Action Detection in Cluttered Video with Successive Convex Matching

Hao Jiang, Mark S. Drew, and Ze-Nian Li

Abstract—**We propose a novel successive convex matching method for human action detection in cluttered video. Human actions are represented as sequences of poses, and specific actions are detected by matching pose sequences. Since we represent actions as the evolution of poses and shapes, the proposed method can detect actions in videos that involve fast camera motions. Template sequence to video registration is nonlinear and highly nonconvex. Instead of directly solving the hard problem, our method convexifies it into a sequence of linear programs and refines the matching by successive trust region shrinkage. The proposed scheme further simplifies the linear programs by representing the target point space with a small set of basis points. The low complexity of the proposed method enables it to search efficiently in a large range. Experiments show that successive convex matching can robustly match a sequence of coupled shape templates simultaneously to target sequences and effectively detect specific actions in cluttered videos.**

Index Terms—**Action detection, matching, optimization.**

I. Introduction

ACTION DETECTION in a controlled environment has been intensively studied, and different real-time systems have been implemented [1]–[3]. These systems use fixed cameras to facilitate foreground extraction or use magnetic/optical markers for movement extraction. Finding actions in a video recorded in an uncontrolled environment, important for surveillance, content based video retrieval and robotic vision, is still a largely unsolved problem. The main difficulty for action recognition in general video is that there is no effective way to segment an object in such videos. Therefore, we have to be able to locate the object and at the same time detect its action. Factors such as the articulation of the human body, variability of clothing, and background clutter make action recognition a challenging task.

We propose a method to detect a specific human action in such an uncontrolled setting. By representing an action as a sequence of body poses with specific temporal constraints, we search for a given action by matching to the video a sequence of coupled body pose templates. The matching is formulated as an energy minimization problem. The goal is to minimize the matching cost subject to consistency constraints, viz. intraframe feature matching and a low degree of change in the object center's relative position across video frames. The interobject center continuity constraint is important in that it enforces matches in different video frames to stick to a single object in a cluttered video, where multiple objects may indeed appear.

Even though many feature matching schemes have been proposed, they are not sufficient to solve the above-formulated optimization problem. As shown in our experiments, a greedy scheme such as iterative conditional modes (ICM) [4] is not robust enough to match the target if it has large deformation from the template or there is strong clutter in the background. Robust matching methods such as graph cuts [5], belief propagation [6] and, more recently, a linear programming (LP) relaxation scheme [7] have been studied for finding correspondence in single image pairs using pairwise constraints. But these methods are not easily extended to include the center continuity constraint in matching a template sequence to a target video.

We consider a more efficient approach—a successive convex matching scheme—for registering template image sequences to targets in video [17]. We follow an LP relaxation scheme [28] that has been applied to object detection, motion estimation and tracking, reshaping the problem so that the interframe constraint can be introduced. Instead of directly solving the optimal matching problem, the proposed scheme converts the optimization into easier convex problems which can be solved by linear programming. An iterative process updates the trust region and successively improves the approximation. This convex matching scheme has many useful features: it involves only a small set of basis target points, and it is a strong approximation scheme. It is also found to be robust against strong clutter and large deformations, necessary for success of an action recognition scheme. After template to video registration, we compare the similarity of the matching targets in video with the templates using matching cost and degree of deformation.

Finding people and recognizing human actions is a research area with a long history in vision research. Searching for static poses [8]–[10], [32] has been intensively studied. For action recognition, the trajectories of body joints are natural features. Even if the body joints can be accurately detected from video, action recognition based on trajectories is nontrivial [23].

Manuscript received August 19, 2008; revised December 13, 2008 and February 16, 2009. First version published July 7, 2009; current version published January 7, 2010. This paper was recommended by Associate Editor D. Schonfeld.

H. Jiang is with the Department of Computer Science, Boston College, Chestnut Hill, MA 02467 USA (e-mail: hjiang@cs.bc.edu).

M. S. Drew and Z.-N. Li are with the School of Computing Science, Simon Fraser University, Vancouver, British Columbia V5A 1S6, Canada (e-mail: mark@cs.sfu.ca; li@cs.sfu.ca).

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/TCSVT.2009.2026947

Invariants from these trajectories have been used for action classification [21], [26]. Most previous action recognition methods rely on foreground and background segmentation. Space-time shape [27], motion object [25], and motion history volume [24] have been used in action detection in situations where a relatively clean silhouette could be extracted. Motion patterns have also been widely used to find actions, because motion is resistant to clothing change. Motion histogram is used for detecting actions in movies [20]. Spatial structure of motion field can be included to improve the performance, which has been used to detect human actions from a distance [11]. Properly formulated, motion can also be matched without explicit motion estimation [12]. Recently, we propose a motion matching method for action detection using shape flows [31]. Even though motion is a reliable feature for static or slow motion camera settings, large camera motion will introduce a good deal of difficulty for motion based action recognition systems. Appearance based schemes can be used to relieve the problem. In such schemes, an appearance model is explicitly matched to a target video sequence for action detection. One approach is to recognize action based on body part model [13]–[15]. Detecting human body configuration based on smaller local features is another appearance matching method [8], [10], [19], [22]. Recently, shape models based on superpixels and tree relations among parts are used to find actions in clutter [30].

In this paper, we follow the appearance matching direction, and go on to propose and investigate a convex method for video sequence matching. Instead of using body parts, we match frames by using easily detected local features, with an intra-frame pairwise feature matching constraint and an inter-frame position constraint over a longer time interval, thus enabling the scheme to detect complex actions.

II. Matching Templates to Video

We formulate the sequence matching problem as follows. Assume that there are n templates extracted from a reference video sequence, which represent key poses of an object in some specific action. We label the templates from 1 to n. Template i is represented as a set of feature points S_i, a feature vector for each feature point and the set of neighboring pairs $\mathcal{N}_i$. Set $\mathcal{N}_i$ consists of all the pairs of feature points in S_i connected by edges in the Delaunay graph of S_i. Fig. 1 illustrates intra-frame and inter-frame constrained deformable video matching. Matching a template sequence to a video is formulated as an optimization problem. We search for a matching function $\mathbf{f}$ to minimize the following objective function:

$$\min_{\mathbf{f}} \left\{ \sum_{i=1}^{n} \sum_{\mathbf{s}\in S_i} C^i(\mathbf{s}, \mathbf{f}^i_{\mathbf{s}}) + \lambda \sum_{i=1}^{n} \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}_i} d(\mathbf{f}^i_{\mathbf{p}} - \mathbf{p}, \mathbf{f}^i_{\mathbf{q}} - \mathbf{q}) + \mu \sum_{i=1}^{n-1} d(\bar{\mathbf{s}}^{(i+1)} - \bar{\mathbf{s}}^i, \bar{\mathbf{f}}^{(i+1)} - \bar{\mathbf{f}}^i) \right\}.$$

Here, $C^i(\mathbf{s}, \mathbf{f}^i_{\mathbf{s}})$ is the cost of matching feature point $\mathbf{s}$ in template i to point $\mathbf{f}^i_{\mathbf{s}}$ in a target frame; $\bar{\mathbf{f}}^i$ and $\bar{\mathbf{s}}^i$ are centers of

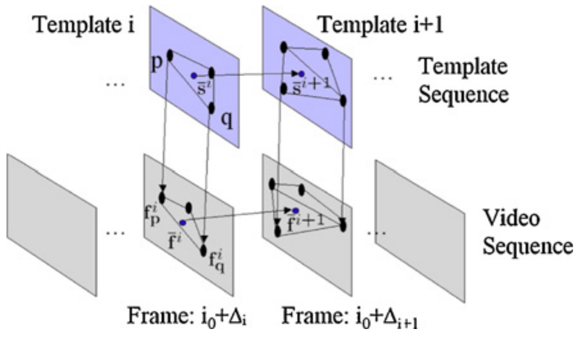


Fig. 1. Deformable video matching. In this example, template i and template $i+1$ match the target frames $i_0+\Delta_i$ and $i_0+\Delta_{i+1}$. The matching is constrained by both intra-frame feature spatial layout and inter-frame object center relative location.

the matching target points and template points, respectively, for the ith template. The first term in the objective function represents the cost of a specific matching configuration. The second and third terms are intra-frame and inter-frame regularization terms respectively. The coefficients λ and μ are used to control the weight of the regularization terms. In this paper, we focus on problems in which $d(\cdot,\cdot)$ is defined as an L_1 norm. As will be shown later, in this case a linear programming relaxation of the problem can be constructed. To simplify the matching process, we enforce that target points for one template cannot be dispersed into several target frames. The matching frame for template i is specified as $i_0+\ \Delta_i$, in which i_0 is a start frame number and Δ_i is the time stamp of a template frame.

The above optimization problem is non-linear and usually non-convex, because matching cost functions $C^i(\mathbf{s}, \mathbf{t})$ are usually highly non-convex with respect to $\mathbf{t}$ in real applications. Searching for the optimum of a non-convex function is hard because many local minima exist. Simple greedy methods are easily trapped in a locally best solution. On the other hand, exhaustive search is infeasible for a large scale problem. In the following, we discuss feature selection and methods to cast the non-convex optimization problem into a sequential convex programming problem, so that a robust and efficient optimization solution can be obtained.

A. Features for Matching

To match articulated objects, we need to choose features that are at the same time not sensitive to colors and robust to deformations. Even though different feature types can be used, here we use edge features to demonstrate the usefulness of the matching scheme. Edge maps have been found to be very useful for object class detection, especially matching human objects [10]. To increase matching robustness, instead of directly matching edge maps, we match a transformed edge map. A distance transform [29] is applied to turn edge maps into a greyscale representation in which values are the distances to the nearest edge pixels. The distance transform image is then fully rectified to generate a truncated distance transform image. Fig. 2(c) and (d) are the truncated distance transform of the edge maps of the images in Fig. 2(a) and (b). Small image patches on these truncated distance transform images

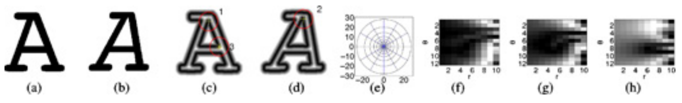


Fig. 2. Log-polar features. (a) and (b) Test images. (c) and (d) Truncated distance transform images. (e) Log-polar bins. (f), (g), and (h) Log-polar features at locations 1, 2, and 3, respectively.

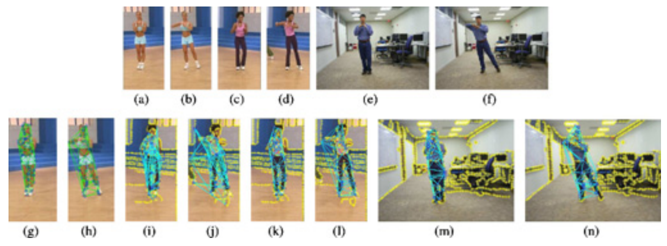


Fig. 3. Matching using log-polar features. (a) and (b) Starting and ending frames of the template action. (c) and (d) Starting and ending frames of a target action. (e) and (f) Starting and ending frames of another target action. (g) and (h) Template mesh and features overlaid on top of the template images. (i) and (j) Matching using a greedy method. (k), (l), (m), and (n) Matching using successive convex matching. Blue circles in (i)–(n) are matched target points. The yellow crosses indicate the potential target points for each site in the templates. Pairwise matched features themselves are not enough for reliable matching.

are found to provide good features in matching. To make the local features incorporate more context, we calculate the log-polar transform of the distance transform image centered on selected feature points in the target and template images. Log-polar transform re-samples the image centered at a given point using log-polar coordinates. The coordinate transform from the Euclidean frame xy to the log-polar frame $r\theta$ is

$$r = \alpha \ln((x - x_0)^2 + (y - y_0)^2)$$
$$\theta = \tan^{-1}((y - y_0)/(x - x_0))$$

where (x_0, y_0) is the center of transformation and α is a constant coefficient. Fig. 2(e) shows a typical log-polar partition of a circular area into bins assuming that the log-polar coordinates are linear. In this example, r has 10 levels, θ has 12 levels, and α is 6. The log-polar transform simulates the human visual system's foveate property and puts more focus in the center view than the periphery views. This feature is similar to the blurred edge features in [7] for object class detection. Fig. 2(f)–(h) show log-polar features for points at locations 1, 2, and 3 in Fig. 2(c) and (d). Here, we abuse the notion of θ; we also use θ to denote its discrete index. Even though the object deforms, the corresponding local features at locations 1 and 2 are quite similar, while they are distinct from features at other locations such as the one at location 3.

Fig. 3 illustrates an example of matching log-polar features. This example also illustrates how the proposed successive convex matching finds similar actions. Before going into details of the algorithms, we show the context and application using a comparatively simple example. We wish to find the correspondence between the target object in Fig. 3(c)–(f) and the template object in Fig. 3(a) and (b). As shown in Fig. 3(g) and (h), log-polar features are computed on randomly selected edge pixels on the template object. Fig. 3(i) and (j) show the greedy matches using the log-polar feature for the fitness images. For each template point, greedy matching finds the target point with the lowest matching cost. The matching cost is simply the Euclidean distance between the template log-polar feature and the target feature. The log-polar feature increases robustness in matching but nevertheless, without a robust scheme, the matching is still very likely to fail. Using the proposed successive convex matching we incorporate global spatial constraints and thus achieve more robust results—the result of the convex matching for this example is shown in Fig. 3(k)–(n).

B. Linear Programming Relaxation and Simplex Method

In this section, we propose methods to relax the original optimization problem into a sequence of simpler linear programs which can be efficiently solved. The following scheme is extended from the single frame object matching scheme [28].

The first step of linear relaxation is to linearize each term in the objective function. To linearize the matching cost term, we select a set of *basis target points* for each feature point in a template. Then, a target point can be represented as a linear combination of these basis points, e.g., $\mathbf{f}_\mathbf{s}^i = \sum_{\mathbf{t}\in B_\mathbf{s}^i} w_{\mathbf{s},\mathbf{t}}^i \cdot \mathbf{t}$, where $\mathbf{s}$ is a feature point in template i, and $B_\mathbf{s}^i$ is the basis target point set for $\mathbf{s}$. We will show that the "cheapest" basis set for a feature point consists of the target points corresponding to the matching cost surface's *lower convex hull* vertices. Therefore, $B_\mathbf{s}^i$ is usually much smaller than the whole target point set for feature point $\mathbf{s}$. This is a key step to speed up the algorithm. We then represent the cost term as a linear combination of the costs of basis target points. For template i, the matching cost term can thus be represented as $\sum_{\mathbf{s}\in S_i} \sum_{\mathbf{t}\in B_\mathbf{s}^i} w_{\mathbf{s},\mathbf{t}}^i C^i(\mathbf{s},\mathbf{t})$. A standard linear programming trick of using auxiliary variables can be used further to turn L_1 terms in the objective function into linear functions [16]: we represent each term in $|\cdot|$ as the difference of two non-negative auxiliary variables x^+, x^-, y^+, y^-, u^+, u^- or v^+ and v^-. Substituting this into the constraint, we replace the term in the objective function with the summation of two auxiliary variables. In our formulation, the summation equals the absolute value of the original term when the linear program is indeed optimized.

The complete linear program is written as

$$\min \Bigg\{ \sum_{i=1}^{n} \sum_{\mathbf{s}\in S_i} \sum_{\mathbf{t}\in B_\mathbf{s}^i} w_{\mathbf{s},\mathbf{t}}^i C^i(\mathbf{s},\mathbf{t}) + \lambda \sum_{i=1}^{n} \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}_i} (x_{\mathbf{p},\mathbf{q}}^{i+} + x_{\mathbf{p},\mathbf{q}}^{i-} + y_{\mathbf{p},\mathbf{q}}^{i+} + y_{\mathbf{p},\mathbf{q}}^{i-}) + \mu \sum_{i=1}^{n-1} (u^{i+} + u^{i-} + v^{i+} + v^{i-}) \Bigg\}$$

$$\begin{aligned}
\text{s.t.} \sum_{\mathbf{t}\in B_\mathbf{s}} w_{\mathbf{s},\mathbf{t}}^i &= 1 \quad \forall \mathbf{s} \in S_i \quad i = 1, \ldots, n \\
x_\mathbf{s}^i &= \sum_{\mathbf{t}\in B_\mathbf{s}^i} w_{\mathbf{s},\mathbf{t}}^i \cdot x(\mathbf{t}) \quad y_\mathbf{s}^i = \sum_{\mathbf{t}\in B_\mathbf{s}^i} w_{\mathbf{s},\mathbf{t}}^i \cdot y(\mathbf{t}) \\
x_{\mathbf{p},\mathbf{q}}^{i+} - x_{\mathbf{p},\mathbf{q}}^{i-} &= x_\mathbf{p}^i - x(\mathbf{p}) - x_\mathbf{q}^i + x(\mathbf{q})
\end{aligned}$$

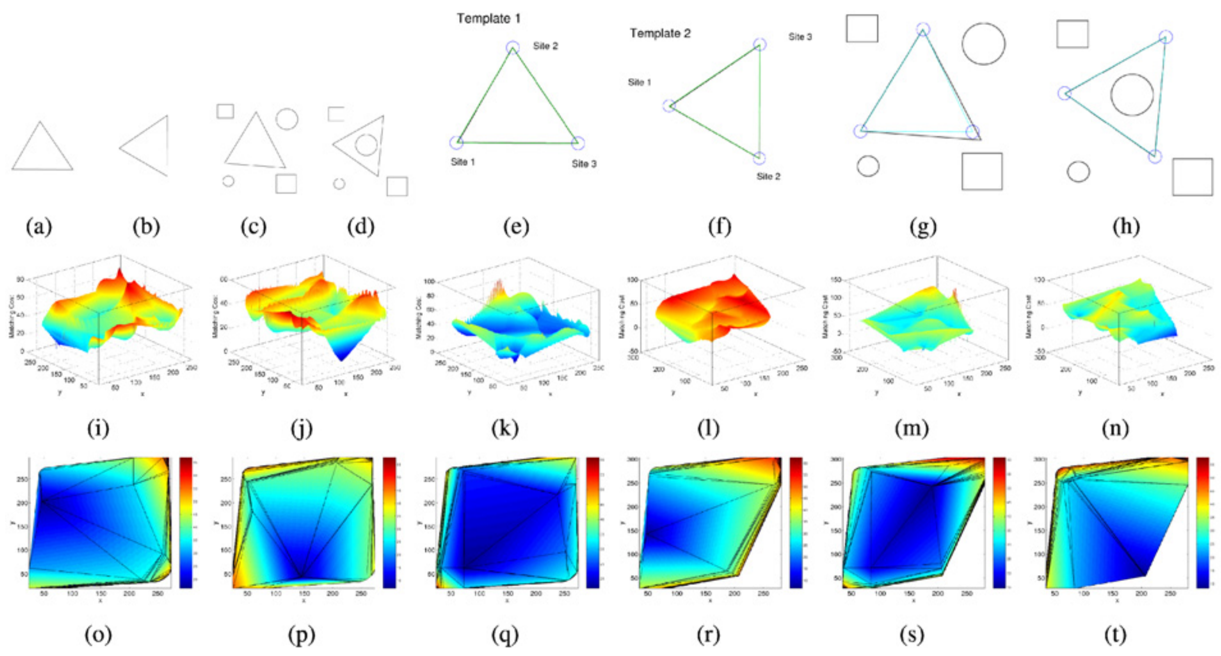


Fig. 4. Example of convex relaxation. (a) and (b) Template images. (c) and (d) Target images. (e) and (f) Feature points and template graph. (g) and (h) Matching result. (i), (j), (k), (l), (m), and (n) Matching cost surfaces for the sites on the template. (o), (p), (q), (r), (s), and (t) Convexified matching cost surfaces.

$$y^{i+}_{\mathbf{p},\mathbf{q}} - y^{i-}_{\mathbf{p},\mathbf{q}} = y^i_{\mathbf{p}} - y(\mathbf{p}) - y^i_{\mathbf{q}} + y(\mathbf{q}) \quad \forall\{\mathbf{p},\mathbf{q}\} \in \mathcal{N}_i \quad i = 1, \ldots, n \tag{1}$$

$$\begin{aligned}
u^{i+} - u^{i-} &= \frac{1}{|S_i|}\sum_{\mathbf{s}\in S_i}[x^i_{\mathbf{s}} - x(\mathbf{s})] - \frac{1}{|S_{i+1}|}\sum_{\mathbf{s}\in S_{i+1}}[x^{i+1}_{\mathbf{s}} - x(\mathbf{s})]\\
v^{i+} - v^{i-} &= \frac{1}{|S_i|}\sum_{\mathbf{s}\in S_i}[y^i_{\mathbf{s}} - y(\mathbf{s})] - \frac{1}{|S_{i+1}|}\sum_{\mathbf{s}\in S_{i+1}}[y^{i+1}_{\mathbf{s}} - y(\mathbf{s})]\\
& \quad i = 1..n-1\\
& \quad \text{all variables } \geq 0.
\end{aligned}$$

Here functions $x(\mathbf{s})$ and $y(\mathbf{s})$ extract the x and y components of point $\mathbf{s}$. The matching result $\mathbf{f}^i_{\mathbf{s}} = (x^i_{\mathbf{s}}, y^i_{\mathbf{s}})$. It is not difficult to verify that either $x^{i+}_{\mathbf{p},\mathbf{q}}$ or $x^{i-}_{\mathbf{p},\mathbf{q}}$ (similarly $y^{i+}_{\mathbf{p},\mathbf{q}}$ or $y^{i-}_{\mathbf{p},\mathbf{q}}$, u^{i+} or u^{i-} and v^{i+} or v^{i-}) will become zero when the linear program achieves its minimum; therefore, we have $x^{i+}_{\mathbf{p},\mathbf{q}}+x^{i-}_{\mathbf{p},\mathbf{q}} = |x^i_{\mathbf{p}} - x(\mathbf{p}) - x^i_{\mathbf{q}} + x(\mathbf{q})|$, $y^{i+}_{\mathbf{p},\mathbf{q}} + y^{i-}_{\mathbf{p},\mathbf{q}} = |y^i_{\mathbf{p}} - y(\mathbf{p}) - y^i_{\mathbf{q}} + y(\mathbf{q})|$, and so on. The second and third regularization terms in the linear program objective function equal the corresponding terms in the original non-linear formulation. In fact, if $B^i_{\mathbf{s}}$ contain all the target points and $w^i_{\mathbf{s},\mathbf{t}}$ are binary variables (0 or 1), the LP becomes an integer programming problem which exactly equals the original non-convex problem. But, integer programming is as hard as the original non-linear optimization, and therefore we are most interested in the relaxed linear program. The linear program has close relation with the continuous extension of the matching problem: the continuous extension is defined by first interpolating the matching cost surfaces $C^i(\mathbf{s}, \mathbf{t})$ piecewise-linearly with respect to $\mathbf{t}$ and then relaxing feasible matching points into a continuous region (the convex hull of the basis target points $B^i_{\mathbf{s}}$).

This linear program has very similar properties to the ones in [28]. The linear relaxation optimizes an approximation convex problem that replaces the matching cost surface of each site with the lower convex hull. We, therefore, can use only the target points and matching costs corresponding to the lower convex hull vertices to construct the linear program without changing the optimum solution. Since the number of lower convex hull vertices is usually much fewer than the number of the matching target points, the searching is much more efficient.

The initial basic variables can be selected in the following way.

1) Only one $w^i_{\mathbf{s},\mathbf{t}}$ is selected as a basic LP variable for each site $\mathbf{s}$ in template i.
2) $x^i_{\mathbf{s}}$, $y^i_{\mathbf{s}}$ are basic LP variables.
3) Whether $x^{i+}_{\mathbf{p},\mathbf{q}}$ or $x^{i-}_{\mathbf{p},\mathbf{q}}$, $y^{i+}_{\mathbf{p},\mathbf{q}}$ or $y^{i-}_{\mathbf{p},\mathbf{q}}$, u^{i+} or u^{i-} and v^{i+} or v^{i-} are basic variables depends on the right-hand side of the constraint; if the right-hand side of a constraint is greater than 0, the plus term is a basic variable; otherwise the minus term becomes a basic variable.

Similar to the equivalent property in [28], if we use a simplex method to optimize the linear program, we will search through a sequence of *triangles* in each target frame. For each site i, the proposed LP relaxation searches only the triangles on the target points corresponding to lower convex hull vertices, in an efficient energy descent manner. (And note that the triangles may be degenerate.) Fig. 4 illustrates the solution procedure of the simplex method for an example two-frame video matching problem. In this simplified example, three features points are selected on the object in Fig. 4(a), (b) respectively and form triangular graph templates, shown in Fig. 4(e), (f). Fig. 4(c) and (d) show the target object in clutter. Fig. 4(g)

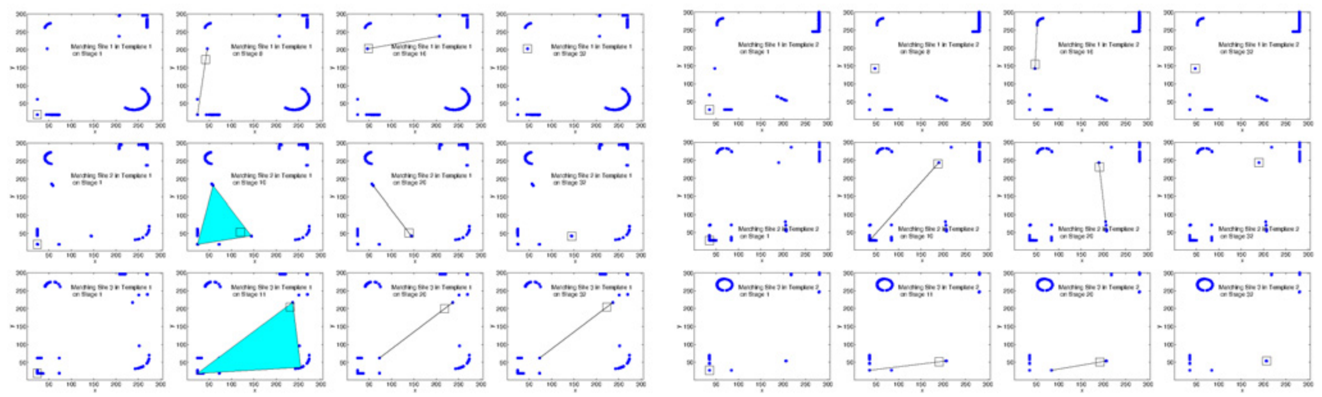


Fig. 5. Searching process of the linear program for Template 1 (left) and Template 2 (right).

and (h) show the matching result. Fig. 4(i), (j), (k), (l), (m), and (n) show the matching cost surface for each of the six points in the template: the matching cost surfaces are highly nonconvex. So directly searching in the cluttered images for the target is a hard problem. Based on the linear relaxation, the nonconvex matching cost surfaces are approximated with their lower convex hulls. Fig. 4(o), (p), (q), (r), (s), and (t) are the lower convex hull surfaces for the respective cost surfaces. As shown in the figure, the convexified surfaces are much simpler than the original cost surfaces and at the same time keep the main structures such as the dominant local minima and the trend of the surfaces. Finding targets using this relaxation is efficient. The searching process (selected from 32 stages) for each site is illustrated in Fig. 5. Each row corresponds to a site in the templates. The blue dots indicate the target points located at the coordinates of the lower convex hull vertices. The searching involves only these target points at each iteration. During the search, the linear program updates a set of basic variables. The basic variables for w correspond to a selection of target points. To illustrate the solution process, we connect the points corresponding to the basic variables by lines. The small rectangle is the weighted linear combination of the target points corresponding to the basic variables at each stage. It indicates the (float) current estimation of the target location for a site. As expected, the proposed LP only checks triangles (filled-blue) or their degenerates (lines or points) formed by basis target points. When the search terminates, the patch generated by the basic variables for each site must correspond to vertices, edges or facets of the lower convex hull for each site. As shown in this example, a single LP relaxation usually has a matching result near the target but the initial match is usually not completely accurate. We can refine the result by successively "convexifying" the matching cost surfaces.

C. Successive Relaxation

As discussed above, a single LP relaxation approximates the original problem's matching cost functions by their lower convex hulls. In real applications, several target points may have equal matching cost and, even worse, some incorrect matches may have lower cost. In this case, because of the convexification process, many local structures are removed which on the one hand facilitates the search process by removing many false local minimums and on the other hand might make the solution not exactly locate on the true global minimum. A successive relaxation method can be used to solve the problem. Instead of one-step LP relaxation, we can construct linear programs iteratively based on the previous searching result and gradually shrink the matching trust region for each site. A trust region for one site is a rectangle area in the target image. Such a scheme can effectively solve the coarse approximation problem in a single LP relaxation step.

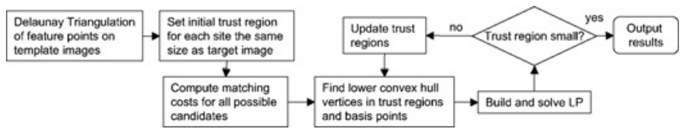

Delaunay Triangulation of feature points on template images
Set initial trust region for each site the same size as target image
Compute matching costs for all possible candidates
Find lower convex hull vertices in trust regions and basis points
Build and solve LP
Trust region small?
yes
no
Output results
Update trust regions

Fig. 6. Successive convex matching.

Introducing a trust region shrinking technique, we go on to use control points to anchor trust regions for the next iteration. We keep the control point in the new trust region for each site and we allow the boundary to shrink inward. If the control point is on the boundary of the previous trust region, other boundaries are moved inward. For the first LP relaxation, the trust region is the whole target image. Then we refine the regions based on previous LPs solution. After we shrink the trust region, the lower convex hull may change for each site. Therefore, we have to find the new target basis and solve a new LP.

We select control points using a consistent rounding process. In consistent rounding, we choose a site randomly and check all the possible discrete target points and select the one that minimizes the nonlinear objective function, by fixing other sites' targets as the current stage LP solution. (This step is similar to a single iteration of an ICM algorithm by using LP solution as initial value.) We also require that new control points have energy not greater than the previous estimation.

D. Scale Invariant Formulation

The above linear formulation is not scale invariant even though it can handle small scale changes. One solution to the problem is that we apply successive convex matching in multiple scales and use the best matching score at each instant to quantify the similarity of an action to the template. We usually need to search through only a few scales because the matching is deformable. For short video sequences, the simple method is applicable, but the increase in complexity may greatly slow down the processing of a long video. We extend the convex matching method so that it is invariant to scale and has a complexity independent of the number of discrete scale levels. In the following, we assume that objects have small rotation changes, which is true for most action videos.

The scale invariant video matching can be obtained by optimizing the following modified energy function:

$$\min_{\mathbf{f},l} \left\{ \sum_{i=1}^{n} \sum_{\mathbf{s}\in S_i} C^i(\mathbf{s}, \mathbf{f}_{\mathbf{s}}^i) + \lambda \sum_{i=1}^{n} \sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}_i} d(\mathbf{p}-\mathbf{q}, l\cdot(\mathbf{f}_{\mathbf{p}}^i - \mathbf{f}_{\mathbf{q}}^i)) + \mu \sum_{i=1}^{n-1} d(\bar{\mathbf{s}}^{(i+1)} - \bar{\mathbf{s}}^i, l\cdot(\bar{\mathbf{f}}^{(i+1)} - \bar{\mathbf{f}}^i)) \right\} \quad (2)$$

where l is a positive scaling factor. We, therefore, have to find the point correspondence and scale simultaneously. Scaling target vectors is necessary; the seemingly simple formulation that scales the template vectors is in fact wrong because it introduces a strong bias for small objects. Here we assume that the matching cost from a template point to a target point is determined by only their locations, e.g., the matching cost is scale invariant. The matching cost scale invariance does not imply that the features have to be invariant. A simple way of computing invariant cost using non-invariant features is to compute the feature distances at a sequence of discrete scales and use the minimum as the invariant cost. The above nonlinear optimization is hard to solve directly. We can approximate it with a linear formulation involving discrete scales; the linear approximation is further relaxed into linear program which can be efficiently solved.

To linearize the objective function, we introduce a new assignment variable $z_{\mathbf{s},\mathbf{t},l_k}^i$ which indicates the true or false of the matching from template point $\mathbf{s}$ to target point $\mathbf{t}$ at discrete scale l_k; if the matching is true z equals 1 and otherwise 0. We quantize the scale into K discrete values $l_1, l_2, ..., l_K$. For example, we can quantize scale from 0.5 to 2 into 7 values with 0.25 step size. Scale dependent assignment variable z and scale independent assignment variable w has the following relation assuming that w is strictly 0 or 1:

$$w_{\mathbf{s},\mathbf{t}}^i = \sum_{k=1}^{K} z_{\mathbf{s},\mathbf{t},l_k}^i.$$

Recalling that $\sum_{\mathbf{t}\in B_{\mathbf{s}}^i} w_{\mathbf{s},\mathbf{t}}^i = 1$, z is therefore guaranteed to have a single 1 for each template point. The term $l\cdot\mathbf{f}_{\mathbf{p}}^i$ can now be approximated by the linear combination

$$l\cdot\mathbf{f}_{\mathbf{p}}^i \approx \sum_{k=1}^{K} \sum_{\mathbf{t}\in B_{\mathbf{p}}^i} l_k \cdot \mathbf{t} \cdot z_{\mathbf{p},\mathbf{t},l_k}^i.$$

Based on the definition of $\bar{\mathbf{f}}_{\mathbf{p}}^i$, we also have

$$l\cdot\bar{\mathbf{f}}_{\mathbf{p}}^i \approx \frac{1}{|S_i|} \sum_{\mathbf{p}\in S_i} \sum_{k=1}^{K} \sum_{\mathbf{t}\in B_{\mathbf{p}}^i} l_k \cdot \mathbf{t} \cdot z_{\mathbf{p},\mathbf{t},l_k}^i.$$

We further need to make sure that each template point chooses the same scale in matching by introducing the following constraint:

$$\sum_{k=1}^{K} \sum_{\mathbf{t}\in B_{\mathbf{s}}^i} l_k \cdot z_{\mathbf{s},\mathbf{t},l_k}^i = l \quad \forall \mathbf{s}, i.$$

If $d(.)$ uses L_1 norm, we can linearize the second and third terms in (2) using the auxiliary variable trick. Including other linear terms in (1), we have an integer linear program which is equivalent to the original optimization in (2) except the approximation introduced by the scale quantization. We can further relax the integer program into a linear program by dropping the binary constraints for both w and z.

The relaxed linear program has similar properties to the non-scale-invariant version and can be further simplified using the lower convex hull property. It is not hard to verify that only the variables of w and z corresponding to the lower convex hull vertices of each matching cost surface need to be included; the redundant variables can be removed without changing the linear program solution. The scale also has a similar lower convex hull property. For this simple formulation in which the matching cost is invariant to scale changes, we only keep the variables of z that correspond to the boundary scales (the maximum and minimum scales). After relaxing z into floating point number, we simulate continuous scale changes in a specific interval.

The relaxed linear program needs to be refined to match the target accurately. The successive approximation method can still be applied. Trust region shrinkage can be applied to both the template points and the scale. In our implementation, we only shrink trust regions of the template points; the scale is always optimized in the largest range.

The average complexity of a linear program is roughly proportional to the number of constraints and logarithm of the number of variables [16]. The number of constraints of the proposed linear program has nothing to do with the number of target feature points. Using the lower convex hull trick, the number of variables is also largely independent of the number of target points. As mentioned before, the complexity of the linear program is constant to the number of discrete scales. The refinement procedure usually takes constant number of iterations. The average complexity of the proposed linear program is therefore a low order polynomial of the number of templates points and largely independent of the number of target candidates. This makes the proposed method efficient in searching through large range with huge number of target feature points.

The proposed method is not explicitly invariant to temporal scale changes. Fortunately, most actions have similar temporal scales constrained by the physics of movement. For videos in normal temporal sampling rate, the temporal scale changes are usually small. The temporal scale changes result in mis-alignment of the template images with the target video frames. Because of the movement continuity, the mis-alignment is reflected as the distortion of the shapes of objects at specific instants. Since the proposed method uses deformable template matching, such distortion does not cause problems as shown in our experiments.

E. Action Detection

By using the proposed scheme, we register the template pose sequence to the targets in videos. Because of the center continuity constraint, the matching finds objects and poses consistently in the spatial locations through time. As will be shown in the experiments, the location consistency constraint is important for finding actions in cluttered videos. After the registration process, we locate an object that has potential to perform a specific action at an instant. We need further to verify whether the match is a real target. In verification, we compare the matching targets with templates to decide how similar these two constellations of matched points are and whether the matching result corresponds to the same action. We use the following quantities to measure the difference between the template and the matching object: the first measure is D, defined as the average of pairwise length changes from the template to the target. To compensate for the global deformation, a global affine transform $\mathcal{A}$ is first estimated based on the matching and then applied to the template points before calculating D. Further, measure D is normalized with respect to the average edge length of the template. The second measure is the average template matching cost M using the log-polar feature. The total matching cost is simply defined as $M + \alpha D$, where α has a typical value of ten if image pixels are in the range of 0–255. Experiments show that only about 100 randomly selected feature points are needed in calculating D and M.

III. Experiment Results

A. Matching Random Sequences

In the first experiment, we test the proposed scheme with synthetic images. In each testing, three 128×128 coupled random template images are generated. Each template image contains 50 random points. The 256×256 target images contain a randomly shifted and perturbed version of the data points. The perturbation is uniformly disturbed in two settings: 0–5 and 0–10. The centers of the target are also randomly perturbed in the range 0–5. We use the log-polar feature in all our experiments. We compare our result with a greedy searching scheme. Other standard schemes, such as BP, cannot be easily extended to solve the minimization problem in this paper. Instead we use BP to match each image pair separately as a benchmark in comparison. Each experiment is repeated in a deformation and clutter setting over 100 trials. Fig. 7 shows the average matching error distribution in different assumed-error regions. A good performance should have high values on the left and low values on the right. When both the noise level and distortion level are low, the greedy scheme has comparable performance. Since there is one single target

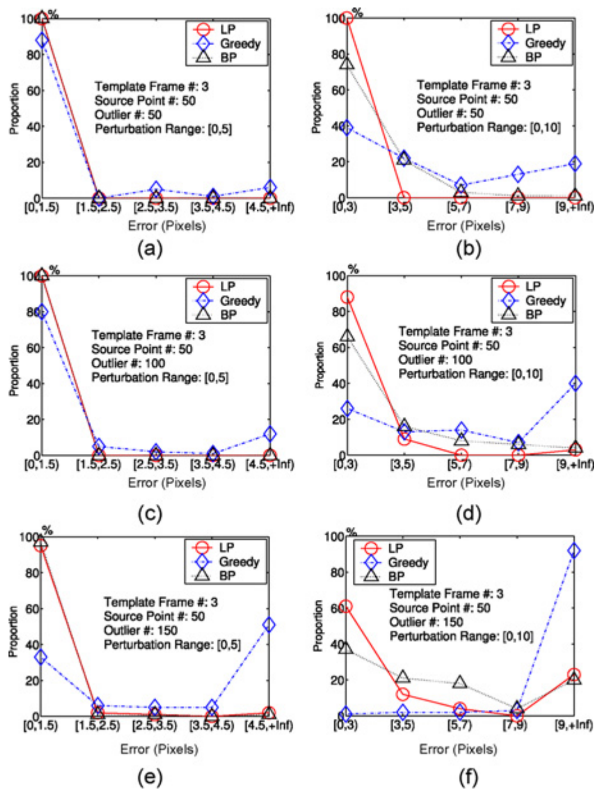


Fig. 7. Average matching error distribution for random sequence test I. Three-frame random sequences are used in testing. In each template frame, there are 50 template points. In the target images, their locations are shifted and randomly perturbed to simulate deformation ranging from 5 to 10. 50, 100 and 150 clutter points are added into the target images. The normalized error histograms for the proposed method (LP), the greedy search method (Greedy) and belief propagation method (BP) are illustrated.

in each image, BP has similar performance as the proposed scheme for experiment settings with low deformation. The performance of greedy schemes degrades rapidly when the levels of noise and distortion increase. In these cases, the proposed scheme greatly outperforms the greedy scheme. It is also better than baseline BP when there is large distortion. Fig. 8 shows the comparison results of matching random sequences in a different outlier setting which introduces an extra duplicated and perturbed object into the second target frame. For BP and greedy method, matching error for the second template frame is the smaller one of matching either of the two objects in the target frame. In this test, the proposed sequence matching scheme yields much better result. A center continuity constraint is necessary for correct matching in cutter.

B. Matching Actions in Video

Fig. 9 shows the results of matching for real images with the proposed scheme, the greedy scheme and the BP matching for single image pairs. The proposed scheme still works well in cluttered images, while the greedy scheme and BP fail to locate the target. BP is also about 100 times slower.

We further conducted experiments to search for a specific action in video. In these test videos, a specific action only appears a few times. The template sequence is swept along

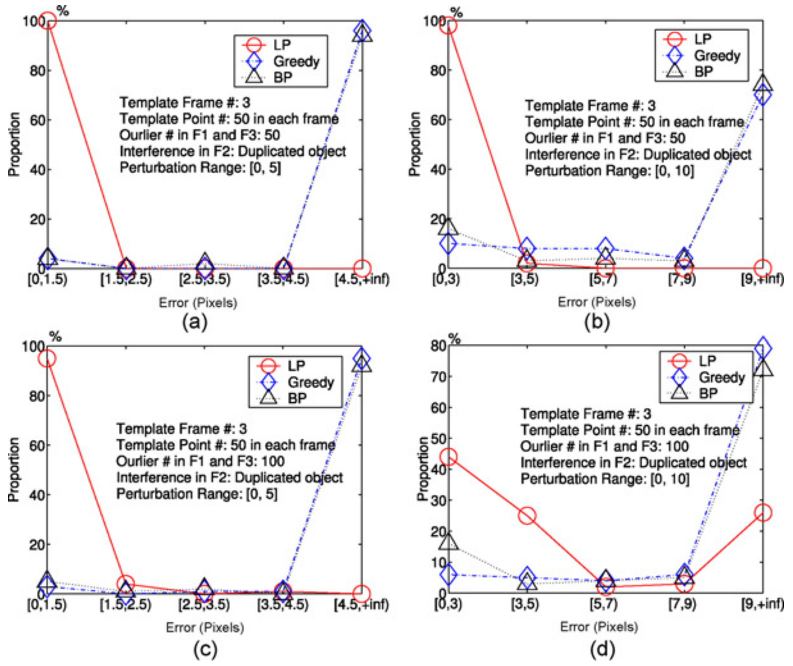


Fig. 8. Average matching error distribution for random sequence test II. Apart from similar settings to random sequence test I, we add another duplicated object in the second target frame to simulate multiple objects. The normalized error histograms for LP, the greedy search and belief propagation are illustrated.

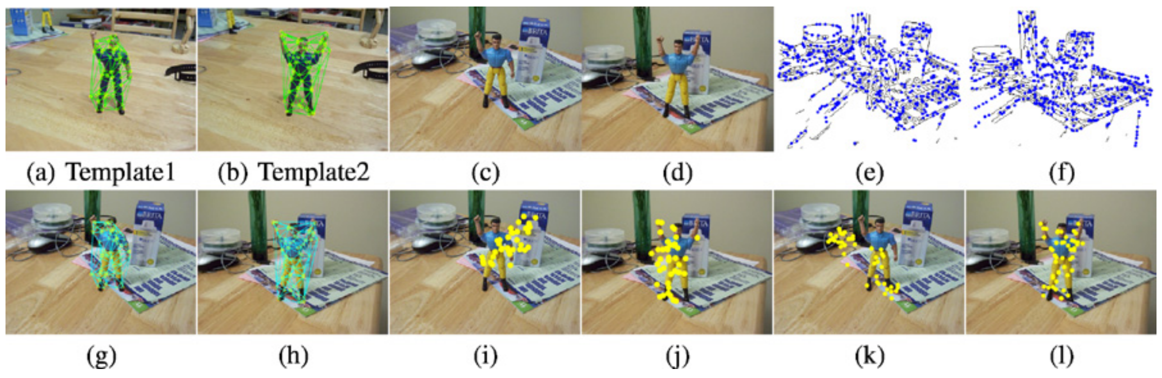


Fig. 9. Matching flexible objects. (a) and (b) Templates. (c) and (d) Target images. (e) and (f) Edge map and target feature points. (g) and (h) Matching with the proposed scheme. (i) and (j) Matching with the greedy scheme. (k) and (l) Matching with BP for each image pair.

the time axis with a step of one frame, and for each instant we match video frames with the templates.

We first applied the matching scheme to detect actions in a 1000-frame fitness sequence. As shown in Fig. 10, two actions are correctly detected at the top of each short list. Fig. 11 illustrates how the number of key frames affects the performance of the action detector. As illustrated, the detection result using one key frame is worse than the result in Fig. 10 where we use two key frames; the detection result using three key frames does not differ much from the result using two key frames. For simple actions, a small number of key frames, e.g., two or three, are found sufficient. More key frames would not improve the performance substantially. The proposed method has a complexity largely decoupled from the number of target candidates and therefore the matching time at each instant is almost constant for different type of videos. With 3 key frames and about 100 feature points in each template frame, the running time for matching the template at each instant in the target video is about 1 s using a 2.8 GHz machine. We further test the proposed method on detecting specific sign language gestures. Sign language is challenging because of the very subtle differences. Fig. 13 shows a search result for the gesture “go” in a 500-frame video. Fig. 13(a) shows the two key poses used in searching. The templates are generated from a different subject. Fig. 13(b)–(g) show the matched starting postures and ending postures ranked with their matching costs. The proposed scheme locates both of the two appearances of the gesture in the video in the top two ranks. Fig. 14 shows another searching result for the gesture “work” in a 1000-frame video. The two gestures in the video are successfully located in the top two rank positions. Fig. 15 shows a searching result for the gesture “year” in a 1000-frame video. Five appearances of the gesture are located in top six of the short list. One false detection is inserted

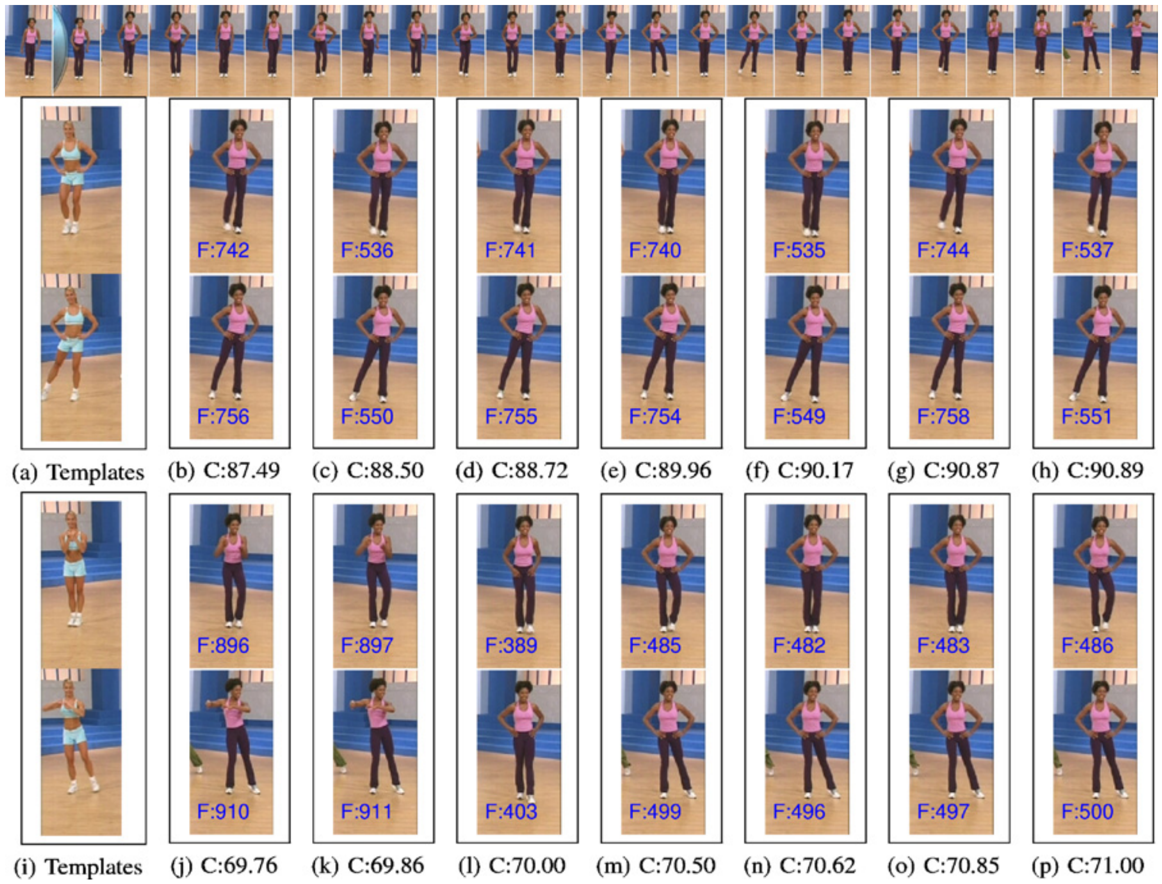


Fig. 10. Finding actions in a 1000-frame fitness sequence. (a) and (i) Template images for 2 actions. (b)–(h) and (j)–(p) Short lists ranked by the matching costs. The 2 right-leg-out actions and 1 right-arm-out-left-leg-out action are ranked at the top of each short list.

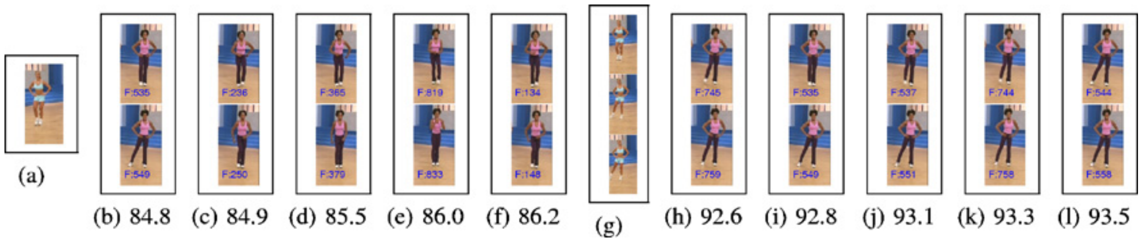


Fig. 11. Action detection using different number of key frames. (a) and (g) 1-frame and 3-frame templates. (b)–(f) and (h)–(l) Action short lists using the 1-frame and 3-frame templates respectively.

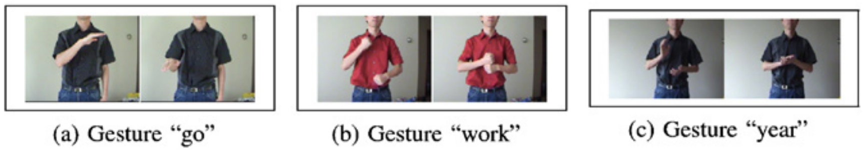


Fig. 12. Template images in gesture detection. (a)–(c) Starting and ending template frames for Figs. 13–15.

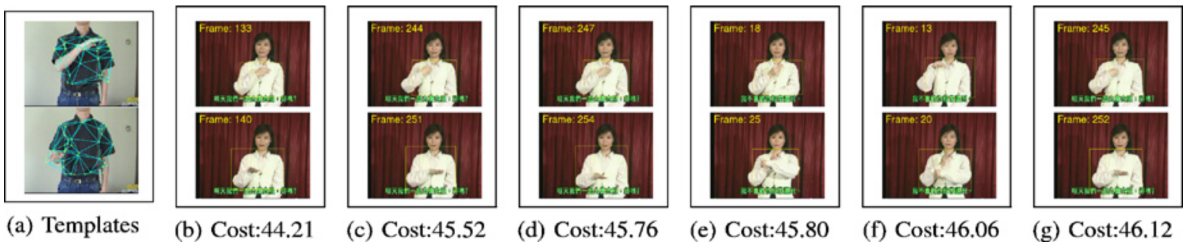


Fig. 13. Searching gesture "go" in a 500-frame sign language sequence. (a) Templates. (b)–(g) Top 6 matches.

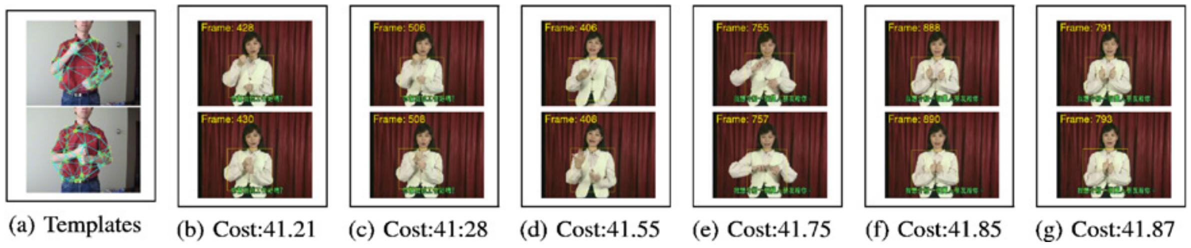


Fig. 14. Searching gesture "work" in a 1000-frame sign language sequence. (a) Templates. (b)–(g) Top 6 matches.

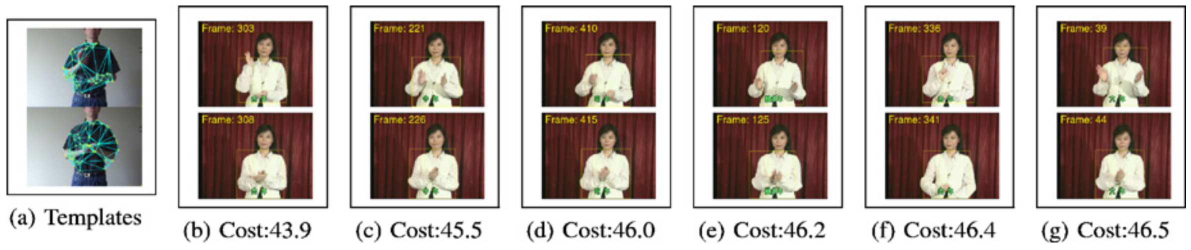


Fig. 15. Searching gesture "year" in a 1000-frame sign language sequence. (a) Templates. (b)–(g) Top 6 matches.

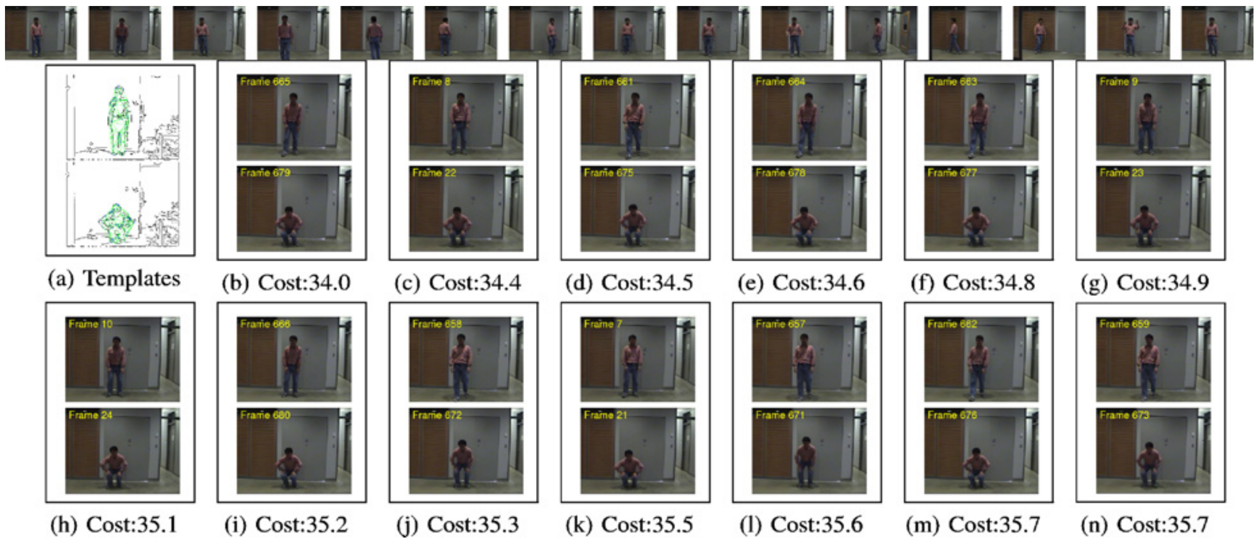


Fig. 16. Searching "kneeling" in a 800-frame indoor sequence. (a) Templates. (b)–(n) Top 13 matches.

at rank 5. Figs. 16 and 17 show experiments to locate two actions, kneeling and hand-waving, in indoor video sequences of 800 and 500 frames respectively. The two-frame templates are from videos of another subject in different environments. The videos are taken indoors and contain many bar structures which are very similar to human limbs. The proposed scheme finds all the two kneeling actions in the test video in the top two of the short list; and all the waving hand actions in the top 11 ranks. Fig. 18 shows the result of search for a "throwing" action in a 2500-frame baseball sequence. Closely interlaced matching results are merged and our method finds all four appearances of the action at the top of the list.

Fig. 19 shows the result of detecting ballet actions in cluttered background. The sequence involves large camera motion. Two key frames are used as templates. We show the first frames of the actions detected in a short list. The short list has been shortened by applying non-minimum suppression to the matching costs. Referencing the matching cost curve in Fig. 19, 6 action segments out of 8 are detected in the top 12 of the short list with 2 false detections. The proposed matching method performs well.

We applied Chamfer matching to the same ballet video sequence using the same template. The Chamfer matching result is shown in Fig. 20. Chamfer matching gives much worse result than the proposed method. Since Chamfer matching uses a rigid template, it has difficulty in distinguishing between smooth object shape deformation and shape change caused by a distinctive action. It also tends to match templates to cluttered backgrounds instead of true objects.

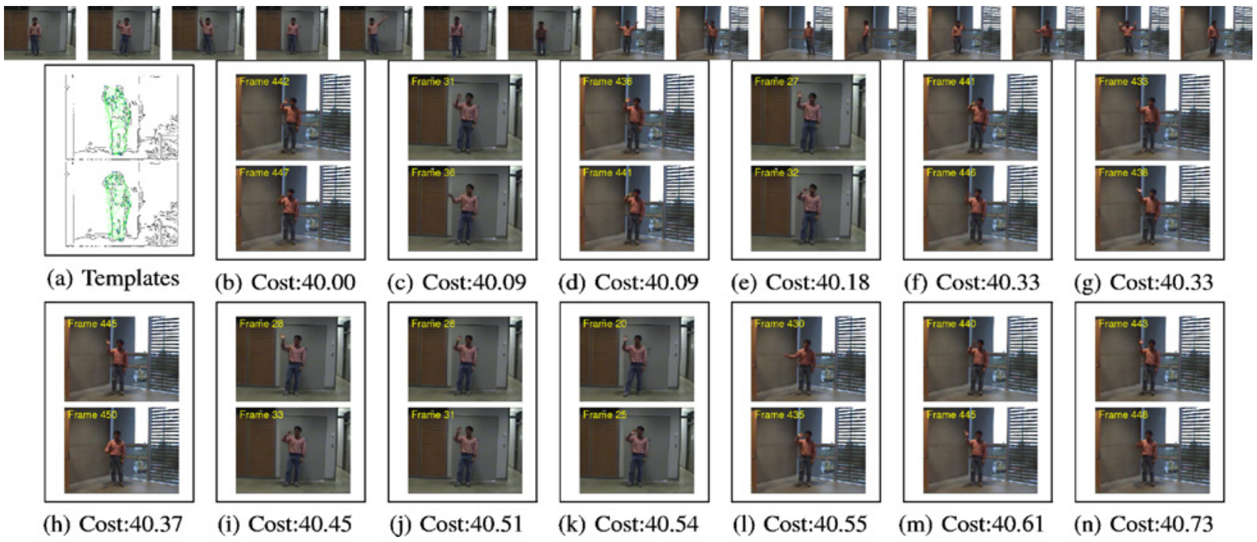


Fig. 17. Searching “right hand waving” in a 500-frame indoor sequence. (a) Templates. (b)–(n) Top 13 matches.

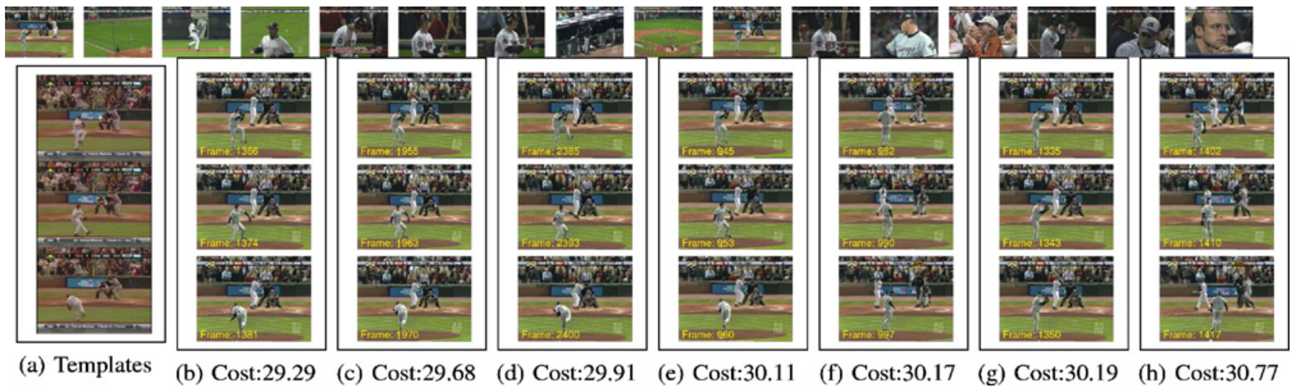


Fig. 18. Searching “throwing ball” in a 2500-frame baseball sequence. (a) Templates. (b)–(g) Top 6 matches.

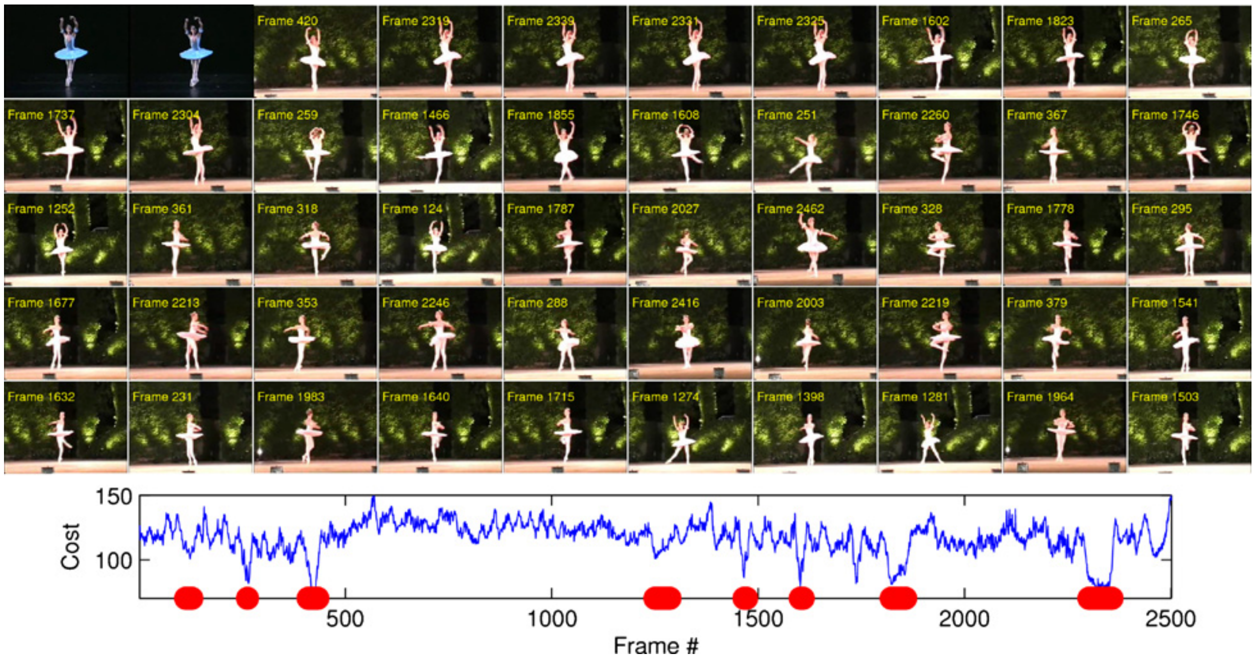


Fig. 19. Finding actions in a 2500-frame ballet sequence. The sequence involves complex actions, moving camera and cluttered background. The first two images in the list are first and last frames of the template; the following images form the short list ranked by the matching costs (non-minimum suppression is applied to remove some close detections). The matching cost curve is shown below the images, with the red dots indicating the true action locations.

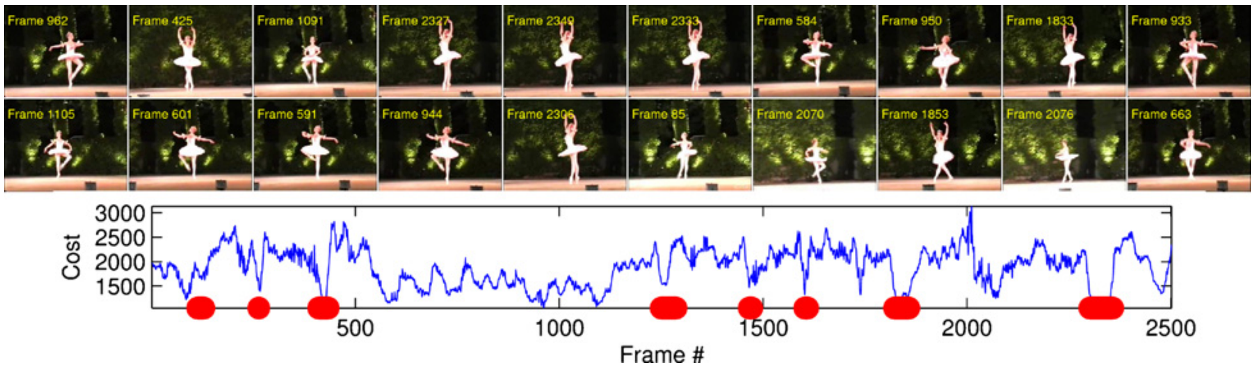


Fig. 20. Chamfer matching result using the same template as Fig. 19. Chamfer matching performs poorly due to the clutter background and object deformation.

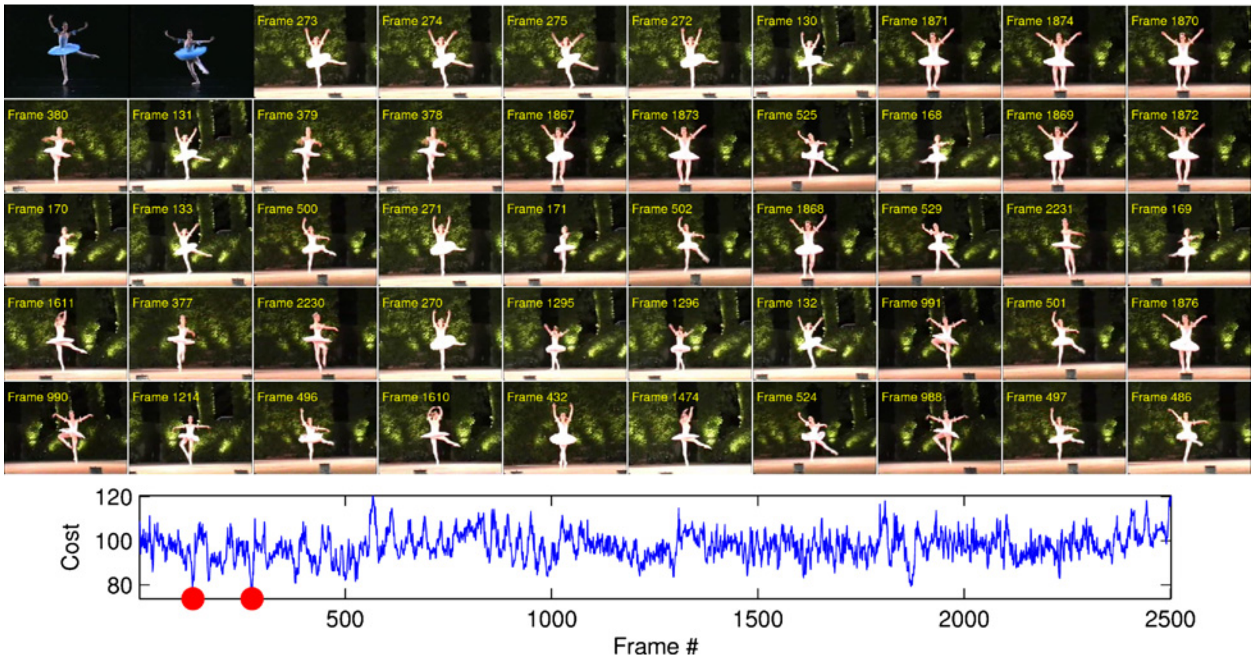


Fig. 21. Finding another action in the 2500-frame ballet sequence. The proposed method accurately locates the action at the top of the short list.

Fig. 21 shows the action detection result for a different ballet action. The proposed method accurately detects the two instances of the action in the ballet video. Note that the objects at the two action instants have different scales. Since the proposed method is scale invariant, it successfully detects both of the instances. Fig. 22 shows action detection in very cluttered background. The proposed method successfully found the two segments of actions at the top of the short list. We found that false detection in our experiments is mainly due to similar structures in the background near the subject. Very strong clutter is another factor that may cause the matching scheme to fail. Prefiltering or segmentation operations to partially remove the background clutter can further increase the robustness of detection.

We further test the proposed action detection method with the KTH dataset [18] which includes six action classes. We select templates with two key poses from the first video clip in each category. Fig. 23 shows examples of the two-key-frame templates in the first row of each sub-figure. We select regions in the two-frame templates for each of the six actions. Graph templates are automatically generated using randomly selected edge points in the region of interest. The templates are then used to compare with each testing video clip at each instant using the proposed matching scheme and the minimal matching cost is used as the matching cost for a video clip. Fig. 23 illustrates some matching examples. Fig. 24 shows the performance of action detection. The equal precision and recall point is from 65% to 80%. The detected template clip is removed from the short lists when computing the recall-precision curves. As shown in Fig. 24(a)–(f), confusion happens for similar actions. Clapping is quite similar to the arm-down action in hand waving. Boxing tends to confuse with hand waving and clapping because it matches half body movement in these actions. The action detection method performs quite well considering the similarity of these actions, a large variety of actors, and that we use a single template for each action class. Comparing with Chamfer matching whose result on the same dataset is shown in Fig. 25, the proposed method performs much better.

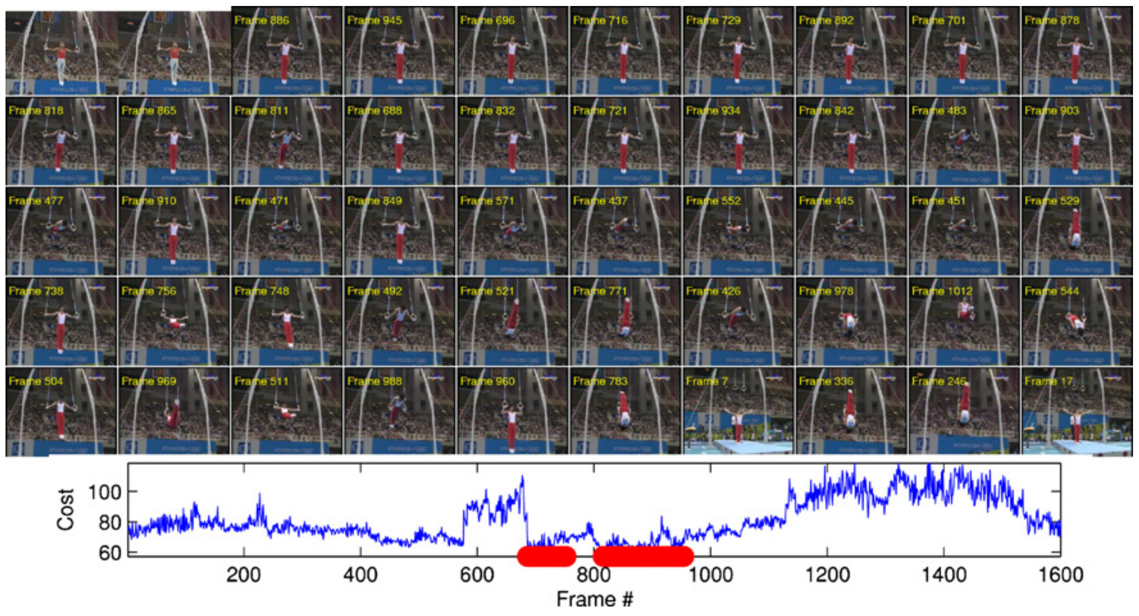


Fig. 22. Search an action in a 1600-frame gymnastic video. The first two images show the templates of the action. The non-minimum suppressed short list is shown for the located actions. The proposed method successful locates two action segments in the video.

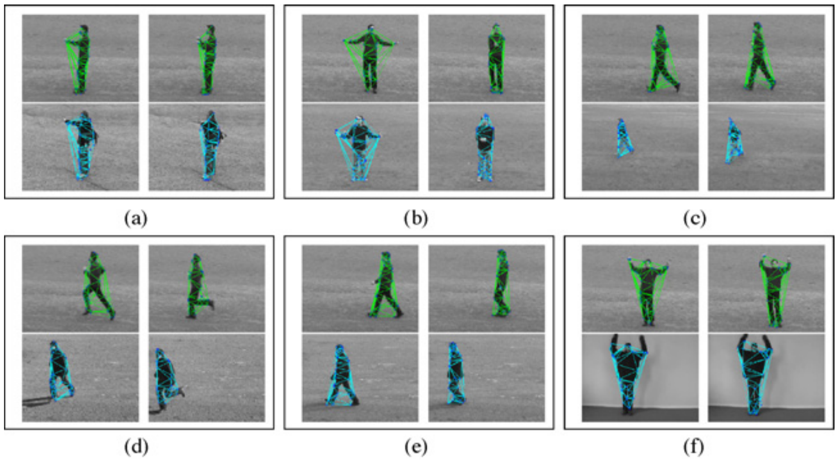


Fig. 23. Matching examples. In (a), (b), (c), (d), (e), and (f) top rows are templates and bottom rows are matching results.

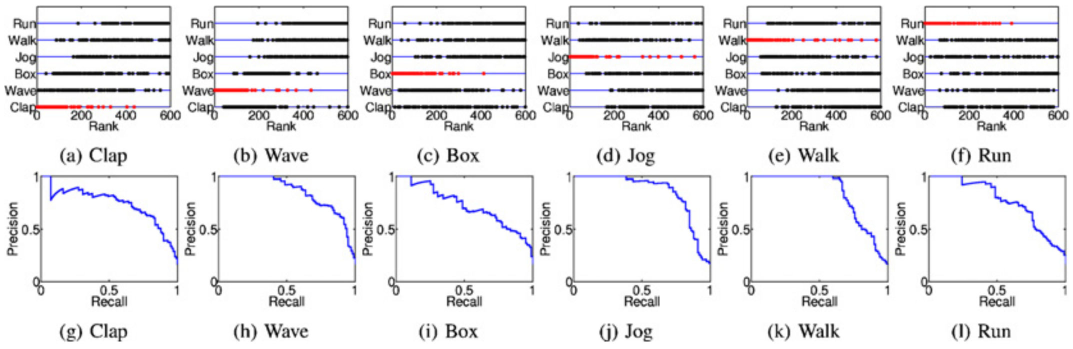


Fig. 24. Action detection using the proposed method on the KTH dataset. There are six action classes (clapping, waving, boxing, jogging, and running) and 599 video clips. (a)–(f) The rank of videos for each action detection test. Red dots indicate the videos that contain the target action. (g)–(l) Recall-precision curves.

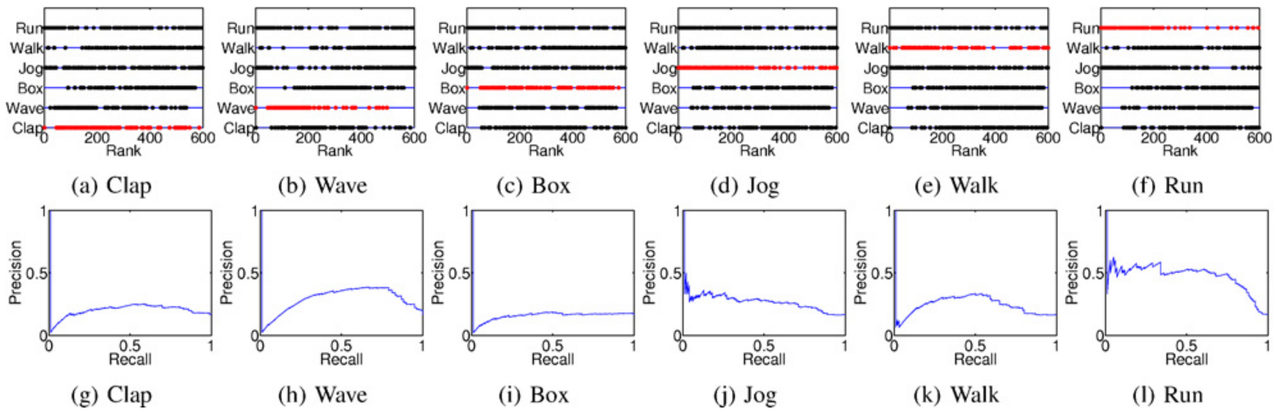


Fig. 25. Action detection using Chamfer matching on the KTH dataset. (The template clips are not removed from the short lists.)

	clap	wave	box	jog	walk	run
clap	53	33	7	4	1	0
wave	14	76	6	3	0	0
box	2	2	69	19	7	0
jog	0	0	3	84	4	8
walk	0	0	4	28	67	0
run	0	0	2	30	9	58

Fig. 26. Confusion matrix for each action class. We randomly select 1 video clip from each class as a template and use the other 593 video clips for testing.

Methods	**Training method**	**Accuracy %**
Our method	1 sample per class	68.61
Fathi et al. [34]	Splits	90.50
Jhuang et al. [35]	Splits	91.70
Niebles et al. [33]	Leave one out	81.50
Schuldt et al. [18]	Splits	71.72
Ke et al. [36]	Splits	62.96

Fig. 27. Comparison with other reported results on the KTH dataset.

We can also construct an action classifier using the six action templates. We categorize a video clip into an action class if it matches the action exemplar with the smallest cost. The confusion matrix of the proposed method is shown in Fig. 26. The average classification accuracy is 68.61%. In this experiment, we use only one training sample from each action class and test on all the other videos in the dataset. The comparison with other methods is shown in Fig. 27.

IV. Conclusion

We have proposed a successive convex programming scheme to match video sequences using intra-frame and inter-frame constrained local features. By convexifying the optimization problem sequentially with an efficient linear programming scheme which can be globally optimized in each step, and gradually shrinking the trust region, the proposed method is much more robust than greedy searching schemes. Different from other robust matching methods, successive convex matching has unique features: it involves a very small number of basis points and thus can be applied to large-scale problems that involve a large number of target points. We included experiments demonstrating the success of the proposed scheme in detecting specific actions in video sequences. Because the template deforms, this scheme can deal with large distortions between the template and the target object. In future work, we will investigate efficient implementations so that the proposed method can be used in real-time applications.

References

[1] *Kidsroom: An Interactive Narrative Playspace*, Cambridge, MA [Online]. Available: http://vismod.media.mit.edu/vismod/demos/kidsroom/ kidsroom.html
[2] K. M. G. Cheung, S. Baker, and T. Kanade, "Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, vol. 1. 2003, pp. I77–I84.
[3] L. Emering and B. Herbelin, "Body gesture recognition and action response," *Handbook of Virtual Humans*. Chichester, U.K.: Wiley, 2004, pp. 287–302.
[4] J. Besag, "On the statistical analysis of dirty pictures," *J. R. Stat. Soc. Lond.*, ser. B, vol. 48, pp. 259–302, 1986.
[5] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001.
[6] Y. Weiss and W. T. Freeman, "On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 723–735, Feb. 2001.
[7] A. C. Berg, T. L. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondences," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, vol. 1. 2005, pp. 26–33.
[8] S. Carlsson and J. Sullivan, "Action recognition by shape matching to key frames," in *Proc. IEEE Comput. Soc. Workshop Models Versus Exemplars Comput. Vision*, vol. 23. no. 3, Mar. 2001, pp. 257–267.
[9] G. Mori, X. Ren, A. Efros, and J. Malik, "Recovering human body configurations: Combining segmentation and recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, vol. 2. Jul. 2004, pp. 326–333.
[10] G. Mori and J. Malik, "Estimating human body configurations using shape context matching," in *Proc. Eur. Conf. Comput. Vision*, vol. 2352. 2002, pp. 666–680.
[11] A. A. Efros, A. C. Berg, G. Mori, and J. Malik, "Recognizing action at a distance," in *Proc. Int. Conf. Comput. Vision*, vol. 2. 2003, pp. 726–733.
[12] E. Shechtman and M. Irani, "Space-time behavior based correlation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, vol. 1. Jun. 2005, pp. 405–412.

[13] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient matching of pictorial structures," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, vol. 2. 2000, pp. 66–73.

[14] R. Ronfard, C. Schmid, and B. Triggs, "Learning to parse pictures of people," in *Proc. Eur. Conf. Comput. Vision*, vol. 2353. 2002, pp. 700–714.

[15] D. Ramanan, D. A. Forsyth, and A. Zisserman, "Strike a pose: Tracking people by finding stylized poses," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, vol. 1. Jun. 2005, pp. 271–278.

[16] V. Chvátal, *Linear Programming*. New York: Freeman, 1983.

[17] H. Jiang, M. S. Drew, and Z. N. Li, "Successive convex matching for action detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, vol. 2. Jun. 2006, pp. 1646–1653.

[18] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," in *Proc. Int. Conf. Pattern Recognit.*, vol. 3. 2004, pp. 32–36.

[19] I. Laptev and T. Lindeberg, "Space-time interest points," in *Proc. Int. Conf. Computer Vision*, 2003, pp. 432–439.

[20] I. Laptev, P. Prez, "Retrieving actions in movies," in *Proc. Int. Conf. Computer Vision*, 2007, pp. 1–8.

[21] V. Parameswaran and R. Chellappa, "Human action-recognition using mutual invariants," *Comput. Vision Image Understanding*, vol. 98, no. 2, pp. 295–325, 2005.

[22] P. Scovanner, S. Ali, and M. Shah, "A 3-D SIFT descriptor and its application to action recognition," in *Proc. ACM Multimedia*, 2007, pp. 357–360.

[23] Y. Sheikh and M. Shah, "Exploring the space of an action for human action recognition," in *Proc. Int. Conf. Comput. Vision*, vol. 1. 2005, pp. 144–149.

[24] D. Weinland, R. Ronfard, and E. Boyer, "Free viewpoint action recognition using motion history volumes," *Comput. Vision Image Understanding*, vol. 104. no. 2, pp. 249–257, Nov./Dec. 2006.

[25] A. Yilmaz and M. Shah, "Actions as objects: A novel action representation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Jun. 2005, pp. 984–989.

[26] A. Yilmaz and M. Shah, "Recognizing human actions in videos acquired by uncalibrated moving cameras," in *Proc. IEEE Int. Conf. Comput. Vision*, vol. 1. 2005, pp. 150–157.

[27] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in *Proc. Int. Conf. Comput. Vision*, 2005, pp. 1395–1402.

[28] H. Jiang, M. S. Drew, and Z. N. Li, "Matching by linear programming and successive convexification," in *Proc. IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29. no. 6, Jun. 2007, pp. 959–975.

[29] A. Rosenfeld and J. Pfaltz, "Sequential operations in digital picture processing," *J. ACM*, vol. 13, no. 4, pp. 471–494, 1966.

[30] Y. Ke, R. Sukthankar, and M. Hebert, "Event detection in crowded videos," in *Proc. Int. Conf. Comput. Vision*, 2007, pp. 1–8.

[31] H. Jiang and D. R. Martin, "Finding actions using shape flows," in *Proc. Eur. Conf. Computer Vision*, 2008, pp. 278-292.

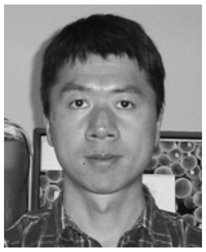
[32] Y. Wang, H. Jiang, M. S. Drew, Z. N. Li, and G. Mori, "Unsupervised discovery of action classes," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, vol. 2. 2006, Jun. pp. 1654–1661.

[33] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *Int. J. Comput. Vision*, vol. 79, no. 3, pp. 299–318, Sep. 2008.

[34] A. Fathi and G. Mori, "Action recognition by learning mid-level motion features," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, Jun. 2008, pp. 1–8.

[35] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in *Proc. Int. Conf. Comput. Vision*, 2007, pp. 1–8.

[36] Y. Ke, R. Sukthankar, and M. Hebert, "Efficient visual event detection using volumetric features," in *Proc. Int. Conf. Comput. Vision*, vol. 1. Oct. 2005, pp. 166–173.



Hao Jiang received the B.Sc. and M.Sc. degrees in electronic engineering from Harbin Engineering University, Harbin, China, in 1993 and 1995, respectively and the Ph.D. degree in computer science from Simon Fraser University, Vancouver, British Columbia, Canada, in 2006.

He is currently an Assistant Professor at the Department of Computer Science, Boston College, Chestnut Hill, MA. He was a Postdoctoral Research Fellow at the University of British Columbia, Vancouver, BC, Canada, in 2006, and an Associate Researcher at Microsoft Research Asia, Beijing, China, in 1999. His research interests include computer vision, image processing, multimedia, and graphics.

Mark S. Drew received the M.A.Sc. degree in mathematics, M.Sc. degree in physics from the University of Toronto, Toronto, Ontario, Canada, in 1970 and 1971, respectively and the Ph.D. degree in physics from the University of British Columbia, British Columbia, Canada, in 1976.

He is currently a Professor in the School of Computing Science, Simon Fraser University, Vancouver, British Columbia, Canada. His research interests include fields of multimedia, computer vision, image processing, color, photorealistic computer graphics, and visualization. He is the author of over 150 refereed papers in journals and conference proceedings. He is the holder of a U.S. Patent in digital color processing, a U.S. patent application in color computer vision, and two U.K. and international patent applications in digital image processing.

Ze-Nian Li received the B.Sc. degree in electrical engineering from the University of Science and Technology, Hefei, China, and the M.Sc. and Ph.D. degrees in computer science from the University of Wisconsin, Madison.

He is a Professor at the School of Computing Science, Simon Fraser University, Vancouver, British Columbia, Canada. Previously, he was an Electronic Engineer in charge of design of digital and analogical systems. He had been the Director of the School of Computing Science from 2001 to 2004. His current research interests include computer vision, pattern recognition, multimedia, image processing, and artificial intelligence. He is the author of over 100 referred papers in journals and conference proceedings. He is the coauthor of the book *Fundamentals of Multimedia* (Englewood Cliffs, NJ: Prentice Hall, 2004).