versions of the shape and motion primitives fitting to the corresponding training images. Better to view on screen.

are required to synchronize actions.

In this paper, we try to solve a problem more practical in real-world scenarios: Given one example for each action class, can we automatically discover other similar actions from a set of weakly supervised training videos and learn a common action model from them? This can be boiled down into two separate problems: Firstly, how to build an action template, from aligned training examples, that is robust to variations (view-point, illumination, clothing, size, age, gender, etc.). Secondly, how to effectively detect the action from real-world videos using the learned template, which is a similar task as object detection since the action can be located anywhere in the scene (in both space and time) and requires reliable detection in the presence of significant background clutters.

To solve the first problem, we propose a Deformable Action Template model extended from [22]. Fig.1 illustrates the basic idea. The leftmost column displays 3 training videos of a "jumping jack" action. We first assume these actions are aligned by a bounding box per frame centered at the actor (In fact, only the red bounding box is labeled. Blue ones are automatically detected). Then the action can be represented by a deformable action template consisting of a sequence of image templates, each of which consists of 70 shape primitives (Gabor wavelets) and 30 motion primitives (optical-flow patches). The first row of Fig.1 displays 3 frames of the learned template. Shape and motion primitives are illustrated by a bar and a colored rectangle respectively. The primitives are allowed to locally perturb in position and orientation when they are linearly combined to encode each training or testing example, as illustrated by the 2nd and 3rd columns of Fig.1. The DAT model can be learned from one or more aligned training videos by a shared pursuit algorithm modified from [22].

To solve the second problem, we adopted a Dynamic Time Warping (DTW) algorithm [15], which is widely used in voice recognition domain, and modified it to suit our case. The original DTW seeks to find a best match between two one dimensional signals (e.g. voice signals) by dynamic programming (DP). We extend this algorithm to find a best match between an action template with an action located at unknown position, scale and time inside a video clip, which can also be solved efficiently by DP similar to DTW (Thus we call it Dynamic Space-Time Warping (DSTW)). It is worth mention that the DSTW not only detects the starting and ending points of an action, but also establishes a frame-to-frame correspondence between the template and the target (i.e. find a best time warping function that synchronize the target action and the template). In this sense, our model is deformable not only in space, but also in temporal domain. We use semi-supervised learning algorithm to learn from one labeled and a set of weakly supervised training example (as illustrated in Fig.3). The algorithm can be summarized into three steps: 1) A deformable action template is initialized from one labeled training video. 2) The DSTW algorithm is applied to detect the actions from other unlabeled training videos and align them with the template. 3) The template is updated by a shared pursuit algorithm. Step 2) and 3) iterate several times to arrive at an optimal template.

1.1. Contribution and related work

The contributions of this paper are: (1) A generative model of action with a likelihood defined directly on the video image intensities and motion speeds (in stead of defining on features such as [13]), which enables efficient learning in that both the adaptive basis selection and spatial-temporal alignment can be solved together in a unified maximum likelihood framework. (2) A semi-supervised learning approach for learning actions from cluttered videos that is valuable to real-world use.

To credit past work, the deformable action template and shared pursuit algorithm are mainly inspired by the *active basis* model proposed by Ying-Nian Wu et al. [22], which focus on building object shape models from static images. Our work also has strong connections to the biological inspired vision systems proposed by Riesenhuber and Poggio [16] and a recent work that extends the model to action recognition by H. Jhuang et al.[6]. Following the same line, Schindler et al. proposed an *action snippets* model studying the minimum amount of frames required to recognize an action [17]. Both Jhuang and Schindler, however, focus on the classification problem and assume that the actions are aligned by bounding boxes. Other work for action recognition and detection can be roughly divided into two groups. One group represents actions using global models such as motion history [1], spatial-temporal visual hull [5, 23] and example-based model [3, 7, 19, 21]. Motion and/or shape representation are used to be robust against appearance variations. The articulated skeleton models [4, 14] also fall in this category. By representing the human action with kinematic models, skeleton representation grasped the intrinsic structure of the human body, thus is able to encode the compositional structure and variations. But this intrinsic representation is very hard to obtain from raw videos. Another group represents actions as a distribution over local spatial-temporal interest points [2, 13, 9]. While the sparsity and resulting computational efficiency are appealing, the S-T interest points cannot be reliably extracted from raw videos. Among recent work, J. Liu et al. proposed to use page-ranking algorithm to extract reliable features from video and presented an action recognition method is able to detect recognize realistic action from cluttered real-world videos [10]. J. Yuan et al. proposed to use brand-and-bound algorithm to speed up action detection [24]. Both these two methods are tested on real-world datasets.

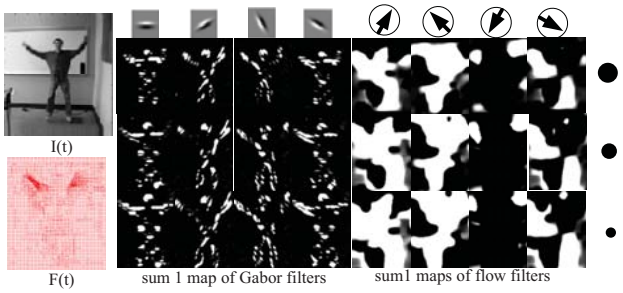


Figure 2. Shape and flow maps. Shape $S1$ maps are computed by convolving Gabor filters of multiple orientations with video frames ($I(t)$). Flow $S1$ maps are computed by projecting optical flow ($F(t)$) within a window onto multiple directions. Both maps are computed at multiple scales.

2. Deformable action template formulation

2.1. Shape and flow primitives

In this paper we adopt a design where shape and flow primitives are defined on image and optical flow of each frame as illustrated in Fig.2. This is a similar design as the $S1$ (sum1) maps used in [6].

Shape. We use Gabor wavelets as the shape primitive dictionary and use $B_{x,y,o,s}$ to denote a wavelet located at position (x, y), orientation o and scale s. The filter response is the squared inner product between $B_{x,y,o,s}$ and the image patch $I_{\Lambda_j}(t)$ centered at (x, y):

$$r^{shape}(I(t); B_{x,y,o,s}) = Sigmoid(|\langle I_{\Lambda_j}(t), B_{x,y,o,s}\rangle|^2) \tag{1}$$

where the sigmoid transformation saturates large responses. A large value of $r^{shape}(I(t); B_{x,y,o,s})$ indicates that there is a strong shape element (edge or bar) of orientation o at location (x, y, s). To suppress strong static edges detected in the background (e.g. the corner of a wall), we subtract the response $|\langle I_{\Lambda_j, B_{x,y,o,s}}(t)\rangle|^2$ by the average response at (x, y, o, s) over all frames.

Flow. Flow primitives are obtained by first computing the optical flow of the input image sequence using Lucas & Kanade's algorithm [12]. The optical flow computed between $I(t)$ and $I(t + 1)$ is denoted as $F(t)$. To obtain a representation consistent with the shape primitives, F is converted to a set of response maps for different "flow primitives" each with different preferred flow direction. We use $f_{x,y,o,s}$ to represent a flow primitive located at position (x, y), orientation o and scale s. Let $\Lambda(x, y, s)$ be the domain of $f_{x,y,o,s}$, which is a small window centered at (x, y, s). The filter response is the sum of projection lengths of all flow vectors within $F_{\Lambda(x,y,s)}(t)$ onto direction o (half-wave rectified):

$$r^{flow}(F(t); f_{x,y,o,s}) = \sum_{ij \in F_{\Lambda(x,y,s)}(t)} |D(v_{ij}, \theta_{ij}, o)| \tag{2}$$
$$D(\theta_{ij}, v_{ij}, o) = \{\frac{1}{2}[1 + \cos(\theta_{ij} - o)]\}^2 * v_{ij}$$

where (v_{ij}, θ_{ij}) is the scale and direction of a flow vector at point (i, j), o is the direction preferred by the filter. A large value of $r^{flow}(F(t); f_{x,y,o,s})$ implies large absolute speed of direction o inside $F_{\Lambda(x,y,s)}(t)$, which means our model encourages flow primitives with hight speed. This is slightly different from previous methods used in [6, 17], where optical flow maps are computed according to grids in both direction and speed. We find, through experiments, that our method is more robust in presence of background motions. The explanation is simple: high speed primitives often locates on or close to foreground objects, whereas low speed ones are more likely to be on the background.

2.2. Deformable template for image

The backbone of our action template model is the active basis model [22], which can be summarized as following:

$$I_m = \sum_{i=1}^{n} c_{m,i} B_{m,i} + \epsilon_m, \tag{3}$$
$$B_{m,i} \approx B_i,\ i = 1, ..., n.$$

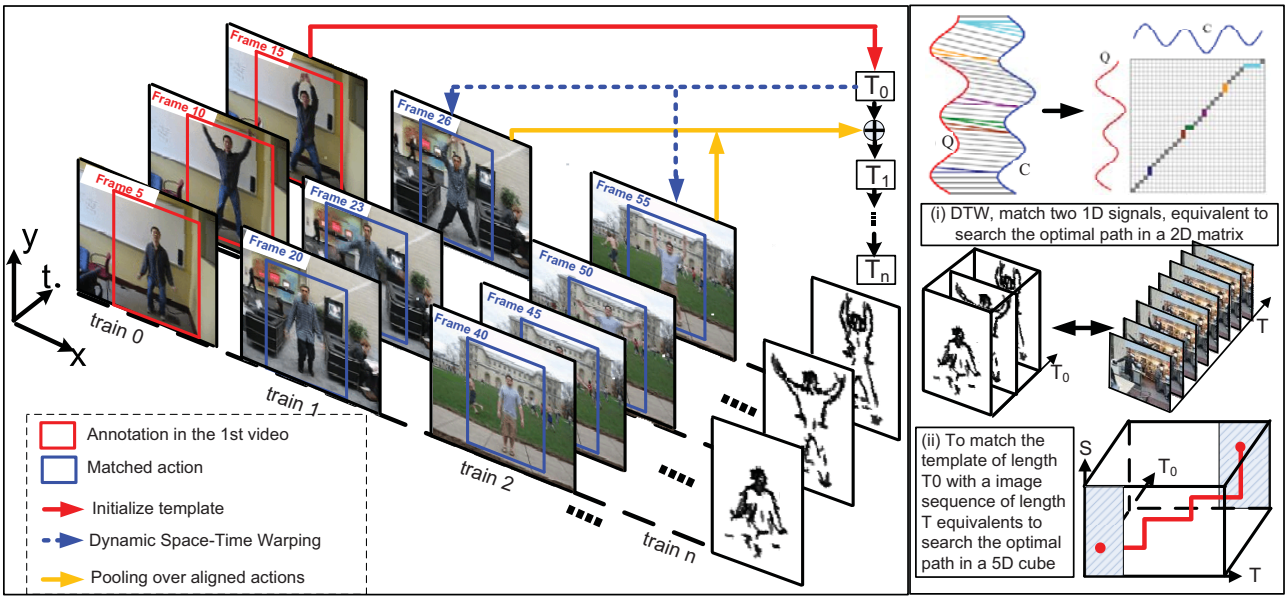
where $\{I_m, m = 1, \ldots, M\}$ is a set of training images, $(c_{m,i}, i = 1, ..., n)$ are coefficients, $B_i \in \Omega$, $B_{m,i} \in \Omega$, $\Omega = \{B_{x,y,s,o}, \forall(x, y, s, o)\}$, $B_{m,i}$, is a slightly perturbed version of B_i within range $(d_{m,i}, \delta_{m,i})$ such that $d_{m,i} \in [-b_1, b_1]$, $\delta_{m,i} \in [-b_2, b_2]$. That is, we allow B_i to shift its location along its normal direction, and we also allow B_i to shift its orientation. b_1 and b_2 are the bounds for the allowed displacement in location and turn in orientation (e.g., $b_1 = 6$ pixels, and $b_2 = \pi/15$). Therefore, the deformable template is the active basis $T = (B_i, i = 1, ..., n)$. The deformed template is $T_m = (B_{m,i}, i = 1, ..., n)$. We may use the least squares criterion to find the best set of primitives:

$$(\mathbf{T}, c)^* = \arg\min_{\mathbf{T},c} \sum_{m=1}^{M} \|I_m - \sum_{i=1}^{n} c_{m,i} B_{m,i}\|^2 \tag{4}$$

This is equivalent to the maximum-likelihood estimation of the following probability over training examples $\{I_m\}$ [22]:

$$p(I_m|T_m) = q(I_m) \prod_{i=1}^{n} [\exp\{\lambda_i r_{m,i}\} z_i^{-1}],$$

where $q(I_m)$ is the density of white noise model, $r_{m,i} = Sigmoid(|\langle I_m, B_{m,i}\rangle|^2)$, $z_i = E_q[\exp\{\lambda_i r_{m,i}\}]$, is the normalizing constant for the i-th term, λ_i is weight,.



(a) Semi-supervised learning steps (b) Dynamic Space-time Warping

Figure 3. (a)Semi-supervised learning, starting from a labeled action video (red bounding box), a template T_0 is initialized. This template is then matched to other video clips by dynamic space-time warping, a new template T_1 is learned by pooling over matched videos. This process iterates several times to generate an optimal template T_n. (b) An illustration of the dynamic space time warping algorithm.

The above formulation can be extended to a mixed template of both shape and flow primitives, if we use I_m to represent either image I or optical flow F, use $\mathbf{B}$ to represent either shape primitive B or flow primitive f and use $r_{m,i}$ to represent either $r^{shape}(I(t); B_{m,i})$ in Eq.(1) or $r^{flow}(F(t); f_{m,i})$ in Eq.(2). (see [20] for another example of mixed template model with more details). For clarity, we will use such representations for the rest of this paper and stick with the least-square formulations (Eq.4).

2.3. Deformable template for action

The active basis model works on roughly aligned images of an object class. We can use the same model for videos of a human action class if we find their spatial-temporal alignments. Given a set of weakly supervised training videos $\{V_m = \mathbf{I}_m[0, t_m], m = 1, \ldots, M\}$. We first assume they are temporally aligned (i.e. actions are synchronized). Therefore, we only need to consider position (x, y) and scale s of the actor. (x, y, s) should be continuous functions of time t. V_m can be interpreted as generated from an underlying template process $\{T(t)\}$ corrupted by an additive white zero-mean Gaussian "noise" $\epsilon_m(t)$.

$$\begin{aligned} T(t, \mathcal{S}_m) &\approx T_m(t, \mathcal{S}_m) = \sum_{i=1}^{n} c_{i,m,t} B_{i,m,t}(\mathcal{S}_m(t)) \\ I_m(t) &= T(t, \mathcal{S}_m) + \epsilon_m(t), t \in [0, t_m] \end{aligned} \quad (5)$$

where $\mathcal{S}_m(t) = [x_m(t), y_m(t), s_m(t)]$ denotes the position and scale of the actor at frame t, $T_m(t, \mathcal{S}_m)$ is the locally deformed template process at frame t, $B_{i,m,t}$ and $c_{i,m,t}$ are locally deformed primitives and coefficients respectively. The least squares condition in Eq.(4) can be rewritten as:

$$(T, \mathcal{S})^* = \min_{T \in \Omega, \mathcal{S} \in \Psi} \sum_{m=1}^{M} \int_0^{t_m} \|I_m(t) - T(t, \mathcal{S}_m)\|^2 dt \quad (6)$$

subject to Eq.(5), where $\Omega = \{T(t)\}$ is the template process, Ψ is a set of all possible trajectories, $\mathcal{S}^*$ are the best trajectories of all videos. Since this is an over-determined problem, we must introduce regularizers [8], for instance of the form $\phi_{reg}(\mathcal{S}) = \int_0^{t_m} \|\nabla \mathcal{S}\| dt$. It enforces smoothness of the trajectory. The RHS of Eq.(6) is now:

$$\min_{T, \mathcal{S}} \sum_{m=1}^{M} \int_0^{t_m} \|I_m(t) - T(t, \mathcal{S}_m)\|^2 + \lambda \|\nabla \mathcal{S}_m\| dt \quad (7)$$

where λ is a Largrange multiplier.

We now remove the assumption of synchronization. To account for different temporal pattern of actors, we need introduce a time warping function for each video. Consider an arbitrary infinite-dimensional diffeomorphism w of the interval $[0, t_m]$, so that Eq.(5) becomes:

$$I_m(w_m(t)) = T(t, \mathcal{S}_m) + \epsilon_m(t), t \in [0, t_m] \quad (8)$$

In order for $w(t)$ to be a viable temporal index, it must satisfy two properties: continuity and causality. The ordering of time instants has to be preserved by the time warping.

This can be formalized by imposing that w be continuous and monotonic. We define a set $\Gamma = \{w(t)\}$ be all valid time warping functions. Then we can rewrite Eq.(7) into:

$$\min_{T\in\Omega,\mathcal{S}\in\Psi,w\in\Gamma} \sum_{m=1}^{M} \int_0^{t_m} \|I_m(w_m(t)) - T(t,\mathcal{S}_m)\|^2 + \lambda\|\nabla\mathcal{S}_m\|dt \quad (9)$$

This equation defines the Deformable Action Template (DAT) model studied in this paper. We will introduce an algorithm to learn the model parameters in the next section.

3. Learning deformable action template from cluttered videos

To minimize Eq.(9), we can decouple its parameters into two groups. The first group contains the variable indicating the selection of action template T, which includes selection of primitives $B_{x,y,o,s}$ and weights $c_{x,y,o,s}$ at each frame. This is shared by all training videos. The second group includes the time warping functions w and trajectories $\mathcal{S}$, and is different for each training video. In an ideal supervised-learning setting, group 2 should be derived by manual annotation. But it is extremely tedious to label human actions from real-world videos for two reasons: 1) One bounding box each frame is required to offset global motion; 2) Frame-to-frame correspondences are required to synchronize actions. Therefore, the supervised learning approach is of limited practical use. Instead, we developed a semi-supervised learning approach (see Fig.3 for an illustration diagram): For each human action class, we annotate one training video and use it to learn an initial action template T_0. The template is first used to detect actions from other training videos by a dynamic space-time warping algorithm, and is then updated by a shared pursuit algorithm over all aligned videos. Our algorithm iterates several times over these two steps to get an optimal template.

3.1. Dynamic space-time warping algorithm

Given T_0, the objective function Eq.(9) can be solved globally using dynamic programming. The procedure is based on an technique call Dynamic Time Warping. The original DTW solve the problem of finding a temporal match between two 1 dimensional signal Q and C, as shown in Fig.3 (b), the matching process is equivalent to finding an optimal path in a $[T_1 \times T_2]$ matrix, therefore could be solved by dynamic programming, where T_1 and T_2 are the length of signal Q and C respectively. The path is the warping function w in our definition, which satisfies time continuity and causality. We want to optimize $\mathcal{S}$, which happens to be a continuous function too, together with w. As illustrated in Fig.3 (b), this problem can also be interpreted as trying to find an optimal path in a 5 dimensional space (x,y,s,$t_{template}$,t_{target}). In Fig.3(b), we draw a 3D cube for illustration purpose. In the original DTW algorithm [15], the starting and ending points of two signals are aligned. But we cannot assume the same thing for template and target videos. We solve this problem by searching the starting and ending points on two surfaces of the cube (as illustrated by shaded areas in Fig.3(b)).

3.2. Shared pursuit algorithm

As illustrated in the Fig.3(a), given a set of aligned videos (i.e. $\mathcal{S}$ and w known), shape and motion primitives are selected sequentially from a dictionary of Gabor wavelets and flow filters for each frame. Primitives shared by most examples are first selected, which equivalents to minimizing the summed squared error. Each selected primitive explains a small patch in either $I(t)$ or $F(t)$, and inhibits other primitives nearby to be selected again. We only give a sketchy description of the shared pursuit algorithm here as it is not a new point of our paper. Interested readers please refer to [22, 20] for details about how to adaptively select primitives and learn weights. The algorithm can be applied on a single video and is used to initialize the template of our semi-supervised learning procedure.

4. Experiments

Parameter values: The original size of Gabor wavelets and optical flow filters are both 9 pixels, and the scale factor $s = [0.8, 0.9, 1.0, 1.1, 1.2]$. (x,y) is sub-sampled every 2 pixels. The orientation of Gabor filters takes 15 equally spaced angles in $[0, \pi]$, the direction of flow filters take 12 equally spaced angles in $[0, 2\pi]$. The shift along the normal direction $[-b_1, b_1] = [-4, -2, 0, 2, 4]$ pixels. The shift of orientation $[-b_2, b_2] = [-1, 0, 1]$ grades. The smoothness constraint $\lambda = 100$ is set empirically, and proved to be good for all categories.

Dataset: Our experiments mainly focused on the CMU action dataset [7], which has approximately 20 minutes of video containing 110 events of interest separated into 5 categories, namely "jumping jacks", "pick up","push elevate button","single hand wave" and "two handed wave". Videos were downscaled to 160×120 in resolution. Each category contains a template video, which is used as initialization to our learning algorithm. The templates are manually segmented using a bounding box typically $80\times120\times50$ in size. Each category contains about 15 to 30 testing events, which is annotated for performance evaluation.

Experiment 1: Learning deformable action templates from semi-supervised dataset. We apply the iterative learning algorithm on the above mentioned dataset and learn a separate deformable action template for each category. Beside the initialization video, we randomly select 10 clips containing the action-of-interest as training data. Typical

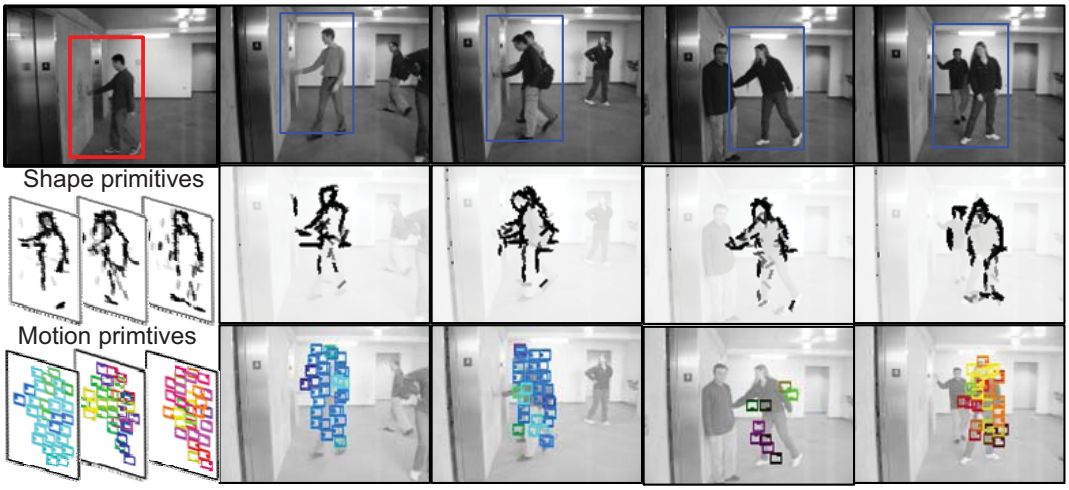


Figure 4. Action template and deformed versions for "pushing button" action. Illustrations same as Fig.1.

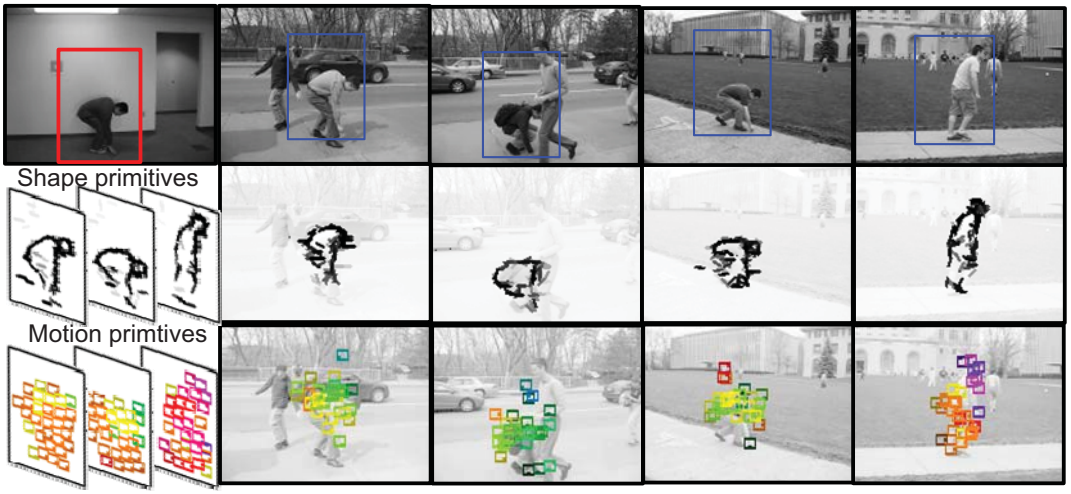


Figure 5. Action template and deformed templates for "pick up" action. Illustration same as Fig.1.

examples of learned deformable action templates are shown in Fig.(1,5, 4 and 9). The first row of these figures (except Fig.1) illustrates the training videos, where the initializing clip is marked out with a red bounding box. The second row shows the shape primitives of the learned action template and their deformed versions corresponding to each training video. Similarly, the 3rd row shows the flow primitives and their deformed versions. The learning algorithm stops after three iterations. Difference in choosing unlabeled training clips will slightly affect the results of learned templates, but in most cases the iterations converge to a reasonable results. (See next section for quantitative analysis).

In Fig.7, we illustrate one example of temporal deformations generated by the DSTW algorithm. The left panel displays frames both from the template and a target video. The actual frame number of each column is marked on its top. Temporal correspondences between template and target frames are illustrated by the red lines between them. The right panel is a matrix representing the time warping path, where T and T_0 axis stand for the temporal domain of the target video and template respectively. In side the matrix, the white zig-zag line stands for the time warping path. The dark area denotes the space of all possible time warping paths. The gray area represents for impossible search range, which is setup manually to save computation time. We can find that the actor in the target video hold the bending-pose longer than the template. Therefore, the time warping path mainly goes horizontally in the area marked by the dotted red rectangle (implies that one template frame is associated with several target frames).

Experiment 2: Event detection. We use the learned templates from the experiment 1 to detect actions from all testing videos from the CMU dataset. To save time, we use the DSTW algorithm to detect multiple events at a single

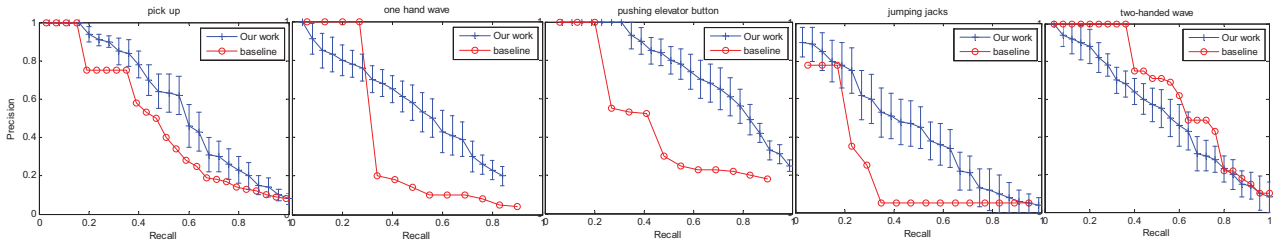


Figure 6. Event Detection Precision Recall Curves. The red curve is the base line performance reported in [7]. The blue curve is the detection performance averaged over 5 separate experiment with different random sample of unlabeled training examples. The error bonds illustrate best and worst detection performance

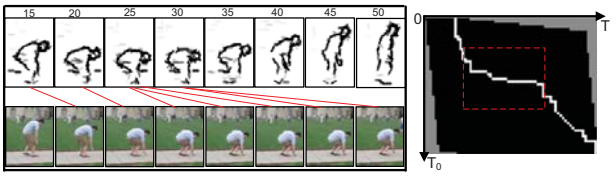


Figure 7. Illustration of temporal deformations, see text for details

search. The rationale is that although dynamic programming only yields one optimal path at a time, paths that are not overlapping heavily in space-time barely affect each other. In specific, we collected all the paths with a higher-than-threshold matching score (normalized by path length) and eliminated paths that overlap more than 20% in time (thanks to the factor that most actions in the dataset do not happen together). Ideally, we can detect all actions from a target video clip with a single run of the DSTW algorithm. But because of memory limitation, we divided target video clips into small clips of 100 frames each (overlapping 20% with its neighbors) and tested separately. We generate the precision-recall (P-R) curve of each action, which is illustrated by the blue curves in Fig.6, by changing the threshold. The red curves are baseline detection algorithm reported in [7]. To consider a detection, we use the same decision criterion as in [7]. Our performance curve is computed by averaging detection results from 5 separate runs with random selection of training videos, error bounds illustrate the best and worst cases when doing the random selection. The computation time of each 100 frame clip is about 2-3 minutes on an Intel 2.5 GHz PC.

Experiment 3: Contribution of shape and flow primitives. To analyze the contribution of shape and flow primitives separately, we conducted the experiment 2 using only one kind of primitives at a time. Fig.8 shows an intuitive result of how shape and flow primitives work together. In scene (a), it is very hard to separate the actor from background crowds using only shape primitives, which leads to detection failure. With the help of flow primitives in (c), it is easy to separate the vertical motion of the actor from background motion, which is predominantly horizontal. In scene (b), the motion generated by the moving car in the background confuses with foreground. However, the actress is detected in (c) by integrating shape information. Table.1 shows quantitative performance of separated channels in terms of average area under the P-R curve (AUC). It shows that shape primitives are generally better than flow primitives. But flow primitives are also crucial in classes such as "jumping-jack" and "hand-waving". We believe it is because motion patterns in these classes are special.

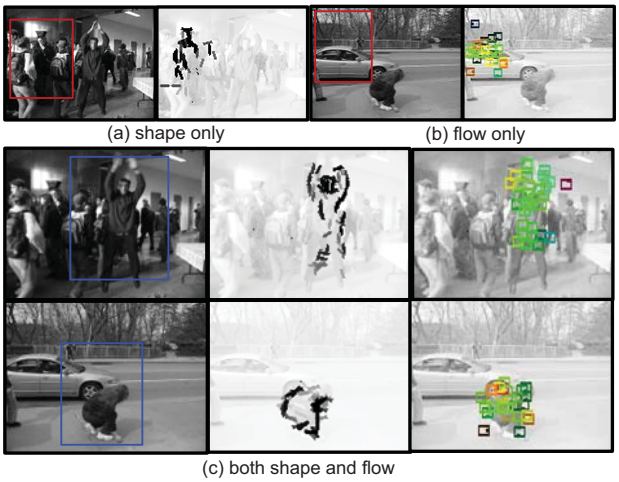


Figure 8. (a) and (b) show two failure detection results (red rectangles) using only one kind of primitives. (c) shows correct detection results (blue rectangles) by combining shape and flow primitives.

Table 1. Average AUC of separated channels

Category	Shape	Flow	Both	baseline[7]
Pick up	0.53	0.21	0.58	0.47
One-hand wave	0.46	0.29	0.59	0.38
Push button	0.68	0.12	0.74	0.48
Jumping jack	0.34	0.31	0.43	0.22
Two-hand wave	0.44	0.27	0.53	0.64

Experiment 4: Action classification on KTH dataset. We tested our algorithm's ability in handling deformation by performing classification the KTH human action dataset

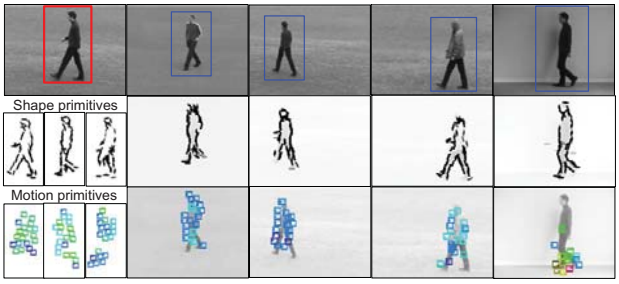


Figure 9. Action template for walking. Illustration same as Fig.1.

[18]. This dataset has little background clutter but is richer in variations (lighting, cloth, viewing angle, etc.). The complete set of actions was recorded under 4 different conditions (s1-s4) (see [18] for details). Since we only need one repeat for each clip, we use a similar experiment setting as in [17]: All sequences are trimmed to 30 frames and flipped a same direction. All evaluations were done with 5-fold cross-validations: 4 folders for training, 1 for testing. The results were computed by averaging 5 permutations. Table.2 (1st and 2nd columns) shows the average correct recognition rates of our method, which is comparable to state-of-the-art algorithms. Similar to [17], we also conducted experiments separately under s1-s4. The results are also shown in Table.2. Our method achieves better result on s2 (outdoors with scale variations) than [17], which indicates the benefit of modeling scale variations explicitly. Fig.9 illustrates the learned deformable templates of walking class and shows that our model is able to capture shape deformations as well as scale and view point variations.

5. Conclusion

We present a generative model of action template composed of deformable shape and flow primitives. We also propose an efficient learning algorithm that both the adaptive basis selection and spatial-temporal alignment can be solved together in a unified maximum likelihood framework. We demonstrate experiments that show promising results of our methods.

Table 2. Average recognition precision of on KTH

	all	s1	s2	s3	s4
Our method	87.8%	90.1%	**84.5%**	86.1%	91.3%
SNIPPET 7[17]	90.9%	93.0%	81.1%	92.1%	96.7%
JHUANG[6]	91.7%				
NEIBLES[13]	81.5%				

Acknowledgement We thank Prof. Ying-Nian Wu and Qiongchen Wang for very helpful discussions and suggestions. This work was supported by NSF grants IIS-0713652 and DMS-0707055, ONR grant N00014-07-M-0287, NSF China grant 60832004 and 863 project 2008AA01Z126.

References

[1] A. Bobick and J. Davis. The recognition of human movement using temporal templates. *PAMI*, 2001. 1, 2

[2] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *PETS05*, 2005. 2

[3] A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *ICCV*, pages 726–733, 2003. 2

[4] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61, 2005. 2

[5] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *PAMI*, 2007. 1, 2

[6] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *ICCV*, 2007. 1, 2, 3, 8

[7] Y. Ke, R. Sukthankar, and M. Hebert. Event detection in crowded videos. In *ICCV*, pages 1–8, 2007. 1, 2, 5, 7

[8] A. Kirsch. *An introduction to the mathematical theory of inverse problems*. Springer Verlag, 1996. 4

[9] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008. 2

[10] J. Liu, J. Luo, and M. Shah. Recognizing Realistic Actions from Videos in the Wild. In *CVPR*, 2009. 2

[11] J. G. Liu, S. Ali, and M. Shah. Recognizing human actions using multiple features. In *CVPR*, 2008. 1

[12] B. D. Lucas and T. Kanade. Optical navigation by the method of differences. In *IJCAI*, pages 981–984, 1985. 3

[13] J. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *IJCV*, 79(3):299–318, 2008. 2, 8

[14] D. Ramanan and D. A. Forsyth. Automatic annotation of everyday movements. In *NIPS*, 2003. 2

[15] J. O. Ramsay and B. W. Silverman. *Functional Data Analysis*. Springer, June 2005. 2, 5

[16] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. In *Nature:Neuroscience*, 1999. 2

[17] K. Schindler and L. van Gool. Action snippets: How many frames does human action recognition require? In *CVPR*, 08. 1, 2, 3, 8

[18] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *ICPR*, 2004. 8

[19] E. Shechtman and M. Irani. Space-time behavior-based correlation - or - how to tell if two underlying motion fields are similar without computing them? *PAMI*, 2007. 2

[20] Z. Si, H. Gong, Y. Wu, and S. Zhu. Learning mixed templates for object recognition. In *CVPR*, 2009. 4, 5

[21] J. Sullivan and S. Carlsson. Recognizing and tracking human action. In *ECCV*, 2002. 2

[22] Y. N. Wu, Z. Z. Si, C. Fleming, and S. C. Zhu. Deformable template as active basis. In *ICCV*, 2007. 2, 3, 5

[23] A. Yilmaz and M. Shah. Actions sketch: A novel action representation. In *CVPR*, pages I: 984–989, 2005. 1, 2

[24] J. Yuan, Z. Liu, and Y. Wu. Discriminative subvolume search for efficient action detection. In *CVPR*, 2009. 2