Visual Tracking with Histograms and Articulating Blocks

S M Shahed Nejhum
Department of CISE
University of Florida
Gainesville, FL 32611
smshahed@cise.ufl.edu

Jeffrey Ho
Department of CISE
University of Florida
Gainesville, FL 32611
jho@cise.ufl.edu

Ming-Hsuan Yang
Honda Research Institute
800 California Street
Mountain View, CA 94041
mhyang@ieee.org

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 3GHz machine. Experimental results have demonstrated that the algorithm is able to efficiently track articulated objects undergoing large variation in appearance and shape.

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 (Figure 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., [3, 7]). For rectangular shapes, efficient algorithms such as integral image [25]

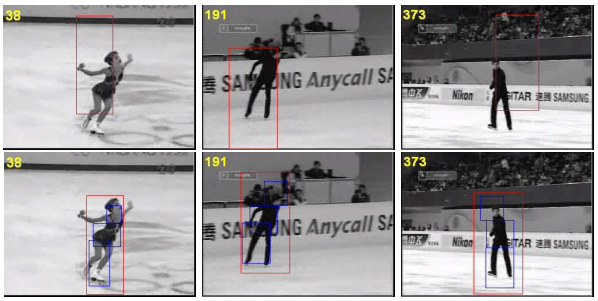

Figure 1. **Top:** Using only histogram for representing object's appearance, the tracking results are often unsatisfactory. **Bottom:** Using the proposed algorithm, the tracking results are much more consistent and satisfactory.

and integral histogram [20] have been successfully applied to object detection and tracking [1]. 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 [10, 19] 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 [7]. 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. Conse-

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

quently, the resulting histogram can be corrupted by background pixels, and the tracking result degrades accordingly (e.g., unstable or jittered results as shown in Figure 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 [3]. 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 [4] 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. In this paper, we propose a tracking algorithm that solves the above problems, and at the same time, it still has comparable running time as the tracking algorithm using (plain) integral histogram [20]. The proposed algorithm consists of global scanning, 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 extract the target contour 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. In this section, we discuss the most relevant works within the context of object tracking.

Active contours using parametric models [5, 17] 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 [24], 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 [8].

Instead of using contours to model shapes, kernel-based methods represent target's appearance with intensity, gradients, and color statistics [3, 7]. While these methods have demonstrated successes in tracking targets with shapes represented by ellipses, they are not effective for modeling non-rigid objects. Although methods using multiple kernels [10, 14] have been proposed to cope with this problem, it is not clear such methods are able to 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 [25]. However, fast algorithms for computing Haar-like features and histograms such as integral images [25] or integral histograms [20] require rectangular windows to model the target's shape. Consequently, it is not straightforward to apply these 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 e.g., [13]. 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.

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.

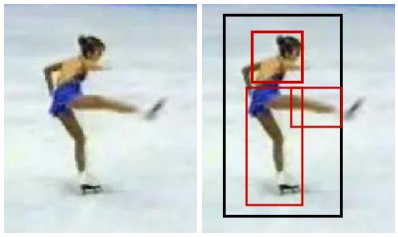

Figure 2. **Left:** An articulating target. **Right:** Given the contour of the target object, we select those blocks (and associated weights) with non-empty intersection with the interior region of the target defined by the contour. Blocks containing only background pixels are not selected. The importance of a block is proportional to the percentage of its pixels belonging to the foreground.

The high-level outline of the proposed algorithm is shown in Figure 3. It consists of three sequential steps: detection, 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, $\mathbf{B}_i$, $1 \leq i \leq K$, within the main tracking window $\mathbf{W}$ as shown in Figure 2. The positions of the blocks within the tracking window are adaptively adjusted throughout the tracking sequence, and they may have small overlaps to account for extreme shape variations. At each frame t, the tracker maintains the followings: 1) a template window $\mathbf{W}_t$ with a block configuration, 2) 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 computed from the blocks, and 3) 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 value when compared with the template window $\mathbf{W}$). 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 template 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 [9, 11], 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 tacker 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 similarity measure $\mathbf{S}$ with respect to the current template window $\mathbf{W}$,

$$\mathbf{W}^* = \max_{\mathbf{W}'} \mathbf{S}(\mathbf{W}', \mathbf{W}), \tag{1}$$

as $\mathbf{W}'$ ranging over all scanned windows. The similarity measure $\mathbf{S}$ is constructed from local foreground histograms computed from the blocks as follows. First, we transfer the block configuration on the template window 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

Tracking Algorithm Outline

1. Detection
The entire image is scanned and the window with the highest similarity value using (1) is determined to be the tracking window $\mathbf{W}^*$.
2. Refinement
Within $\mathbf{W}^*$, the target is segmented out using a graph-cut based segmentation algorithm that computes the target contour. The segmentation uses both estimated foreground and background distributions.
3. 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.

Figure 3. High-level outline of the proposed tracking algorithm.

$\mathbf{B}_i$ is the intersection of the raw histogram $\mathbf{H}_t^{\mathbf{B}_i}$ with the initial foreground histogram of the corresponding block:

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

where b indexes the bins. The similarity measure is defined as the weighted sum of the histogram similarities between $\mathbf{H}_0^{\mathbf{B}_i^f}(b)$ and $\mathbf{H}_t^{\mathbf{B}_i^f}(b)$,

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

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

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

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, $\mathbf{S}$ 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 similarity measure $\mathbf{S}$ defined in Equation 2 actually encodes some amount of shape and spatial information through the block configuration and their weights.

3.2. Refinement

Once the global scan produces a sub-window $\mathbf{W}^*$ in which the target is located, the next step is to extract an approximate boundary contour so that the shape variation can be better accounted for. We apply a graph-cut segmentation algorithm to segment out the the foreground target in $\mathbf{W}^*$. Previous work on this type of segmentation in the context of visual tracking (e.g., [9, 11, 12]) always define the cost function in the form

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

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 $\mathcal{P}$ denote the set of all pixels in $\mathbf{W}^*$. Let P_B denote the background density that we estimated in the previous frame, and $P_i, 1 \leq i \leq K$ the foreground density from $\mathbf{B}_i$ (by normalizing the histogram $\mathbf{H}_i^{\mathbf{B}_i^f}$). Furthermore, we will denote P_f the foreground density obtained by normalizing the current foreground histogram $\mathbf{H}_t^f$. Following [6, 12], the graph-cut algorithm will minimize the cost function

$$E(C_p) = \mu \sum_{p \in \mathcal{P}} R_p(C_p) + \sum_{(p,q) \in \mathcal{N}: C_p \neq C_q} B_{p,q}, \qquad (3)$$

where $C_p : \mathcal{P} \rightarrow \{0, 1\}$ is a binary assignment function on $\mathcal{P}$ such that for a given pixel p, $C(p) = 1$ if p is a foreground pixel and 0 otherwise[1]. μ is a weighting factor and $\mathcal{N}$ denotes the set of neighboring pixels. 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

$$\begin{aligned} R_p(C_p = 0) &= -\log P_F(I(p), p) \\ R_p(C_p = 1) &= -\log P_B(I(p)), \end{aligned}$$

where $P_F(I(p), p) = P_i(I(p))$ if $p \in \mathbf{B}_i$, and $P_F(I(p), p) = P_f(I(p))$ if p is not contained in any $\mathbf{B}_i$. 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 [6]. 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. Figure 4 presents one segmentation result where the extracted target contour is shown in green.

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

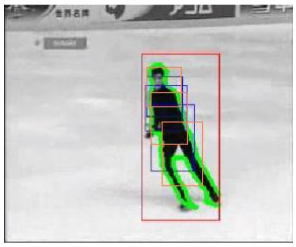

Figure 4. **Blue:** The block configuration from the previous frame. **Green:** The contour is estimated using a fast graph-cut algorithm. **Red:** The blocks are repositioned using a greedy strategy to provide a maximal coverage of the target.

3.3. 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. In addition, these blocks are repositioned in a controlled elastic manner to account for large articulated motions. 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. Figure 4 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. Furthermore, the percentage η_i of pixels in $\mathbf{B}_i$ that are foreground pixels can also be computed quickly. Finally, to maintain the stationary total foreground density, we solve a small-scale non-negative least squares problem

$$\mathbf{H}_0^f = \sum_{i=1}^{K} \alpha_i \mathbf{H}_t^{\mathbf{B}_i^f}$$

with $\alpha_i \geq 0$, and $\mathbf{H}_0^f$ the foreground density of the initial frame. The weights λ_i are updated according to $\lambda_i = \eta_i \alpha_i$ followed by a normalization.

3.4. Discussion

Comparing with the recent work that employ discriminative models (classifiers) for tracking (e.g., [13]), our approach is mainly generative through the use of intensity histograms. While we assume that the intensity distribution stays stationary, the features we constantly update are the block configurations and the associated weights. Online appearance updates (e.g, [13, 15, 18]) 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 [5, 9, 11, 24]. 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., [2, 21]). 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 are available at `http://www.cise.ufl.edu/~smshahed/tracking.htm`. In this implementation, we use intensity histograms with 16 bins for grayscale videos, and for color sequences, each color channel is binned separately with 16 bins. The number of blocks, K, is set to two or three. The tracker has been tested on a variety of video sequences, and five of the most representative sequences are reported in this paper. In particular, we show the comparisons between our tracker and that of [20] using only integral histograms[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. 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 runtime complexity allows our tracker to consistently produce much more stable and satisfactory tracking results.

In the following experiments, for initialization, we manually locate the tracking window in the first frame, and for the experiments, both trackers start with the same tracking window. The five sequences shown below are all collected from the web. In these sequences, the foreground targets undergo significant appearance changes, which mainly due to shape variation. Although the intensity profiles stay more or less constant, the drastic shape variations make

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

these sequences difficult for trackers that use only intensity histograms. The first skating sequence contains over 150 frames, and the dazzling performance is accompanied by an equally dazzling pose variation. As shown in Figure 5, while the background is relatively simple, the integral histogram tracker is 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 figure, one major reason for this improvement is that the spatial locations of the blocks are updated correctly by our algorithm as the skater changes her pose. The second sequence contains 430 frames with a figure skater performing in a cluttered environment. 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 Figure 6 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. This experiment also demonstrates the advantage of having the capability to efficiently scan the entire image for the target.

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 (Figures 7, 8), our tracker is able to track the dancers accurately while the integral histogram tracker again fails to produce consistent and accurate results. Figure 9 presents the tracking result for a color video sequence in which there are significant variations in shape, scale, and appearance of the target. Notwithstanding these difficulties, our tracker is able to follow the target accurately using only two blocks. More tracking results can be found in the supplemental material.

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 [20].

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. In addition, mechanisms for handling occlusion along the line of [1] will further improve the tracker's robustness. Finally, online learning of shape and appearance variations will be a challenging research problem for the future.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 2142–2147, 2006.

[2] A. Balan and M. Black. An adaptive appearance model approach for model-based articulated object tracking. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, volume 2, pages 758–765, 2006.

[3] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 232–237, 1998.

[4] S. Birchfield and S. Rangarajan. Spatiograms versus histograms for region-based tracking. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, volume 2, pages 1158–1163, 2005.

[5] A. Blake and M. Isard. *Active Contours*. Springer, 2000.

[6] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In *Proc. Int. Conf. on Computer Vision*, volume 1, pages 105–112, 2001.

[7] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 2142–2147, 2000.

[8] T. Cootes, G. Edward, and C. Taylor. Active apperance models. In *Proc. European Conf. on Computer Vision*, volume 2, pages 484–498, 1998.

[9] D. Cremers. Nonlinear dynamical shape priors for level set segmentation. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, page Unknown, 2007.

[10] Z. Fan, M. Yang, Y. Wu, and G. Hua. Efficient optimal kernel placement for reliable visual tracking. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, volume 1, pages 658–665, 2006.

[11] D. Freedman and T. Zhang. Active contours for tracking distributions and shape prior. In *Proc. Int. Conf. on Computer Vision*, pages 1056–1062, 2003.

[12] D. Freedman and T. Zhang. Interactive graph cut based segmentation with shape priors. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 755–762, 2005.

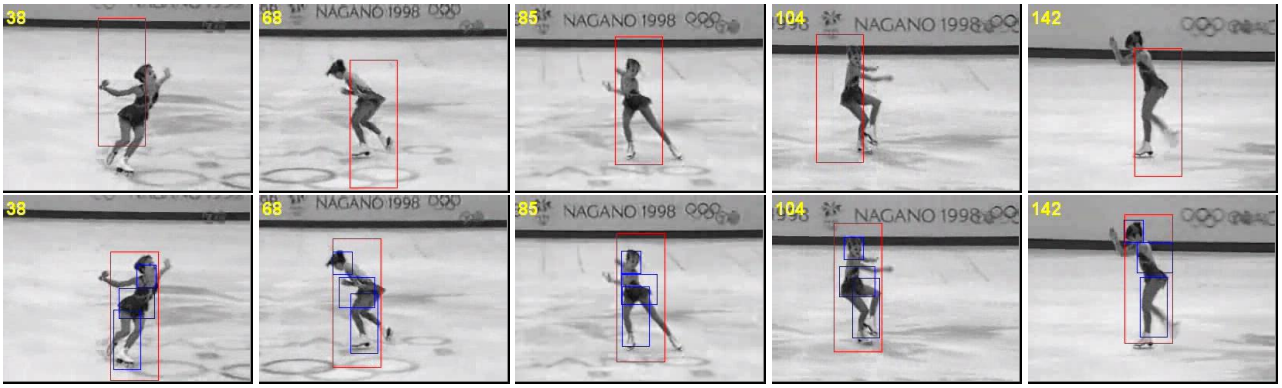

Figure 5. **Top:** Tracking results using only integral histogram. **Bottom:** Tracking results using the proposed algorithm. The shape variation in this sequence is substantial. Notice the unsatisfactory result by the integral histogram tracker. The inaccurate tracking windows it produces is difficult to be utilized by other applications.

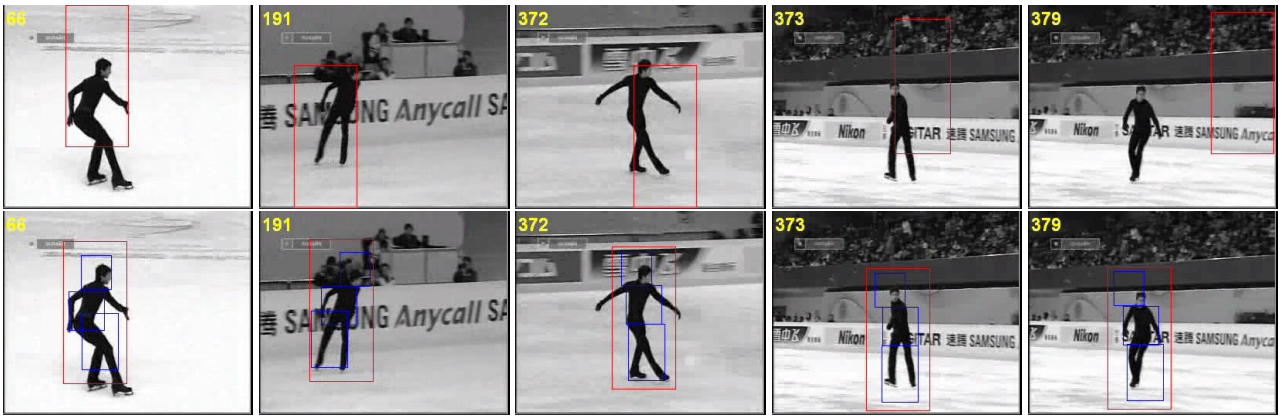

Figure 6. **Top:** Tracking results using only integral histogram. **Bottom:** Tracking results using the proposed algorithm. Note that our algorithm is able to track the skater across shots (e.g., frame 372 to 373 and onward).

[13] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 260–267, 2006.

[14] G. Hager, M. Dewan, and C. Stewart. Multiple kernel tracking with ssd. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, volume 1, pages 790–797, 2004.

[15] J. Ho, K.-C. Lee, M.-H. Yang, and D. Kriegman. Visual tracking using learned subspaces. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, pages 782–789, 2004.

[16] S. Jehan-Besson, M. Barlaud, G. Aubert, and O. Faugeras. Shape gradients for histogram segmentation using active contours. In *Proc. Int. Conf. on Computer Vision*, pages 408–415, 2003.

[17] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. J. Computer Vision*, 1(4):321–331, 1988.

[18] J. Lim, D. A. Ross, R.-S. Lin, and M.-H. Yang. Incremental learning for visual tracking. In *Advances in Neural Information Processing Systems*, pages 793–800, 2004.

[19] V. Parameswaran, V. Ramesh, and I. Zoghlami. Tunable kernels for tracking. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, volume 2, pages 2179–2186, 2006.

[20] F. Porikli. Integral histogram: A fast way to extract histograms in Cartesian spaces. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, volume 1, pages 829–836, 2005.

[21] D. Ramanan and C. Sminchisescu. Training deformable models for localization. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, volume 1, pages 206–213, 2006.

[22] Y. Shi and W. Karl. Real-time tracking using level sets. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, volume 2, pages 34–41, 2005.

[23] M. Swain and D. Ballard. Color indexing. *Int. J. Computer Vision*, 7(1):11–32, 1991.

[24] K. Toyama and A. Blake. Probabilistic tracking in a metric space. In *Proc. Int. Conf. on Computer Vision*, pages 50–59, 2001.

[25] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. IEEE Conf. on Comp. Vision and Patt. Recog.*, volume 1, pages 511–518, 2001.

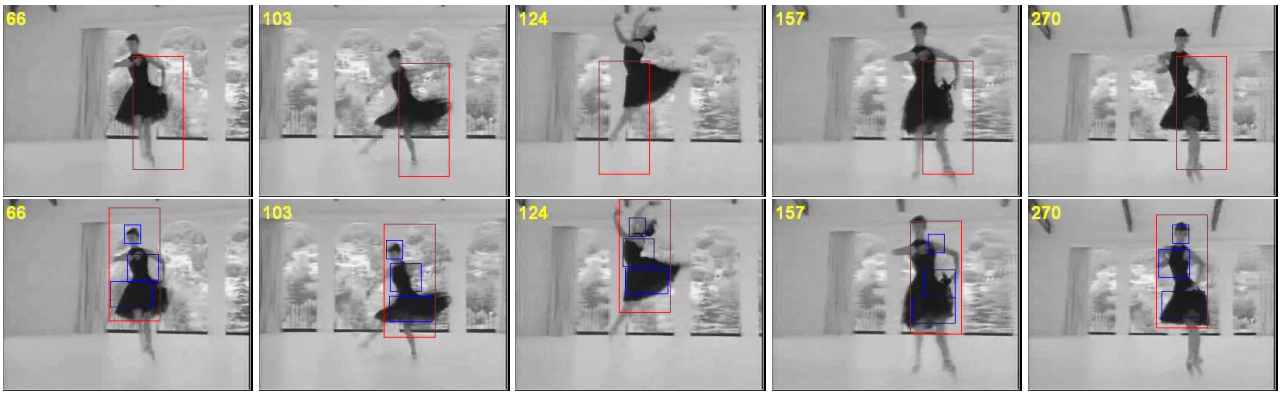

Figure 7. **Top:** Tracking results using only integral histogram. **Bottom:** Tracking results using our algorithm. Note the target undergoes large shape and scale variation.

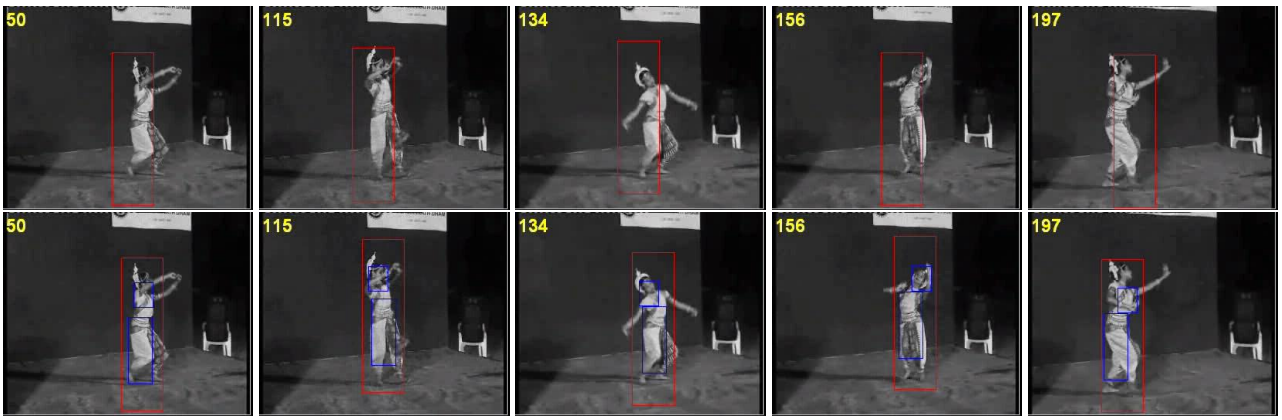

Figure 8. **Top:** Tracking results using only integral histogram. **Bottom:** Tracking results using the proposed algorithm. Note the target undergoes significant shape and appearance variation.

Figure 9. Tracking results using the proposed algorithm. Shape and appearance variations are substantial in this sequence.