Computer Vision and Image Understanding 114 (2010) 901–914

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Online visual tracking with histograms and articulating blocks

S.M. Shahed Nejhum [a], Jeffrey Ho [a], Ming-Hsuan Yang [b,*]

[a] Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32611, United States
[b] Electrical Engineering and Computer Science, University of California, Merced, CA 95344, United States

ARTICLE INFO

Article history:
Received 18 June 2009
Accepted 16 April 2010
Available online 4 May 2010

Keywords:
Object tracking
Image segmentation
Object detection

ABSTRACT

We propose an algorithm for accurate tracking of articulated objects using online update of appearance and shape. The challenge here is to model foreground appearance with histograms in a way that is both efficient and accurate. In this algorithm, the constantly changing foreground shape is modeled as a small number of rectangular blocks, whose positions within the tracking window are adaptively determined. Under the general assumption of stationary foreground appearance, we show that robust object tracking is possible by adaptively adjusting the locations of these blocks. Implemented in MATLAB without substantial optimization, our tracker runs already at 3.7 frames per second on a 3 GHz machine. Experimental results have demonstrated that the algorithm is able to efficiently track articulated objects undergoing large variation in appearance and shape.

Published by Elsevier Inc.

1. Introduction

Developing an accurate, efficient and robust visual tracker is always challenging, and the task becomes even more difficult when the target is expected to undergo significant and rapid variation in shape as well as appearance. While the audience is delighted and awed by the virtuoso performance of the world-renown skaters (Fig. 1), their graceful movements and dazzling poses offer multiple challenges for any visual tracker. In this example (and many others), the appearance variation is mainly due to change in shape while the foreground intensity distribution remains roughly stationary. An important problem is then to efficiently exploit this weak appearance constancy assumption for accurate visual tracking amidst substantial shape variation.

Intensity histogram is perhaps the simplest way to represent object appearance, and tracking algorithms based on this idea abound in the literature (e.g., [1,2]). For rectangular shapes, efficient algorithms such as integral image [3] and integral histogram [4] have been successfully applied to object detection and tracking [5]. In particular, it is possible to rapidly scan the entire image to locate the target. However, computing intensity histogram from a region bounded by some irregular shape cannot be done efficiently and rapidly using these methods. To deal with shape variation in the context of histogram based tracking, one general idea is to use a (circular or elliptical) kernel [6,7] to define a region around the target from which a weighted histogram can be computed. Rapid scanning of the image using this approach is not possible; instead, differential algorithms can be designed to iteratively converge to the target object [2]. Nevertheless, differential approaches become problematic for tracking sequences with rapid and large motions. In a way, the kernel imposes a "regularity" constraint on the irregular shape, thereby relaxing the more difficult problem of efficiently computing the intensity histogram from an irregular shape to that of a simpler one of estimating histogram from a regular shape.

Another way to deal with irregular shapes is to enclose the target with a regular shape (e.g., a rectangular window) and compute histogram from the enclosed region. However, this inevitably includes background pixels when the foreground shape cannot be closely approximated. Consequently, the resulting histogram can be corrupted by background pixels, and the tracking result degrades accordingly (e.g., unstable or jittered results as shown in Fig. 1). Furthermore, complete lack of spatial information in histograms is also undesirable. For problems such as face tracking that do not have significant shape variation, it is adequate to use intensity histogram as the main tracking feature [1]. However, for a target undergoing significant shape variation, the spatial component of the appearance is very prominent, and the plain intensity histogram becomes inadequate as it alone often yields unstable tracking results.

Each of the aforementioned problems has been addressed to some extent (e.g., spatiogram [8] for encoding spatial information in histogram). However, most of them require substantial increase of computation time, thereby making these algorithms applicable only to local search and infeasible for global scans of images. Consequently, such algorithms are not able to track objects undergoing rapid motions.

* Corresponding author.
E-mail addresses: smshahed@cise.ufl.edu (S.M.S. Nejhum), jho@cise.ufl.edu (J. Ho), mhyang@ucmerced.edu (M.-H. Yang).

1077-3142/$ - see front matter Published by Elsevier Inc.
doi:10.1016/j.cviu.2010.04.002

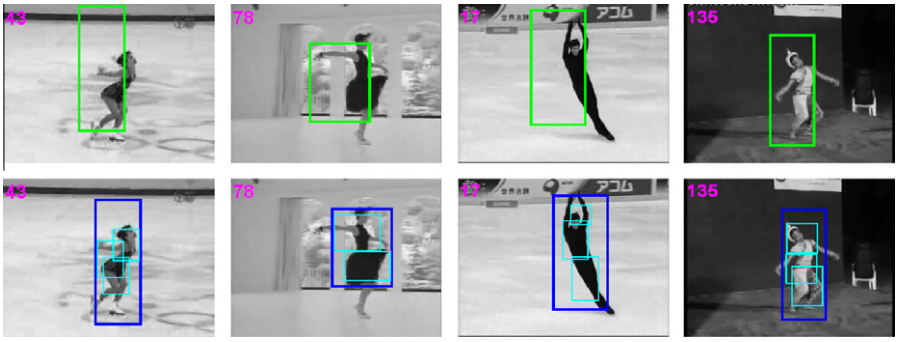


Fig. 1. Top: Using only histogram for representing object's appearance, the tracking results are often unsatisfactory. Bottom: Using the proposed algorithm, the tracking results are much more consistent and satisfactory.

In this paper, we propose a tracking algorithm[1] that solves the above problems, and at the same time, it still has comparable running time as the tracking algorithm using (plain) integral histogram [4]. The proposed algorithm consists of global scanning, scaling, local refinement and update steps. The main idea is to exploit an efficient appearance representation using histograms that can be easily evaluated and compared so that the target object can be located by scanning the entire image. Shape update, which typically requires more elaborated algorithms, is carried out by adjusting a few small blocks within tracked window. Specifically, we approximate the irregular shape with a small number of blocks that cover the foreground object with minimal overlaps. As the tracking window is typically small, we can estimate the foreground region using a fast segmentation algorithm without increasing the run-time complexity significantly. We then update the target shape by adjusting these blocks locally so that they provide a maximal coverage of the foreground target.

The adaptive structure in our algorithm contains the block configuration and their associated weights. Shape of the target object is loosely represented by block configuration, while its appearance is represented by intensity distributions and weights of these blocks. In doing so, spatial component of the object's appearance is also loosely encoded in block structure. Furthermore, these rectangular blocks allow rapid evaluations and comparisons of histograms. Note that our goal is not to represent both shape and appearance precisely since this will most likely require substantial increase in computation. Instead, we strive for a simple but adequate representation that can be efficiently computed and managed. Compared with tracking methods based on integral histograms, our tracker is also able to efficiently scan the entire image to locate the target, which amounts to the bulk of the processing time for these algorithms. The extra increase in running time of our algorithm results from the refinement and update steps. Since segmentation is carried out only locally in a (relatively) small window and the weights can be computed very efficiently, such computation overhead is generally small. Experimental results reported below demonstrate that our algorithm renders much more accurate and stable tracking results compared to the integral histogram-based tracker, with a negligible increase in running time.

2. Previous work

There is a rich literature on shape and appearance modeling for visual tracking, and a comprehensive review is of course beyond the scope of this paper [10]. In this section, we discuss the most relevant works within the context of single articulated object tracking. Specifically, we aim to track generic articulated objects from images acquired with one camera at a distance while undergoing large and rapid deformation in shape as well as appearance. We note that there exist tracking algorithms for specific objects operating under different imaging conditions and constraints, e.g., human tracking [11–14], hand tracking [15–17,6], model-based tracking [18,19], to name a few.

Articulated objects can be modeled with parameterized shapes or contours. Active contours using parametric models [17,20] typically require offline training, and expressiveness of these models (e.g., splines) is somewhat restrictive. Furthermore, with all the offline training, it is still difficult to predict the tracker's behavior when hitherto unseen target is encountered. For example, a number of exemplars have to be learned from training data prior to tracking in [21], and the tracker does not provide any mechanism to handle shapes that are drastically different from the templates. Likewise, there is also an offline learning process involved in the active shape and appearance models [22]. Level set algorithms have also been successfully applied to track articulated objects [23–26]. However, these methods rely mainly on the information near the contours and do not exploit the rich appearance or texture information. In addition, these algorithms usually do not have mechanisms to handle drifting effects.

Instead of using contours to model shapes, kernel-based methods represent target's appearance with intensity, gradients, and color statistics [1,2,27]. These methods have demonstrated successes in tracking targets whose shapes can be well enclosed by ellipses. Although methods using multiple kernels [28,6] and adaptive scaling [29] have been proposed to cope with this problem, it is not clear such methods are able to effectively track articulated objects whose shapes vary rapidly and significantly.

In a somewhat different direction, the use of Haar-like features plays an important role in the success of real-time object detection [3]. However, fast algorithms for computing Haar-like features and histograms such as integral images [3] or integral histograms [4] require rectangular windows to model the target's shape. Consequently, it is not straightforward to apply efficient methods to track and detect articulated object with varying shapes. Haar-like and related features play a significant role in several recent work on online boosting and its applications to tracking and object detection [30]. One interesting aspect of this latter work is to treat tracking as sequential detection problems, and an important component in the tracking algorithm is the online construction of an object-specific detector. However, the capability of the tracker is somewhat hampered by the Haar-like features it uses in that this invariably requires the shapes of the target to be well approximated by rectangular windows.

Finally, our algorithm shares some similarity with the part-based object detection algorithm proposed in [31] as both algorithms use rectangular blocks to define the target object. However, the similarity is only superficial since, in our method, there is no

[1] An early version of this work was presented in [9].

specific part definition as the blocks are online adjusted to provide the coverage for the foreground target only. While decompositions of the target using rectangular blocks are employed in both method, the decomposition in our case is geometrical with the explicit purpose of covering the foreground and accurately estimate the intensity histogram while theirs is semantical in that each block or part has its own unique appearance and characteristic. Our goal is to have a general-purpose tracker and this necessarily requires us to avoid detecting object parts as it will involve more extensive training and require more assumptions on the target's appearance.

3. Tracking algorithm

We present the details of the proposed tracking algorithm in this section. The output of the proposed tracker consists of a rectangular window enclosing the target in each frame. Furthermore, an approximated boundary contour of the target is also estimated, and the region it encloses defines the estimated target region. Our objective is to achieve a balance among the three somewhat conflicting goals of efficiency, accuracy and robustness. Specifically, we treat the tracking problem as a sequence of detection problems, and the main feature that we use to detect the target is the intensity histogram. The detection process is carried out by matching foreground intensity histogram and we employ integral histograms for efficient computation. In the following discussion, we will use the terms histogram and density interchangeably. The main technical problem that we solve within the context of visual tracking is how to approximate the foreground histogram under significant shape variations so that efficient and accurate articulated object tracking is possible under the general assumption (held by most tracking algorithms) that the foreground histogram stays roughly stationary.

The high-level outline of the proposed algorithm is shown in Fig. 2. It consists of four sequential steps: detection, scaling, refinement, and update. At the outset, the tracker is initialized with the contour of the target, it then automatically determines the initial tracking window $\mathbf{W}$ and K rectangular blocks $\mathbf{B}_i$ as well as their weights λ_i according to the procedure described below. The foreground intensity histogram $\mathbf{H}_0^f$ for the initial frame is kept throughout the sequence.

The shape of the foreground target is approximated by K rectangular blocks, $\mathbf{B}_i$, $1 \leqslant i \leqslant K$, within the main tracking window $\mathbf{W}$ as shown in Fig. 3. The positions of the blocks within the tracking window are adaptively adjusted throughout the tracking sequence,

Tracking Algorithm Outline

1. Detection

The entire image is scanned and the window with the highest similarity is determined to be the tracking window $\mathbf{W}^*$.

2. Scaling

Size of tracking window $\mathbf{W}^*$ is adjusted according to scale of target.

3. Refinement

Within $\mathbf{W}^*$, the target is segmented out using a graph-cut based segmentation which divides the tracking window between foreground and background regions. The segmentation uses both estimated foreground and background distributions.

4. Update

Block configuration is adjusted locally based on the segmentation result obtained in the previous step. The non-negative weights λ_i of the blocks are recomputed.

Fig. 2. High-level outline of the proposed tracking algorithm.

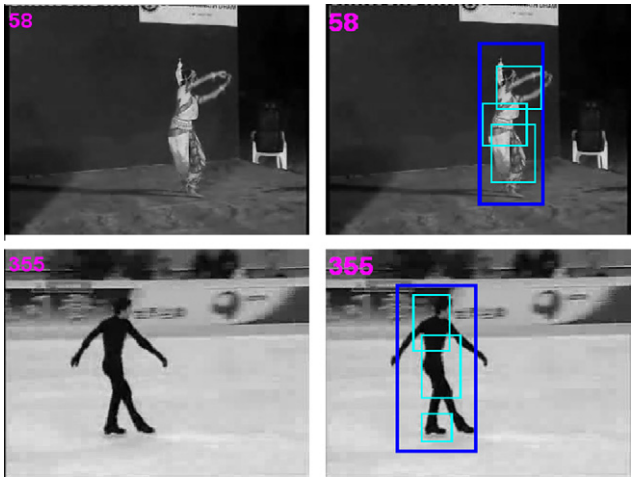


Fig. 3. Left: Examples of articulating targets. Right: Given the contour of the target object, we select those blocks (and associated weights) with non-empty intersection with the interior region of the target defined by the contour. Blocks containing only background pixels are not selected. The importance of a block is proportional to the percentage of its pixels belonging to the foreground.

and they may have some overlaps to account for extreme shape variations. At each frame t, the tracker maintains the following: (i) a tracking window $\mathbf{W}_t$ with a block configuration, (ii) a foreground histogram $\mathbf{H}_t^f$ represented by a collection of "local foreground histograms", $\mathbf{H}_t^{\mathbf{B}_i^f}$ and their associated weights λ_i, computed from the blocks, and (iii) a background histogram $\mathbf{H}_t^b$. The tracker first detects the most likely location of the target by scanning the entire image (i.e., the window with the highest similarity when compared with the tracking window $\mathbf{W}$).

After detection, tracking window size can be adjusted to make it tightly enclose the target without unnecessary background pixels. Note that for tracking articulated objects, it is inevitable for tracking windows to enclose some background pixels as the shapes and sizes of targets vary significantly. We introduce an adaptive scaling step to update the size of the tracking window based on scale of the target object.

In the refinement step, the tracker works exclusively in the detected window and the target is segmented from the background using the current foreground density. This result is then used in the update step to adjust the block positions in the tracking window $\mathbf{W}$ and then the weights assigned to each block is recomputed. In addition, the background density $\mathbf{H}_t^b$ is also updated. Ideally, the number of blocks K and the size of each block $\mathbf{B}_i$ should be adaptively determined during tracking. However, in this paper, we fix the number of blocks, while the position of each block is adjusted accordingly to account for shape variation.

While it is expected that the union of the blocks will cover most of the target, these blocks will nevertheless contain both foreground and background pixels. This happens often when the shape of the target object is far from convex and exhibits strong concavity. In particular, blocks containing large percentages of background pixels should be down weighted in their importance when compared with blocks that contain mostly foreground pixels. Therefore, each block $\mathbf{B}_i$ is assigned a weight λ_i, which will be used in all three steps. In this framework, the shape information is represented by the block configuration and the associated weights. Compared with other formulations of shape priors [24,26], it is a rather fuzzy representation of shapes. However, this is precisely what is needed here since rapid and sometime extreme shape variation is expected, the shape representation should not be rigid and too heavily constrained so as to permit greater flexibility in anticipating and handling hitherto unseen shapes.

3.1. Detection

For each frame, the tracker first scans the entire image to locate the target object. As with many other histogram-based trackers, the target window $\mathbf{W}^*$ selected in this step is the one that has the maximum foreground similarity measure with respect to initial tracking window $\mathbf{W}$. After scanning all possible candidate window $\mathbf{W}'$ in current frame, we select the window as $\mathbf{W}^*$, which minimizes our proposed distance function $\mathbf{D}$.

$$\mathbf{W}^* = \min_{\mathbf{W}'} \mathbf{D}(\mathbf{W}', \mathbf{W}). \tag{1}$$

The distance function $\mathbf{D}$ is constructed from local foreground histograms computed from the blocks as follows. First, we transfer the block configuration of the tracking window $\mathbf{W}_{t-1}$ onto each scanned window $\mathbf{W}'$, and accordingly, we can evaluate K local foreground histograms in each of the transferred blocks. The local foreground histogram $\mathbf{H}_t^{\mathbf{B}_i^f}$ for the block $\mathbf{B}_i$ is the intersection of the raw histogram $\mathbf{H}_t^{\mathbf{B}_i}$ with the initial foreground histogram of the corresponding block:

$$\mathbf{H}_t^{\mathbf{B}_i^f}(b) = \min\left(\mathbf{H}_t^{\mathbf{B}_i}(b), \mathbf{H}_0^{\mathbf{B}_i^f}(b)\right),$$

where b indexes the bins. The distance function is defined as the weighted sum of the Bhattacharyya distance between the densities $\mathbf{H}_0^{\mathbf{B}_i^f}(b)$ and $\mathbf{H}_t^{\mathbf{B}_i^f}(b)$,

$$\mathbf{D}(\mathbf{W}', \mathbf{W}) = \sum_{i=1}^{K} \lambda_i \rho\left(\mathbf{H}_0^{\mathbf{B}_i^f}, \mathbf{H}_t^{\mathbf{B}_i^f}\right) \tag{2}$$

where λ_i is the weight associated to block $\mathbf{B}_i$ and ρ is the Bhattacharyya distance between two densities,

$$\rho\left(\mathbf{H}_0^{\mathbf{B}_i^f}, \mathbf{H}_t^{\mathbf{B}_i^f}\right) = \sqrt{1 - \sum_{b=1}^{N} \sqrt{\mathbf{H}_0^{\mathbf{B}_i^f}(b)\mathbf{H}_t^{\mathbf{B}_i^f}(b)}}$$

where N is the number of bins. Since the blocks are rectangular, all histograms can be computed by a few subtractions and additions using integral histograms. Because of λ_i, $\mathbf{D}$ will down weight blocks containing more background pixels, and this is desirable because it provides some measure against background noise and clutters. Note that comparing with most histogram-based trackers, which invariably uses only one histogram intersection, the distance function $\mathbf{D}$ defined in Eq. 2 actually encodes some amount of shape and spatial information through the block configuration and their weights.

3.2. Scaling

After detection step, we randomly vary size of the tracking window while keeping the target object at the center of these windows. For each scaled window $\mathbf{W}'' = scale(\mathbf{W}^*, s_h, s_w)$, we estimate the foreground density $\mathbf{H}_{t,W''}^f = \left\{\mathbf{H}_{t,W''}^{\mathbf{B}_i^f}\right\}$ and background density $\mathbf{H}_{t,W''}^b$ (the values of s_h and s_w are selected within a range, i.e., $0.8 \leqslant s_h, s_W \leqslant 1.2$). We select the scale of a tracking window within which its foreground matching and background mismatching is maximized. In other words, the adaptive window $\mathbf{W}^*$ should minimize following objective function,

$$\mathbf{W}^* = \min_{\mathbf{W}''} \alpha \sum_{i=1}^{K} \lambda_i \rho\left(\mathbf{H}_0^{\mathbf{B}_i^f}, \mathbf{H}_{t,W''}^{\mathbf{B}_i^f}\right) + (1-\alpha)\left(1 - \rho\left(\mathbf{H}_0^f, \mathbf{H}_{t,W''}^b\right)\right) \tag{3}$$

where α is a parameter for specifying weights of two matching terms. We set α to 0.3 in all our experiments to put more weights on background mismatching term for scale selection. With this scheme, we can better determine the window that tightly encloses the target object. Fig. 4 shows some tracking results using fixed and adaptive scaling.

3.3. Refinement

Once the global scan produces the tracking window $\mathbf{W}^*$ in which the target is located, the next step is to extract an approximate foreground region so that the shape variation can be better

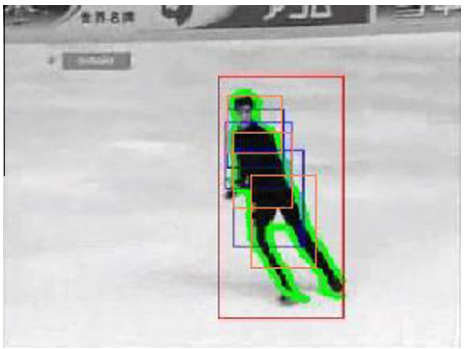


Fig. 5. Blue: The block configuration from the previous frame. Green: The contour is estimated using a fast graph-cut algorithm. Red: The blocks are repositioned using a greedy strategy to provide a maximal coverage of the target. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

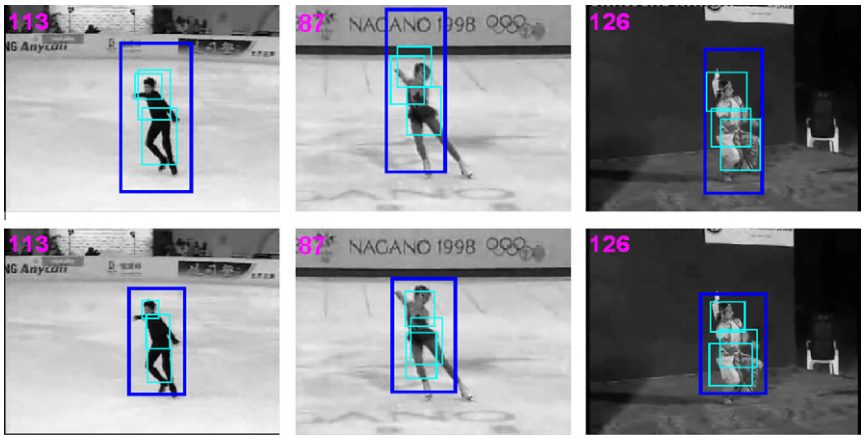


Fig. 4. Top: Tracking with fixed scale window. Bottom: Tracking using adaptive scaling window.

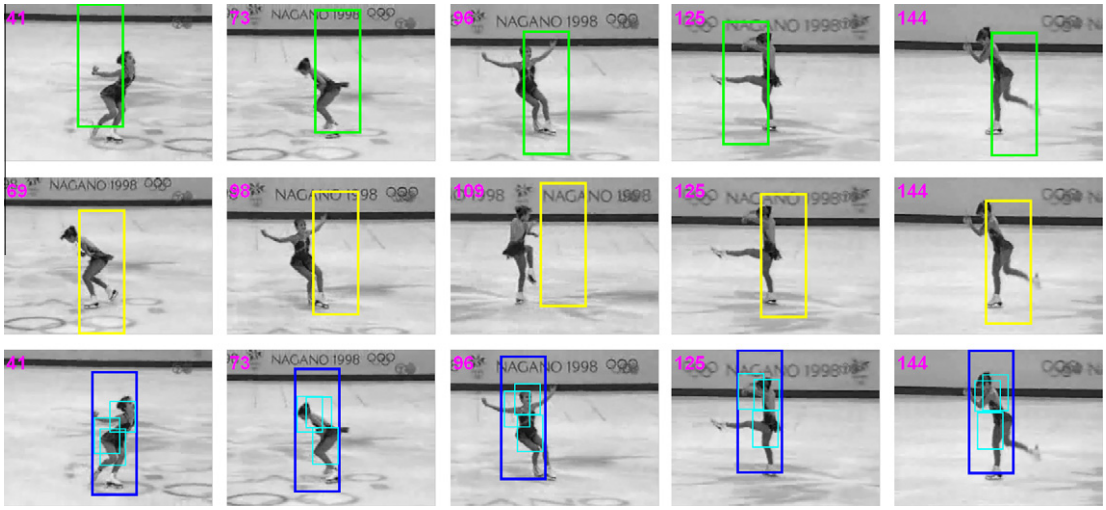


Fig. 6. Top: Tracking results using only integral histogram. Middle: Tracking results using the mean-shift tracker. Bottom: Tracking results using the proposed algorithm. The shape variation in this sequence is substantial. Notice the unsatisfactory result produced by the integral histogram tracker. The inaccurate tracking results are difficult to be utilized by other vision applications.

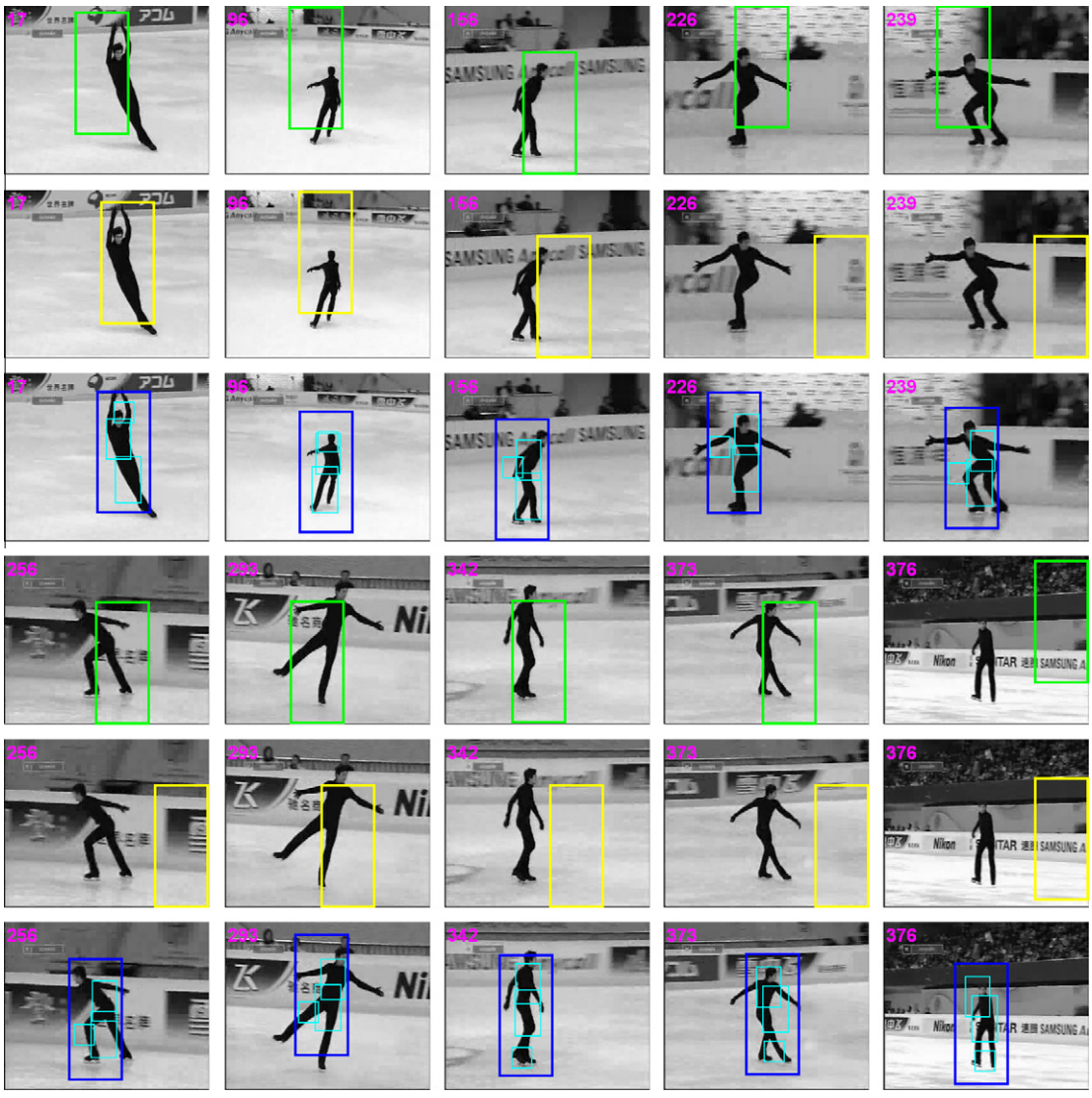


Fig. 7. First and fourth rows: Tracking results using only integral histogram. Second and fifth rows: Tracking results using the mean-shift tracker. Third and sixth rows: Tracking results using the proposed algorithm.

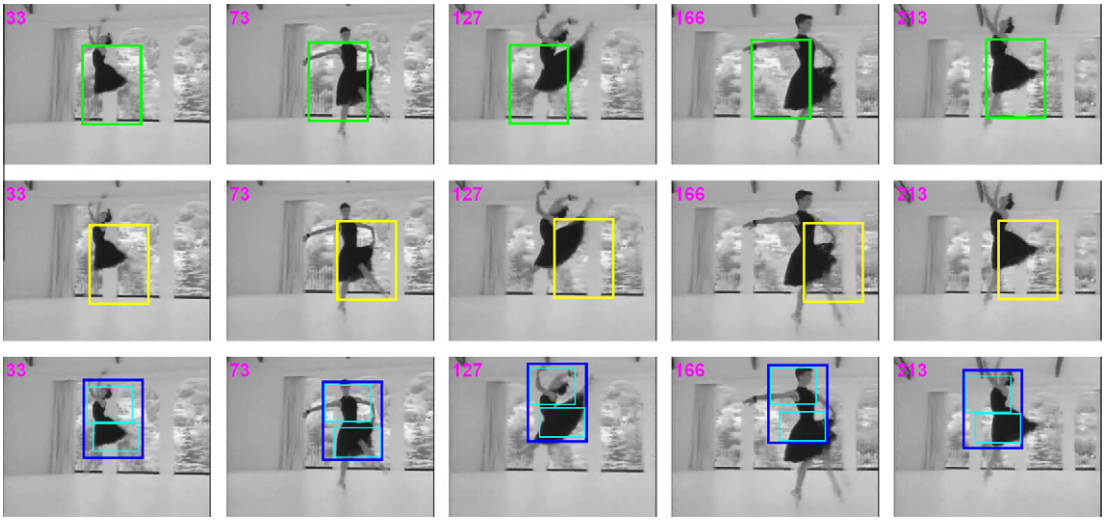


Fig. 8. Top: Tracking results using only integral histogram. Middle: Tracking results using the mean-shift tracker. Bottom: Tracking results using the proposed algorithm. Note the target undergoes large shape and scale variation.

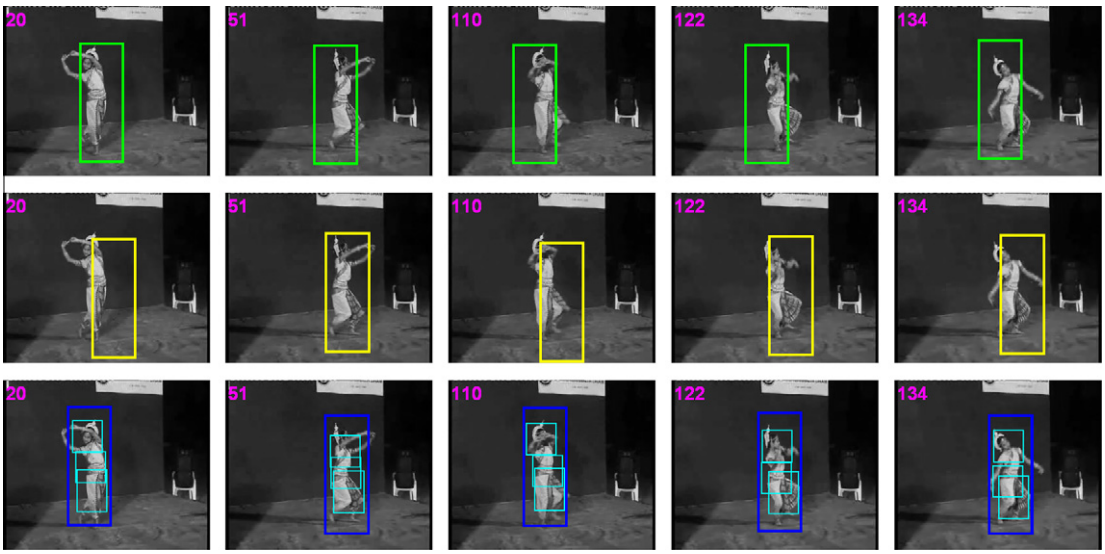


Fig. 9. Top: Tracking results using only integral histogram. Middle: Tracking results using the mean-shift tracker. Bottom: Tracking results using the proposed algorithm. Note the target undergoes significant shape and appearance variation.

accounted for. We apply a graph-cut segmentation algorithm to segment out the foreground region in $\mathbf{W}^*$. Previous work on this type of segmentation in the context of visual tracking (e.g., [24,26,32]) always define the cost function in the form

$$E = E_A + \gamma E_S,$$

where E_A and E_S are terms relating to appearance and shape, respectively. However, we have found that hard coding the shape prior in a separate term E_S is more of a hindrance than help in our problem because of the extreme shape variation as strong shape priors without dynamic information often lead to unsatisfactory results. Instead, our solution will be to use only the appearance term E_A but incorporating shape component through the definition of foreground density.

Specifically, let p denote a pixel and $\mathscr{P}$ denote the set of all pixels in $\mathbf{W}^*$. Let P_B denote the background density that we estimated in the previous frame, and P_i, $1 \leqslant i \leqslant K$ the foreground density from $\mathbf{B}_i$ (by normalizing the histogram $\mathbf{H}_i^{\mathbf{B}_i}$). Furthermore, we will denote P_f the foreground density obtained by normalizing the current foreground histogram $\mathbf{H}_t^f$. Following [33,32], the graph-cut algorithm will minimize the cost function

$$E(C_p) = \mu \sum_{p \in \mathscr{P}} R_p(C_p) + \sum_{(p,q) \in \mathscr{N}: C_p \neq C_q} B_{p,q}, \tag{4}$$

where $C_p : \mathscr{P} \to \{0, 1\}$ is a binary assignment function on $\mathscr{P}$ such that for a given pixel p, $C(p) = 1$ if p is a foreground pixel and 0 otherwise.[2] μ is a weighting factor and $\mathscr{N}$ denotes the set of neighboring pixels. We use μ to 0.5 in our algorithm. We define

$$B_{p,q} \propto \frac{\exp((I(p) - I(q))^2 / 2\sigma^2)}{\|p - q\|},$$

where $I(p)$ is the intensity value at pixel p and σ is the kernel width. The term $R_p(C_p)$ is given as

[2] We will denote $C(p)$ by C_p.

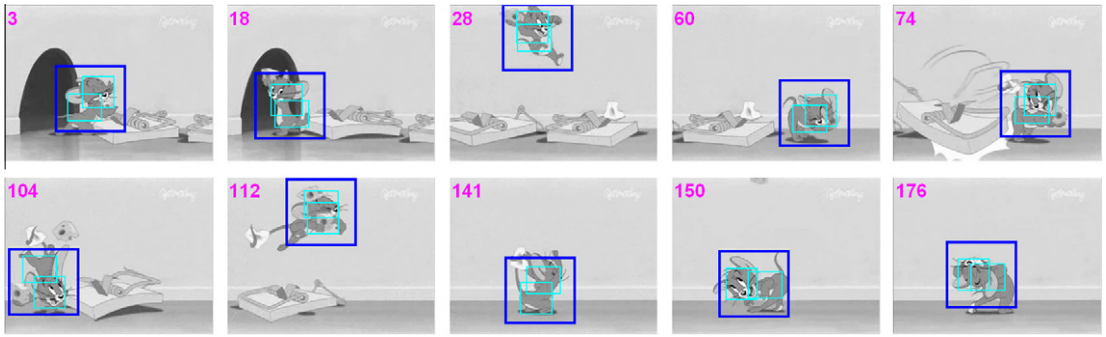


Fig. 10. Tracking results using the proposed algorithm. Note there is a large variation in shape, scale and appearance of the target. In addition, the target exhibits ballistic and non-rigid motions.

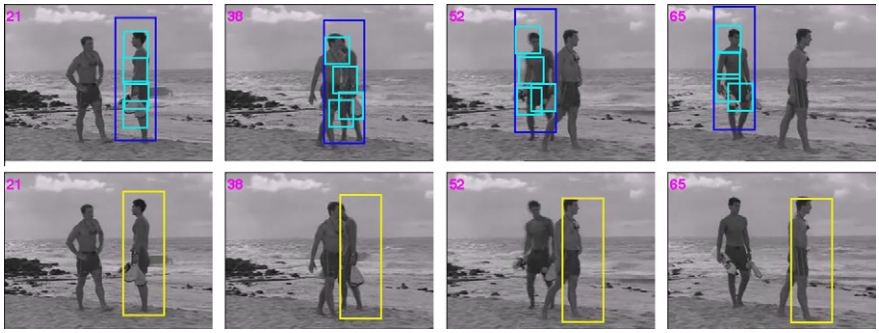


Fig. 11. Tracking occluded target. Top row: Tracking result by the proposed algorithm. Bottom row: Tracking result using the mean-shift tracker. The proposed tracker can recover the target after occlusion, while the mean-shift tracker fails.

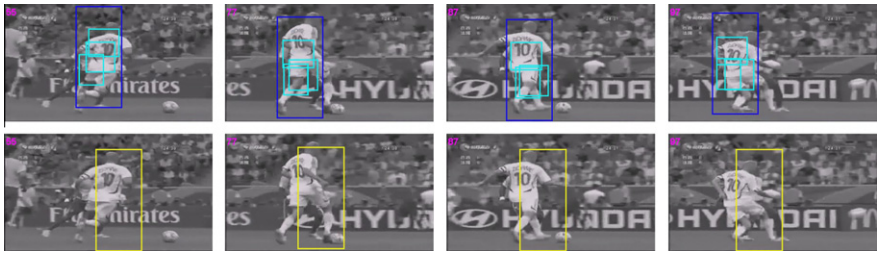


Fig. 12. Tracking with a cluttered background. Top row: Tracking result using the proposed algorithm. Bottom row: Tracking result using the mean-shift tracker. The proposed tracker can track the target more consistently than the mean-shift tracker.

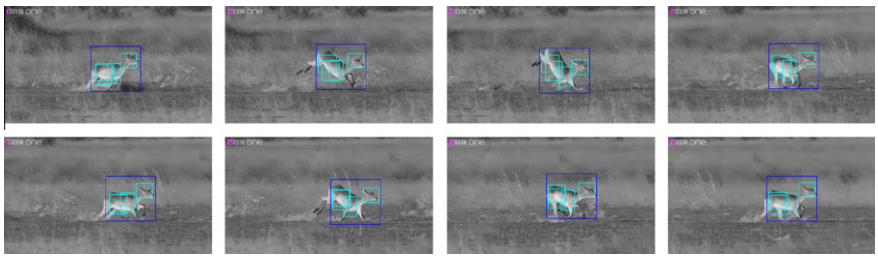
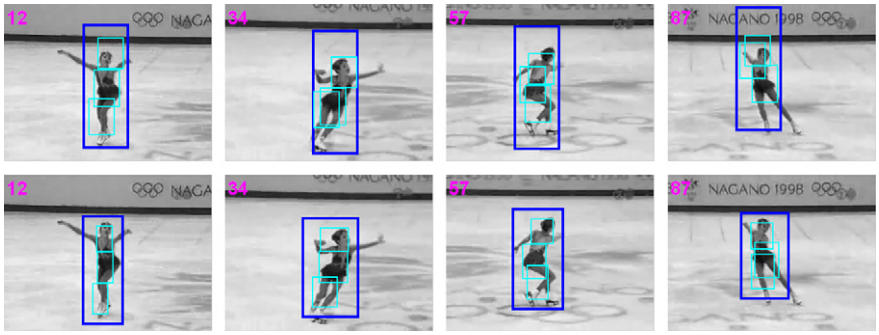


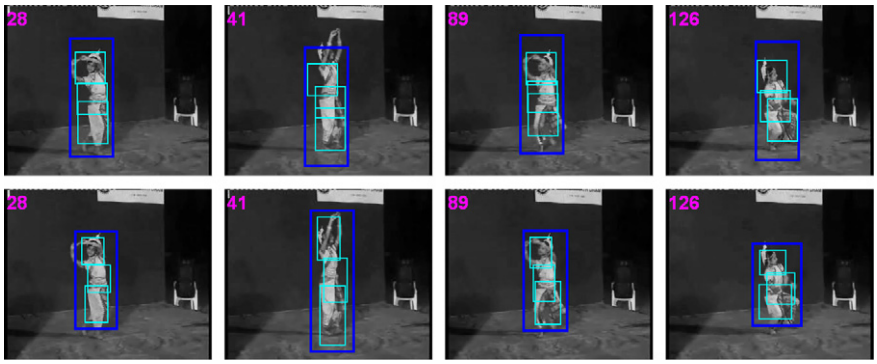
Fig. 13. Tracking of a wildlife target using the proposed algorithm. Eight selected frames from a sequence of 100 frames are shown there.

$$R_p(C_p = 0) = -\log P_F(I(p), p)$$
$$R_p(C_p = 1) = -\log P_B(I(p)),$$

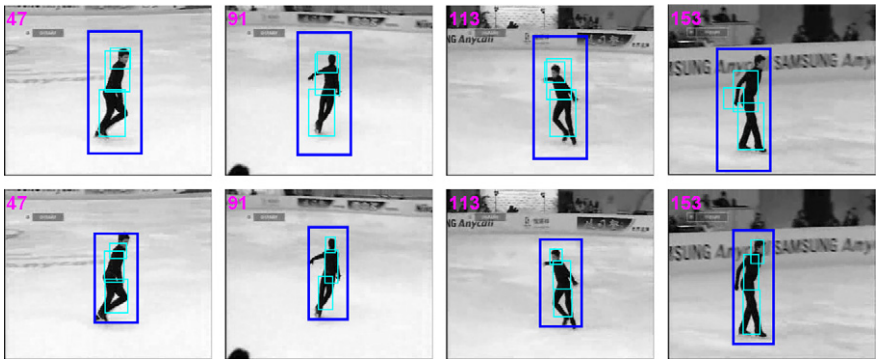
where $P_F(I(p)) = P_i(I(p))$ if $p \in \mathbf{B}_i$, and $P_F(I(p)) = P_f(I(p))$ if p is not contained in any block $\mathbf{B}_i$. Note that the shape information is now implicitly encoded through P_F. A fast combinatorial algorithm with polynomial complexity exists for minimizing the energy function E, based on the problem of computing a minimum cut across a graph [33]. Since we only perform the graph-cut in a (relatively) small window, this can be done very quickly and does not substantially increase the computational load. Fig. 5 presents one segmentation result where the extracted target contour is shown in green.



(a) Female skater



(b) Male skater



(c) Indian dancer

Fig. 14. Top rows: Tracking results of our algorithm with fixed scale. Bottom rows: Tracking results of our algorithm with adaptive scaling.

Fig. 15. Examples of ground truth windows for articulated targets.

3.4. Update

After the object contour is extracted from the segmentation result, we update the positions of the blocks $\mathbf{B}_i$ within $\mathbf{W}^*$. The idea is to locally adjust these blocks so that they provide a maximal coverage of the segmented foreground region. We employ a greedy strategy to cover the entire segmented foreground by moving each block locally using a priority based on their sizes. Note that such an approach (i.e., local jittering) has often been adopted in object detection and tracking algorithms for later-stage refinement and fine-tuning. Fig. 5 shows one result of this block adjustment (shown in red).

As the foreground definition is now known, we can compute the foreground histogram $\mathbf{H}_t^{\mathbf{B}_i^f}$ from each block B_i. After that, we recompute corresponding block weights according to following equation

Table 1
Location errors of our tracker, the integral histogram-based tracker [4] and the mean-shift tracker [2].

Sequence	Integral histogram tracker			Mean shift tracker			Our proposed tracker		
	Max	Mean	Std	Max	Mean	Std	Max	Mean	Std
Center location errors (in pixels)									
Female skater	55.2	29.7	11.8	69.2	26.9	16.1	33.1	13.1	6.8
Male skater	165.0	45.7	29.6	168.0	69.8	26.6	35.0	13.6	7.5
Indian dancer	25.5	12.1	5.8	49.5	25.4	11.6	29.8	8.4	4.8
Dancer	58.9	31.9	10.2	61.5	41.6	11.8	33.7	16.1	6.9

Table 2
Coverage errors of our tracker, the integral histogram-based tracker [4] and the mean-shift tracker [2].

Sequence	Integral histogram tracker			Mean shift tracker			Our proposed tracker		
	Max	Mean	Std	Max	Mean	Std	Max	Mean	Std
Coverage errors (in %)									
Female skater	79.91	61.15	9.37	100.0	63.76	16.29	63.45	48.24	9.39
Male skater	100.00	67.48	13.63	100.0	80.12	19.01	74.79	53.28	11.77
Indian dancer	66.68	50.19	6.38	89.21	61.25	13.13	58.96	46.26	5.66
Dancer	87.59	62.55	9.42	87.38	71.49	8.95	69.82	48.23	8.70

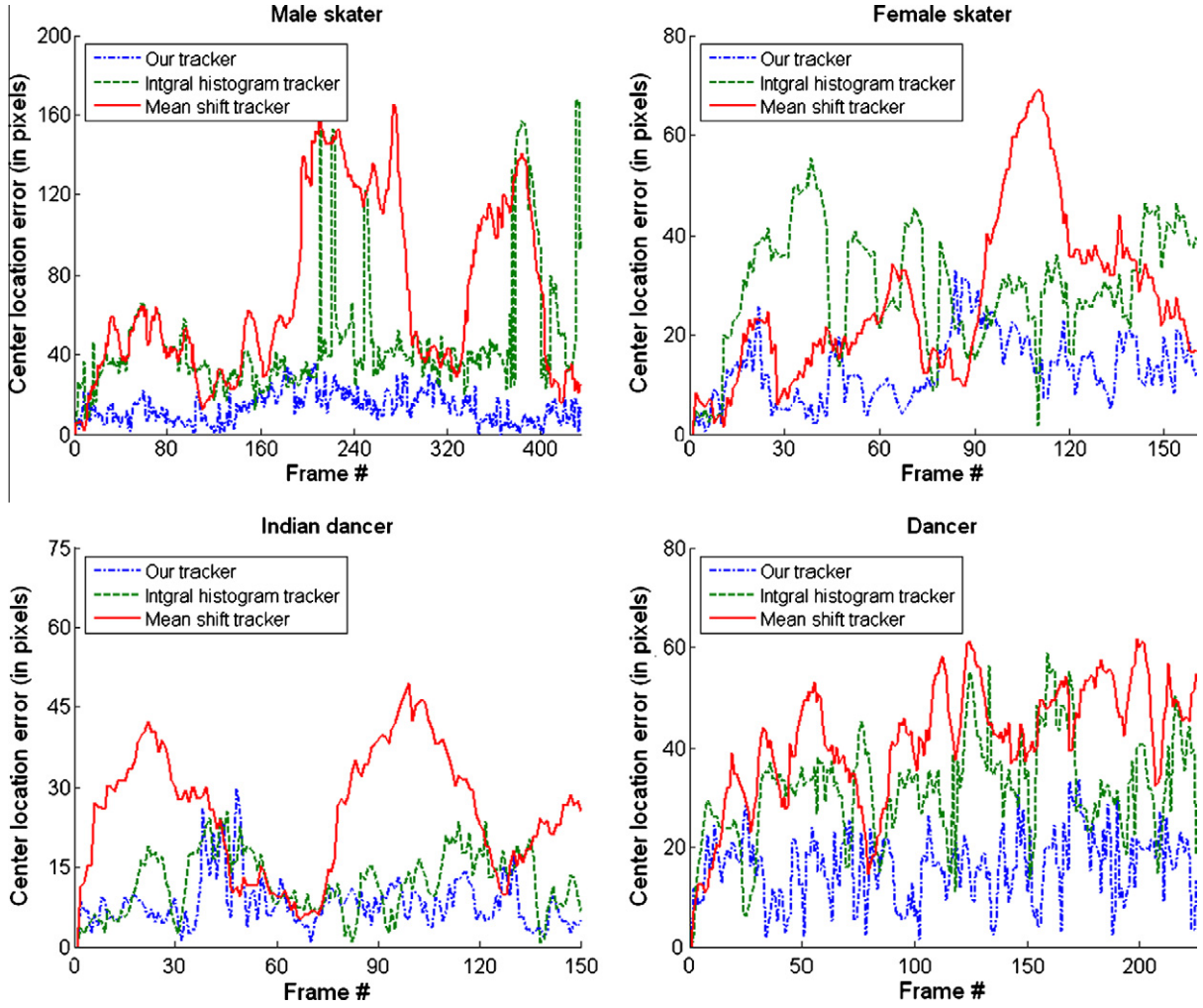


Fig. 16. Location errors of three trackers. Top row: (Left) Male skater, (Right) Female skater, Bottom Row: (Left) Indian dancer and (Right) Dancer sequence.

$$\lambda_i = \frac{\sum_{b=1}^{N} \mathbf{H}_t^{\mathbf{B}_i^f}(b)}{\sum_{p \in W^*} C(p)}.$$

Weights λ_i are normalized to enforce the requirement that their sum is one.

3.5. Discussion

Comparing with the recent work that employ discriminative models (classifiers) for tracking (e.g., [30]), our approach is mainly generative through the use of intensity histograms. While we as-

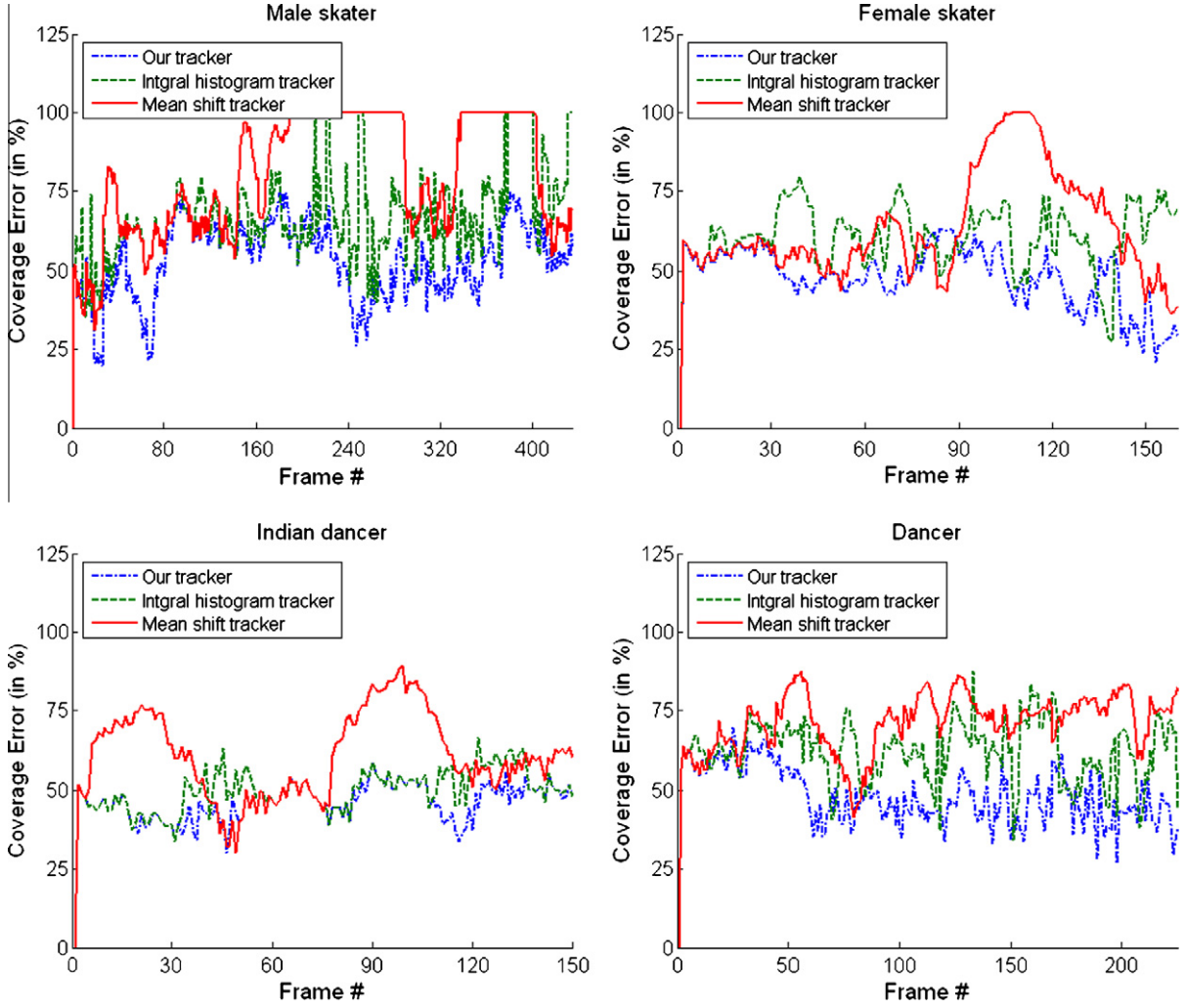


Fig. 17. Coverage errors of three trackers. Top row: (Left) Male skater, (Right) Female skater, Bottom row: (Left) Indian dancer and (Right) Dancer sequence.

Table 3
Center location errors of our tracker with fixed and adaptive scaling.

Sequence	Fixed scale window			Adaptive scaling window		
	Max	Mean	Std	Max	Mean	Std
Center location errors (in pixels)						
Female skater	33.11	12.11	7.72	21.84	9.15	4.98
Male skater	22.64	10.28	5.27	26.73	9.48	5.56
Indian dancer	29.83	8.36	4.75	24.63	6.58	4.35

Table 4
Coverage errors of our tracker with fixed and adaptive scaling.

Sequence	Fixed scale window			Adaptive scaling window		
	Max	Mean	Std	Max	Mean	Std
Coverage error (in %)						
Female skater	63.49	52.44	5.89	59.72	44.95	6.07
Male skater	72.64	52.02	13.37	71.07	44.75	11.08
Indian dancer	58.95	46.26	5.66	57.29	37.29	8.41

sume that the intensity distribution stays stationary, the features we constantly update are the block configurations and the associated weights. Online appearance updates (e.g., [30,34,35]) have been shown to be effective for tracking rigid objects. However, as the examples shown in these work are almost without significant shape variation, it is difficult to see that these techniques can be generalized immediately to handle shape updates. On the other hand, shape variation has often been managed in visual tracking algorithms using shape templates learned offline and the dynamics among the templates [17,24,26,21]. It is also not clear how these algorithms can deal with sequences containing unseen shapes or dynamics. Instead of "hard coding" the shape prior, our algorithm provides a soft update on shape in the form of updating the block configuration, and the update is constrained by the appearance model through the requirement that the foreground intensity distribution stays roughly stationary.

Our use of adaptive block structure is easily associated with recent work that track and detect parts of an articulated object (e.g., [36,37]). However, our goal and motivation are quite different in that the blocks are employed for providing a convenient structure to approximate the object's shape and estimating intensity histogram. Our objective is an accurate and efficient tracker, not the precise localization of parts, which in general requires substantially more processing. Nevertheless, it is interesting to investigate the possibility of applying our technique to this type of tracking/detection problem, and we will leave this to future work.

4. Experiments and results

The proposed algorithm has been implemented in MATLAB with some optimization using MEX C++ subroutines. The code and data

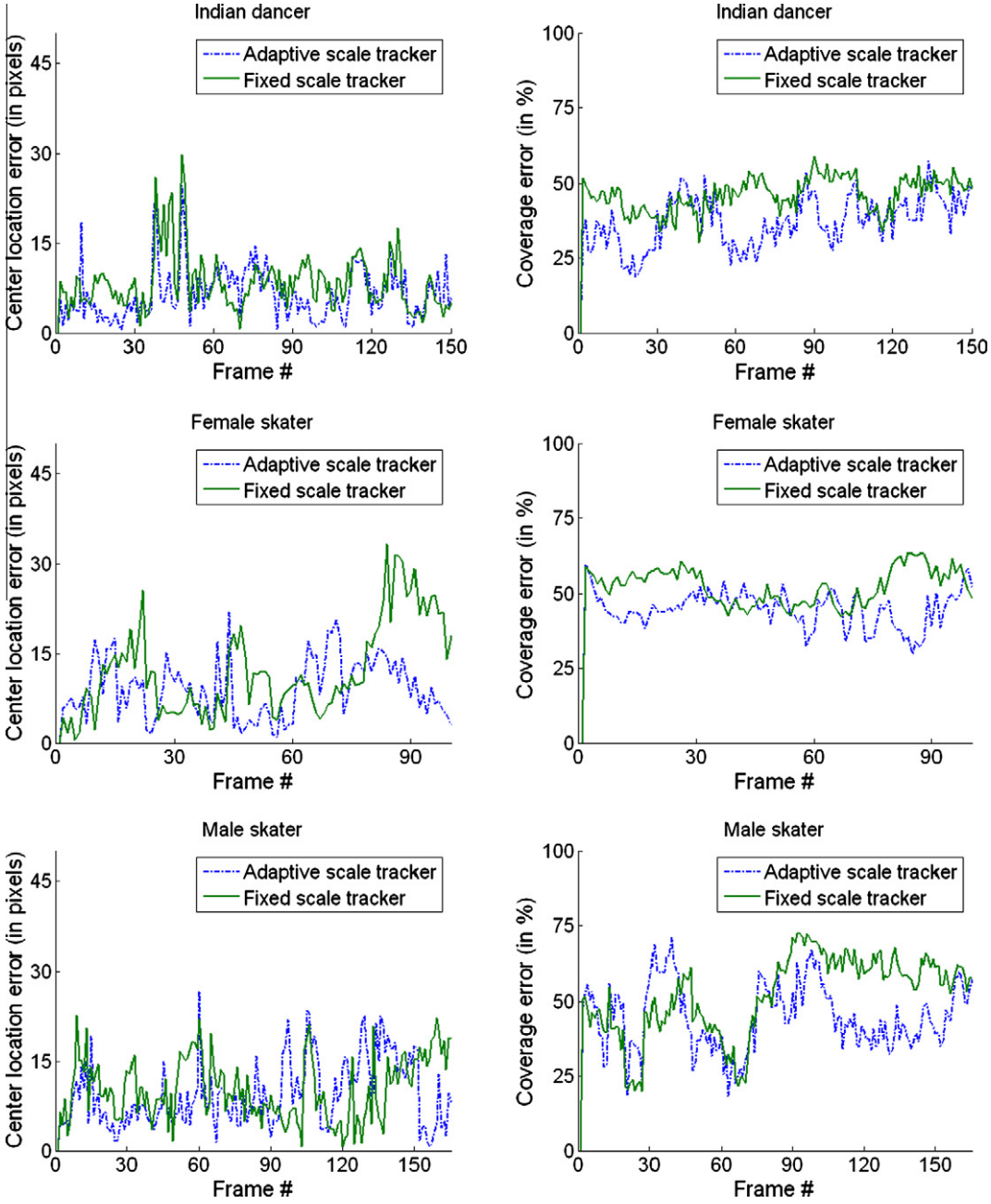


Fig. 18. Comparison between our fixed and adaptive scaling window based tracker. Top: Indian dancer sequence. Middle: Female skater sequence. Bottom: Male skater sequence. Center location errors are shown in **left** column and **right** column contains coverage errors.

are available at http://www.cise.ufl.edu/smshahed/tracking.htm. In this implementation, we use intensity histograms with 16 bins for grayscale videos. Each video consists of 320×240 pixel images recorded at 15 frames per second. The number of blocks, K, is set to two or three. The tracker has been tested on a variety of video sequences, and eight of the most representative sequences are reported in this paper. We compare tracking results of our algorithm with a tracker using the plain integral histogram [4] and mean-shift tracker [2]. On a Dell 3GHz machine, our tracker runs at 3.7 frames per second while the integral histogram tracker has a slightly better performance at 4 frames per second.[3] The additional overhead incurred in our algorithm comes from the update of block configuration, which amounts to a small fraction of the time spent on computing the integral histogram over the entire image. However, experimental comparisons show that this negligible overhead in run-time complexity allows our tracker to consistently produce much more stable and satisfactory tracking results.

In the following experiments, for initialization, we manually outline contour of the target in the first frame, and for the experiments, all trackers start with the same tracking window. The sequences shown below are all collected from the web. In these sequences, the foreground targets undergo significant appearance changes, which is mainly due to shape variation. We first present the qualitative tracking results and then quantitative comparisons with ground truth data.

[3] In our MATLAB implementation, both algorithms share same MEX C++ subroutines.

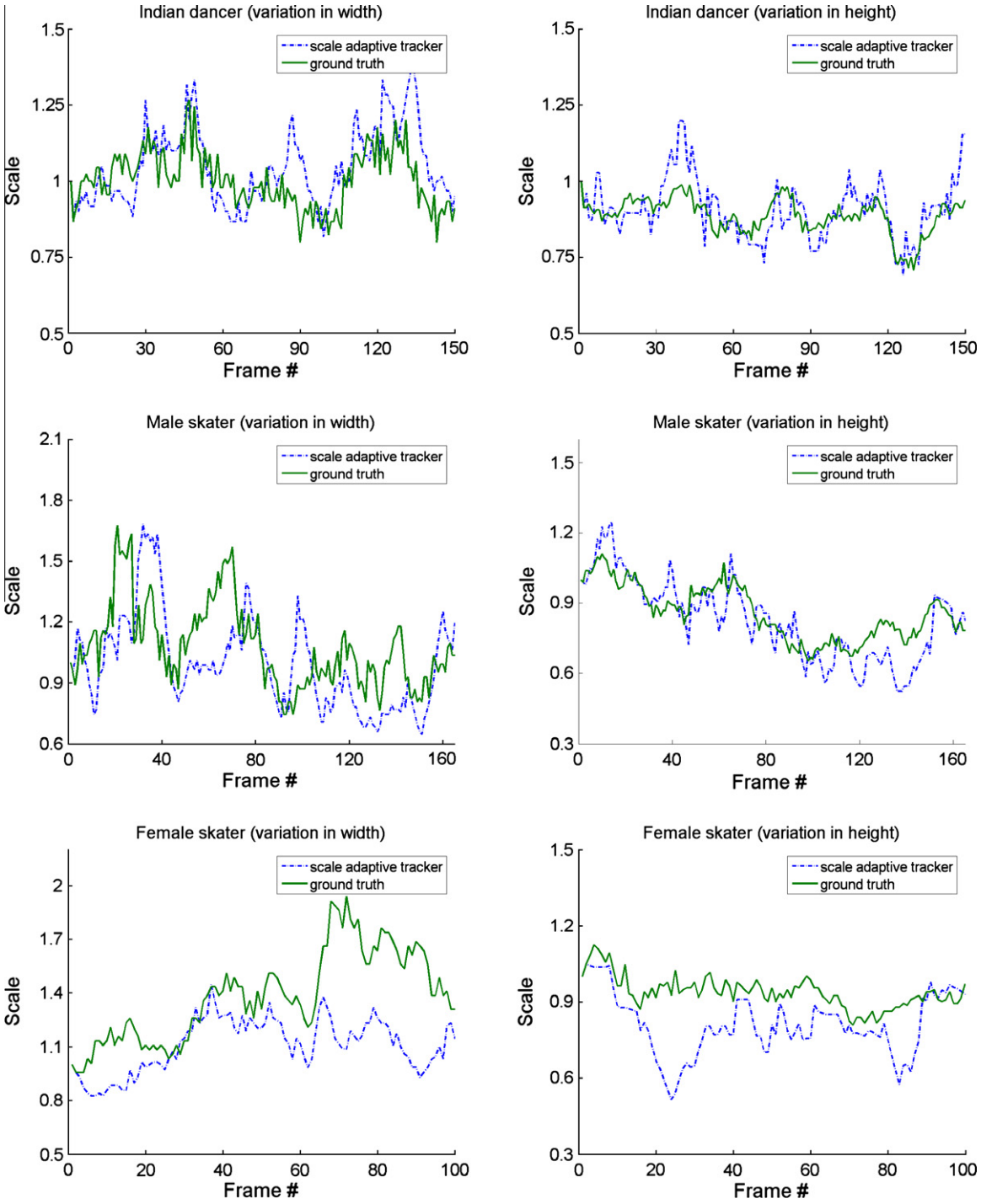


Fig. 19. Results using our algorithm with fixed and adaptive scaling. Top: Indian dancer sequence. Middle: Male skater sequence. Bottom: Female skater sequence. Variations in width and height are shown in the left and the right columns, respectively.

Table 5
Errors of adaptive scale window size.

Sequence	Error in width		Error in height	
	Mean	Std	Mean	Std
RMS error in tracking window size (in %)				
Indian dancer	9.22	7.13	6.13	5.02
Male skater	8.18	6.46	20.15	13.94
Female skater	27.84	21.05	13.55	10.76

4.1. Female skater

The female skating sequence contains over 150 frames, and the dazzling performance is accompanied by an equally dazzling pose variation. As shown in Fig. 6, while the background is relatively simple, the integral histogram tracker and the mean-shift tracker are not able to locate the skater accurately, producing jittered and unstable tracking windows. In particular, it is impossible to utilize this unsatisfactory tracking result for other vision applications such as gait or pose recognition. However, our tracker is able

to track the skater well and provides tracking windows that are much more accurate and consistent. As shown in the figures, one major reason for this improvement is that the spatial locations of the blocks are updated correctly by our algorithm as the skater undergoing significant changes in pose.

4.2. Male skater

The second sequence contains 435 frames with a figure skater performing in a cluttered environment, and the tracking results using our method and the mean-shift algorithm are shown in Fig. 7. Our algorithm is able to accurately track the skater throughout the whole sequence (i.e., the tracking windows are accurately centered around the skater) as shown in Fig. 7 while the integral histogram tracker again produces unsatisfactory results. Perhaps more importantly, our algorithm is able to track the skater across shots taken from two different cameras (e.g., from frame 372 to 373 and onwards), which is difficult to handle for most visual tracking algorithms, particularly those using differential techniques. The results also demonstrate the advantage of having the capability to efficiently scan the entire image for the target as the mean-shift tracker loses the target when the camera angle changes (e.g., frame 372–323 and onwards).

4.3. Dancers and cartoon

The third and fourth sequences contain two stylistically different dances. In both sequences, adverse conditions such as cluttered backgrounds, scale changes and rapid movements have significance presences, and the shape variations in them are even more pronounced when compared with the two previous sequences. In both experiments (Figs. 8 and 9), our tracker is able to track the dancers accurately while the integral histogram and the mean-shift tracker fail to produce consistent and accurate results.

Fig. 10 presents the tracking results using a very challenging sequence in which there is a large variation in shape, scale, and appearance of the target. Furthermore, the target undergoes ballistic movements. Notwithstanding these difficulties, the proposed tracker is able to follow the target accurately using only two blocks.

In Fig. 11, we apply our tracking algorithm to a sequence in which the target object is fully occluded at some point. In this sequence, two persons walk pass each other and the person being tracked is fully occluded. As shown in Fig. 11, our tracking algorithm is able to track the target correctly before and after occlusion while the mean-shift tracker is confused by occlusion and lose the target afterward.

We test our tracker with a sequence in which the target soccer player appears in a cluttered and changing background. As shown in Fig. 12, our tracker produces more stable results than the mean-shift tracker. And finally, in Fig. 13, we apply the proposed tracker to a wildlife sequence with natural scene as the background.

4.4. Tracking with adaptive scale

As described in Section 3.2, our tracker is able to adjust the size of tracking window to tightly enclose the target object. In this section, we present some results of our tracker with fixed and adaptive scaling. As shown in Fig. 14, our tracker with adaptive scaling is able to better enclose the target object than the one with fixed scaling although both are centered at the same target locations.

4.5. Quantitative analysis

For quantitative performance evaluation of our tracker, we manually label the ground truth by selecting the minimal window that encloses the target in every frame. As our sequences contain articulating targets, we do not include parts (hands, legs) in the ground truth window when they are spread out too much. Some of these examples are shown in Fig. 15.

We use two error metrics for quantitative evaluations. The first one measures the deviation of the center of the tracking window from the ground truth, whereas the second one measures the coverage of the tracking window against the ground truth. Certainly an optimal tracker is expected to have small errors in both metrics. Quantitative performance of our tracker, the integral histogram-based tracker and the mean-shift tracker with respect to these error measurements are summarized in Tables 1 and 2 and Fig. 16 as well as Fig. 17. We observe that our tracker outperforms the other two trackers by a large margin as our tracker achieves the lowest mean errors in all sequences with small standard deviations.

We present quantitative comparisons of our trackers with fixed and adaptive scaling. From each original sequence, we select a subsequence which contains substantial scale variation of target for experiments. Experimental results, as summarized in Tables 3 and 4 and Fig. 18, show that adaptive scaling improves the accuracy in location in all cases and coverage in most cases.

As the size of the target objects varies significantly in our experiments, it is of great interest to further analyze whether the proposed algorithm is able to adjust the tracking window size in terms of width and height. Using ground truth data, we compute the variation of width and height the target object and then compare it with the results obtained from our tracker with adaptive scaling. Fig. 19 shows the plots for this analysis and Table 5 summarizes the errors for this experiment. Overall, our tracker with adaptive scaling is able to adjust both the width and height of the tracking window when the target object undergoes large variation in scale.

5. Conclusion and future work

In this paper, we have introduced an algorithm for accurate tracking of objects undergoing significant shape variation (e.g., articulated objects). Under the general assumption that the foreground intensity distribution is approximately stationary, we show that it is possible to rapidly and efficiently estimate it amidst substantial shape changes using a collection of adaptively positioned rectangular blocks. The proposed algorithm first locates the target by scanning the entire image using the estimated foreground intensity distribution. The refinement step that follows provides an estimated target contour from which the blocks can be repositioned and weighted. The proposed algorithm is efficient and simple to implement. Experimental results have demonstrated that the proposed tracking algorithm consistently provides more precise tracking result when compared with integral histogram-based tracker [4] and mean-shift tracker [2].

We have identified several possible directions for future research. Foremost among them is the search for a more efficient algorithm for adjusting and repositioning the blocks. The current greedy algorithm we have is not optimal but it has the virtue of being easy to implement with good empirical results. Finally, online learning of shape and appearance variations will be a challenging research problem for the future.

Acknowledgments

M.-H. Yang is supported in part by a UC Merced faculty start-up fund and a gift from Google.

References

[1] S. Birchfield, Elliptical head tracking using intensity gradients and color histograms, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1998, pp. 232–237.

[2] D. Comaniciu, V. Ramesh, P. Meer, Real-time tracking of non-rigid objects using mean shift, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2000, pp. 2142–2147.

[3] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2001, pp. 511–518.

[4] F. Porikli, Integral histogram: a fast way to extract histograms in Cartesian spaces, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 829–836.

[5] A. Adam, E. Rivlin, I. Shimshoni, Robust fragments-based tracking using the integral histogram, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2006, pp. 2142–2147.

[6] Z. Fan, M. Yang, Y. Wu, G. Hua, Efficient optimal kernel placement for reliable visual tracking, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2006, pp. 658–665.

[7] V. Parameswaran, V. Ramesh, I. Zoghlami, Tunable kernels for tracking, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2006, pp. 2179–2186.

[8] S. Birchfield, S. Rangarajan, Spatiograms versus histograms for region-based tracking, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 1158–1163.

[9] S.M.S. Nejhum, J. Ho, M.-H. Yang, Visual tracking with histograms and articulating blocks, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2008.

[10] A. Yilmaz, O. Javed, M. Shah, Object tracking: a survey, ACM Computing Surveys 38 (4) (2006) 1–45.

[11] J. Aggarwal, Q. Cai, Human motion analysis: a review, Computer Vision and Image Understanding 73 (3) (1999) 428–440.

[12] D. Gavrila, The visual analysis of human movement: a survey, Computer Vision and Image Understanding 73 (1) (1999) 82–98.

[13] T. Moselund, E. Granum, A survey of computer vision-based human motion capture, Computer Vision and Image Understanding 81 (3) (2001) 231–268.

[14] T. Moselund, A. Hilton, V. Kruger, A survey of advances in vision-based human motion capture and analysis, Computer Vision and Image Understanding 104 (2) (2006) 90–126.

[15] J. Rehg, T. Kanade, Model-based tracking of self-occluding articulated objects, in: Proceedings of the International Conference on Computer Vision, 1995, pp. 612–617.

[16] M.J. Black, A.D. Jepson, Eigentracking: robust matching and tracking of articulated objects using a view-based representation, International Journal of Computer Vision 26 (1) (1998) 63–84.

[17] A. Blake, M. Isard, Active Contours, Springer, 2000.

[18] M. La Cascia, S. Sclaroff, V. Athitsos, Fast, reliable head tracking under varying illumination: an approach based on registration of texture-mapped 3D models, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (4) (2000) 322–336.

[19] S. Sclaroff, J. Isidoro, Active blobs, in: Proceedings of the International Conference on Computer Vision, 1998, pp. 1146–1153.

[20] M. Kass, A. Witkin, D. Terzopoulos, Snakes: active contour models, International Journal of Computer Vision 1 (4) (1988) 321–331.

[21] K. Toyama, A. Blake, Probabilistic tracking in a metric space, in: Proceedings of the International Conference on Computer Vision, 2001, pp. 50–59.

[22] T. Cootes, G. Edward, C. Taylor, Active appearance models, in: Proceedings of the European Conference on Computer Vision, 1998, pp. 484–498.

[23] V. Caselles, R. Kimmel, G. Sapiro, Geodesic active contours, International Journal of Computer Vision 22 (1) (1997) 61–79.

[24] D. Cremers, Nonlinear dynamical shape priors for level set segmentation, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2007.

[25] N. Paragios, R. Deriche, Geodesic active contours and level sets for the detection and tracking of moving objects, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (3) (2000) 266–280.

[26] D. Freedman, T. Zhang, Active contours for tracking distributions and shape prior, in: Proceedings of the International Conference on Computer Vision, 2003, pp. 1056–1062.

[27] R.T. Collins, Y. Liu, On-line selection of discriminative tracking features, in: Proceedings of the International Conference on Computer Vision, 2003, pp. 346–352.

[28] G. Hager, M. Dewan, C. Stewart, Multiple kernel tracking with SSD, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2004, pp. 790–797.

[29] R. Collins, Mean-shift blob tracking through scale space, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2003, pp. 234–240.

[30] H. Grabner, M. Grabner, H. Bischof, Real-time tracking via on-line boosting, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2006, pp. 260–267.

[31] P. Felzenszwalb, D. Huttenlocher, Pictorial structures for object recognition, International Journal of Computer Vision 61 (1) (2005) 55–79.

[32] D. Freedman, T. Zhang, Interactive graph cut based segmentation with shape priors, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 755–762.

[33] Y. Boykov, M.-P. Jolly, Interactive graph cuts for optimal boundary and region segmentation of objects in n–d images, in: Proceedings of the International Conference on Computer Vision, 2001, pp. 105–112.

[34] J. Ho, K.-C. Lee, M.-H. Yang, D. Kriegman, Visual tracking using learned subspaces, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2004, pp. 782–789.

[35] J. Lim, D.A. Ross, R.-S. Lin, M.-H. Yang, Incremental learning for visual tracking, in: Advances in Neural Information Processing Systems, 2004, pp. 793–800.

[36] A. Balan, M. Black, An adaptive appearance model approach for model-based articulated object tracking, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2006, pp. 758–765.

[37] D. Ramanan, C. Sminchisescu, Training deformable models for localization, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2006, pp. 206–213.