Video Event Detection: From Subvolume Localization to Spatiotemporal Path Search

Du Tran, *Student Member*, *IEEE*, Junsong Yuan, *Member*, *IEEE*, and David Forsyth, *Fellow*, *IEEE*

Abstract—Although sliding window-based approaches have been quite successful in detecting objects in images, it is not a trivial problem to extend them to detecting events in videos. We propose to search for spatiotemporal paths for video event detection. This new formulation can accurately detect and locate video events in cluttered and crowded scenes, and is robust to camera motions. It can also well handle the scale, shape, and intraclass variations of the event. Compared to event detection using spatiotemporal sliding windows, the spatiotemporal paths correspond to the event trajectories in the video space, thus can better handle events composed by moving objects. We prove that the proposed search algorithm can achieve the global optimal solution with the lowest complexity. Experiments are conducted on realistic video data sets with different event detection tasks, such as anomaly event detection, walking person detection, and running detection. Our proposed method is compatible with different types of video features or object detectors and robust to false and missed local detections. It significantly improves the overall detection and localization accuracy over the state-of-the-art methods.

Index Terms—Event detection, action detection, multiple event detection, max-path search, dynamic programming

1 Introduction

Video event analysis is a challenging problem because video events can vary significantly, for example, from particular human actions (e.g., running, kissing) to object-related events (e.g., kicking ball, riding horse), or even an abnormal event (e.g., cyclist runs into pedestrian walkway). Traditional approaches for video event detection require detecting and tracking objects first, for example, people [1], then recognize what is happening around those tracked objects. However, as tracking and detection are challenging tasks in dynamic and crowded scenes, they may not provide robust event detection results.

Encouraged by the success of object detection, the sliding window approach has been introduced to video event detection [2]. Such a sliding window-based approach does not require to detect and track the objects, while it can locate the whole video event by finding a spatiotemporal video subvolume [3], [4]. To avoid the exhaustive search of the target video subvolume, i.e., a spatiotemporal bounding box, inspired by the branch-and-bound search in object detection [5], [6], spatiotemporal branch-and-bound search has been proposed to detect human actions in videos [7]. Using this approach, one needs to evaluate discriminative scores at local patches, then the detection step is carried out by searching for a video subvolume [7], [8] with the maximum total score. Although subvolume search has its own advantages for video event detection, it still confronts two unsolved problems.

First, most of the current spatiotemporal sliding window search methods only support detecting windows of constrained structure, i.e., the three-dimensional (3D) bounding box. Unfortunately, unlike object detection where a bounding box works reasonably well in many applications [6], [9], the 3D bounding box is quite limiting for video pattern detection. In fact, this assumption works for "static" events (e.g., kiss or handshake), but is inapplicable for "dynamic" events (e.g., cycling or walking). We define an event as static if actor or object does not move when the event occur (e.g., handshake is static while walking is dynamic). To illustrate this, Fig. 1a shows a cycling event. The cyclist starts at the left side of the screen and rides to the right side of the screen. To detect this event, because of the bounding box constraint, one can only locate the whole event using a large video subvolume, which covers not only the cycling event, but also a significantly large portion of the backgrounds (Fig. 1a). In such a case, the detection score of the video event is negatively affected by the cluttered and dynamic backgrounds. Instead of providing a global bounding box that covers the whole event, more often than not it is preferable to provide an accurate spatial location of the video event and track it in each frame. As a result, a more accurate spatiotemporal localization is desirable to detect the video event, as shown in Fig. 1b.

Moreover, as the video space is much larger than the image space, it becomes very time consuming to search 3D sliding windows. For example, given a video sequence of size $w \times h \times n$, where $w \times h$ is the spatial size and n is its length, the total number of 3D bounding boxes is of $O(w^2h^2n^2)$, which is much larger compared to the image space of only $O(w^2h^2)$2D boxes. Although some recent methods have been proposed to handle the large video space [8], the worst case complexity is still of $O(w^2h^2n)$. In general, it is challenging to

- *D. Tran and J. Yuan are with the School of Electrical and Electronics Engineering, Nanyang Technological University, S1-B1B-41, Singapore 639798. E-mail: {dutran, jsyuan}@ntu.edu.sg.*
- *D. Forsyth is with the Department of Computer Science, University of Illinois at Urbana-Champaign, 3310 Siebel Hall, Urbana, IL 61801. E-mail: daf@illinois.edu.*

Manuscript received 11 Oct. 2011; revised 10 July 2012; accepted 18 June 2013; published online 23 July 2013.
Recommended for acceptance by B. Schiele.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2011-10-0729.
Digital Object Identifier no. 10.1109/TPAMI.2013.137.

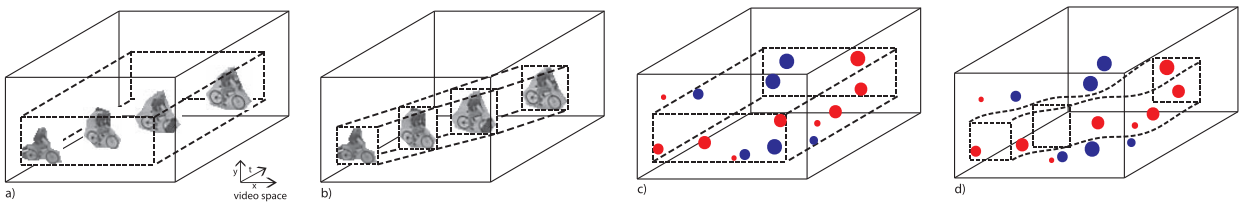

Fig. 1. *Detection of the cycling event* a) Event localization by three-dimensional bounding box. b) More accurate spatiotemporal localization of the event. c) Subvolume-based event detection methods search for video subvolume that maximizes the total score. Local scores are visualized as red and blue dots. Red dots indicate high likelihood of the event while blues are of low likelihood. d) Spatiotemporal-based event detection methods find the optimal spatiotemporal path having the highest score.

search videos of high spatial resolutions. Even worse, if we relax the bounding box constraint of the sliding windows, the number of candidates will further increase. Thus, a more efficient search method is required.

To address the above problems, we propose a novel spatiotemporal localization method that relaxes the 3D bounding box constraint and formulates the video event detection as a spatiotemporal path discovery problem. Figs. 1c and 1d demonstrate this relaxation. Instead of searching for a subvolume, we search for a spatiotemporal path that maximizes the summation score. This relaxation is more applicable and accurate for dynamic events when the object moves across different frames. Suppose a discriminative classifier can assign a local detection score to every 2D sliding window in each frame. To fuse these local evidences and connect them to establish a spatiotemporal path, we build a spatiotemporal trellis that presents all of smooth spatiotemporal paths, where a target event will correspond to one of them. By finding the optimal path in the trellis with the highest detection score, our formulation is a generalization of the 3D bounding box search in [8]: We do not reinforce the fixed spatial location of the video event, but track the event as it moves across multiple frames. Because the discovered path precisely contains the video event, it minimizes the affection of the cluttered and dynamic backgrounds.

Although the search space of our new formulation is much larger than that of searching 3D bounding boxes, we propose an efficient search method that can obtain the global optimal solution with proven lowest search complexity, which is only linear to the video volume size, i.e., $O(whn)$. Experiments on abnormal video event detection, walking pedestrian detection, and running detection validate the following advantages of our new formulation of video event detection:

1. By discovering the optimal spatiotemporal path, our method determines the start and the end of the video event automatically, and can precisely localize the event in each video frame. It is robust to the false and missed local detections and thus can effectively handle heavy occlusions.
2. As both positive and negative training examples are utilized for a discriminative training, our method is robust to intraclass variations of the video events and the cluttered and dynamic backgrounds.
3. Our proposed method can be easily extended to handle scale and shape variations of the event, and can detect multiple events simultaneously.

2 Related Work

2.1 Video Event Detection

Video event detection is an important topic in computer vision, with extensive applications in video surveillance, content-based video search, multimedia retrieval, and so on. The latter two have seen increasing demands due to the exploding number of Internet videos (e.g., YouTube). At the same time, the problem becomes more challenging when dealing with realistic videos because of the diversity of the video events, complex background motions, scale changes, and occlusions, not to mention the high-dimensional search space inherent to videos.

One traditional approach for event detection is to track the actors, stabilize these figures, and then recognize them [1], [10], [11]. Such methods highly rely on the quality of the tracking results, hence suffer from unreliable trackers. This limitation motivates methods that handle detection and recognition simultaneously, normally accomplished by spatiotemporal video volume matching, including action-MACH [3], volumetric features [2], boosted space-time features [12], segment-based features [13], space-time orientation [4], template matching [14], [15], adaptive regression kernels [16], and so on. To localize events, these methods have to apply the sliding subvolume scheme, which is time consuming. Rather than sliding subvolume, Boiman and Irani [17] proposed ensembles of patches to detect irregularities in images and videos. Hu et al. [18] used multiple-instance learning to localize the best video subvolume. Gall et al. [19] proposed Hough Forest for object detection, tracking, and action recognition. Yu et al. [20], [21] combined top-k subvolume search and random forest for action detection. Gaidon et al. [22] introduced Actom sequence to detect action in videos. Cao et al. [23] employed model adaptation to do cross-data set action detection. Recently, with the success of branch-and-bound subwindow search [6], Yuan et al. [8] extended this method to subvolume search that can locate video patterns effectively. However, these approaches are still constrained by the 3D subvolume. Finally, Zhang et al. [24] relaxed the subwindow rectangle constraint to free-shape subwindow search based on contour refinement. This approach works well for object localization but is still difficult to extend to video search due to its complexity.

2.2 Object Tracking

Our event detection problem is quite different compared to online object tracking. More specifically, we aim at detecting a particular class of event (e.g., running) while not tracking or following an object/person. Initialization of the target

object is not required in our method, but is necessary for online tracking. Discriminative training in our case is to learn models for a class of event, while online tracking is to build a model for the target object at running time.

Our approach shares some properties with offline tracking methods [25], [26], [27], [28], [29], [30], [31], [32]. These methods had shown their effectiveness compared to traditional tracking methods thanks to the success of object detectors. Similar to offline tracking, our approach considers joining detection outputs to maximize the discriminative scores while keeping the smoothness of the movement trajectories. In contrast, our proposed method is not limited to object detectors, but can be also applied to more general discriminative confidence maps of event, action, motion, or keypoint detectors. This flexibility makes it more general, and thus applicable to a broader range of video event detection problems. In practice, our proposed method has been used as a general algorithm for efficient inference and complex learning problems for spatiotemporal event detection problem [33] while it is hard to adapt current offline tracking methods to solve general inference and learning problems.

To our best knowledge, our *Maximum Path* algorithm is novel to video event detection and proven to be globally optimal with the lowest computational complexity. Interestingly, this problem has not been discussed in discrete algorithm literature although the *Maximum Subarray* problem had been raised and solved long time ago [34].

3 Problem Formulation

We denote a video sequence as $\mathcal{S} = \{I_1, I_2, \ldots, I_n\}$, where I_k is a $w \times h$ image frame. Treating the video as a spatiotemporal data volume, for each spatiotemporal location $v = (x, y, t)$, we denote by $W(v)$ the local window or subvolume centered at v. Without loss of generality, we suppose all of the windows are of a fixed scale, we further denote by $M(W(v))$, or $M(v)$ for short, the discriminative score of the local window centered at v. A high positive score of $M(v)$ implies a high likelihood that the event occurs at the local position v, while a negative score indicates a low likelihood of the occurrence. There are many ways to obtain the score $M(v)$ using different types of features. For example, one can use the 2D window [35], [36] to slide over the video sequence and get the local scores of each window. Alternatively, individual spatiotemporal interest points (STIPs) [37], [38] can vote for the video event [8], then the score of a local window is the summation of the interest point scores. To handle the events with local structures, instead of using a window-based detector, one also can employ a volume-based detector (e.g., $64 \times 128 \times 4$ volume detector) or a volume-based filter (e.g., ActionMACH [3], or "Action Spotting" [4]), which can handle events with short-time structures and thus can further improves the method's robustness. With the assumption that events can be detected by local detectors (e.g., subwindow or subvolume), our method only applies to periodic events with local structures. Long and complex structured events (e.g., a car changing lane) may require long-time observations and more sophisticated analysis techniques.

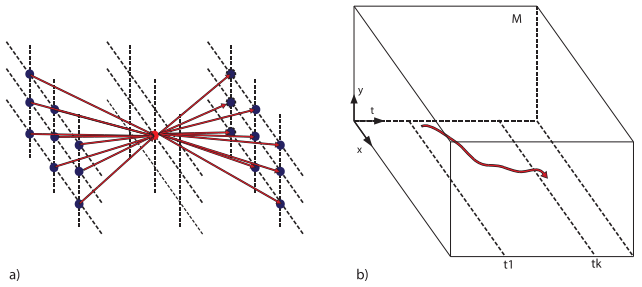

Fig. 2. *Maximum Path problem* a) Nine incoming and nine outgoing neighbors for a node in $\mathcal{G}_M$. b) The visualization of one path. Searching for the maximum path in spatiotemporal space is difficult due to an exponential number of possible paths with arbitrary lengths.

By treating each window $W(v)$ as a node, we obtain a three-dimensional trellis to represent all $W(v)$ in the video. A 3D trellis can be seen as a 3D array with elements are local discriminative scores $M(v)$. Fig. 2a is a concrete example of a 3D trellis with a size of $3 \times 3 \times 3$. Given a 3D trellis $\mathcal{G}_M$ with a size of $w \times h \times n$, $p = \{v_i\}_{i=i_1}^{i_k}$ is a *path* in $\mathcal{G}_M$ if it satisfies 1) the *path connectivity constraints*: $x_i - 1 \leq x_{i+1} \leq x_i + 1, \quad y_i - 1 \leq y_{i+1} \leq y_i + 1, \quad t_{i+1} = t_i + 1$ and 2) the *boundary constraints*: $1 \leq x_i \leq w$, $1 \leq y_i \leq h$, and $1 \leq t_{i_1} \leq t_{i_k} \leq n$. The first constraint set shows that each node $v = (x, y, t)$ has nine incoming and nine outgoing neighbors as showed in Fig. 2a. The second constraint set indicates that the path can start and end at any position in the 3D array as long as the ending point occurs later than the starting point. Let $p = \{v_i\}_{i=i_1}^{i_k}$ be a path in $\mathcal{G}_M$, to evaluate its likelihood we compute the accumulated score of the path p in

$$M(p) = \sum_{i=i_1}^{i_k} M(v_i). \tag{1}$$

As each video event is characterized by a smooth spatiotemporal path in the 3D trellis, to detect the video event, the problem becomes to find the optimal path p^* with the highest accumulated score:

$$p^* = \arg\max_{p \in path(\mathcal{G})} M(p). \tag{2}$$

Solving the *Maximum Path* problem is difficult (Fig. 2b) because of the large search space: We do not know the start location (x_s, y_s, t_s) or the end location (x_e, y_e, t_e) of the event, as well as all of the intermediate states. The search space of all possible paths is exponential: $O(whnk^n)$, where whn is the size of the video volume, k is the number of incoming edges per node (the formal proof is provided in the supplemental materials. Thus, exhaustive search is infeasible. Although the maximum path problem can be addressed by the traditional shortest path search algorithm, for example, the Floyd-Warshall algorithm to find the shortest paths between all pairs of vertices, the search complexity is still quite high. The complexity of the Floyd-Warshall algorithm is to the cube of the number of vertices $O(|V|^3)$. Thus, it becomes $O(w^3h^3n^3)$ to solve (2), which is very time consuming for a large video volume. Other related work includes the *Maximum Subarray* problem which was posed by Ulf Grenander in 1977 and the 1D

case was solved by Jon [34]. Although it works perfectly for the 1D trellis [39], the problem is more complicated with higher dimension, for example, for our 3D trellis. Although the branch-and-bound search has proven to be efficient in searching 2D and 3D bounding boxes [6], [8], they cannot be applied to more flexible structures. To propose an efficient search that can find the global solution in (2), we first present an approach based on dynamic programming, followed by our proposed search method with proven lowest complexity.

4 Optimal Path Discovery

4.1 Efficient Max-Path Search via Dynamic Programming

Before addressing the Max-Path discovery problem, we first study a simplified version of the problem. We assume that the best path starts somewhere in the first frame and ends at the last frame. The following dynamic programming algorithm will find the best path.

Let $S_{i,j,t}$ be the maximum accumulated score of the best path starting somewhere from the first frame and leading to (i,j,t). For short, we denote $u=(i,j)$ and $v=(x,y)$ as 2D indices (e.g., $S_{i,j,t}=S_{u,t}$). We note that these notions are slightly different from the previous section, where v is a 3D index. And $N(u)$ is the set of neighbors of u in the previous frame. Equation (3) gives a solution for the *Max-Path* search problem:

$$S_{u,t}=\begin{cases} M_{u,t}, & t=1,\\ \max\limits_{v\in N(u)}\{S_{v,t-1}+M_{u,t}\}, & t>1. \end{cases} \tag{3}$$

This dynamic programming can be completed in $O(whn)$ to compute S, another $O(n)$ to trace backward to identify the best path, and uses $O(whn)$ memory space.

However, to automatically determine the starting and ending locations of the paths, we need to try different combinations and perform the dynamic programming many times. To improve this, let $S_{u,t,s}$ be the accumulated scores of the best path starting from the sth frame to the end location (u,t). S can be computed by

$$S_{u,t,s}=\begin{cases} -\infty, & s>t,\\ M_{u,t}, & s=t,\\ \max\limits_{v\in N(u)}\{S_{v,t-1,s}+M_{u,t}\}, & s<t. \end{cases} \tag{4}$$

Different from the previous dynamic programming in computing the matrix S, this new algorithm stores all possible solutions from all starting frames. When S is computed, the algorithm traces back for the best solution with all possible starting and ending points. This extension makes the complexity of the extended-algorithm $O(whn^2)$ to construct S and another $O(n)$ to search the best path, and needs $O(whn^2)$ memory. Taking advantage of the trellis structure, the search complexity now is reduced to linear to the volume size times the length of the video. Based on this result, we will show how to further improve the search to reach the lowest complexity in the next section.

4.2 Our Proposed Max-Path Discovery

We now propose a new algorithm with message propagation mechanism for the Max-Path discovery problem, with the complexity of only $O(whn)$. The algorithm consists of two steps: *message forwarding* and *path back-tracing*. Algorithm 1 shows the message forwarding process. Following the notations, let $M(x,y,t)$ be the output predictions of the video. The message propagation starts at $t=1$, then propagates the information from the current frame to the next. Each node needs to store a message value $S(x,y,t)$, which is the maximum accumulated score of the best possible path up to (x,y,t). $P(x,y,t)$ is the previous node that leads to (x,y,t) in the best possible path. These values can be computed by collecting information from each node's neighbors and its local value $M(x,y,t)$. When the message reaches a node, the algorithm looks for the best value S of its neighbors from the previous frame. If this value is positive, then the path continues to grow from existing best path and stores the accumulated score and the previous position. Otherwise, the algorithm starts a new path from the current position. Fig. 3 illustrates a concrete example of the algorithm.

Algorithm 1: Message propagation algorithm.

Input: $M(u,t)$: the local discriminative scores;
Output: $S(u,t)$: the accumulated scores of the best path leads to (u,t);
$P(u,t)$: the best path record for tracing back;
S^* : the accumulated score of the best path;
l^* : the ending location of the best path;
begin
  $S(u,1)=M(u,1),\forall u$;
  $P(u,t)=null,\ \forall(u,t)$;
  $S^*=-\infty$;
  $l^*=$ null;
  for $i\leftarrow 2$ **to** n **do**
    foreach $u\in[1..w]\times[1..h]$ **do**
      $v_0\leftarrow \text{argmax}_{v\in N(u)}\, S(v,i-1)$;
      if $S(v_0,i-1)>0$ **then**
        $S(u,i)\leftarrow S(v_0,i-1)+M(u,i)$;
        $P(u,i)\leftarrow(v_0,i-1)$;
      else
        $S(u,i)\leftarrow M(u,i)$;
      end
      if $S(u,i)>S^*$ **then**
        $S^*\leftarrow S(u,i)$;
        $l^*\leftarrow(u,i)$;
      end
    end
  end
end

Lemma 1. *$S(x,y,t)$ resulted from Algorithm 1 is the accumulated sum of the best path that leads to (x,y,t).*

Proof. Let us define $\mathcal{Q}(t)\triangleq$ "*$S(x,y,t)$ as the maximum accumulated sum of the best path leading to (x,y,t)*". We will prove that $\mathcal{Q}(t)$ is true $\forall t\in[1\ldots n]$ by induction. We initialize $S(x,y,1)=M(x,y,1),\forall(x,y)$, hence $\mathcal{Q}(1)$ is true. Assume that $\mathcal{Q}(k-1)$ is true, we now show $\mathcal{Q}(k)$ is

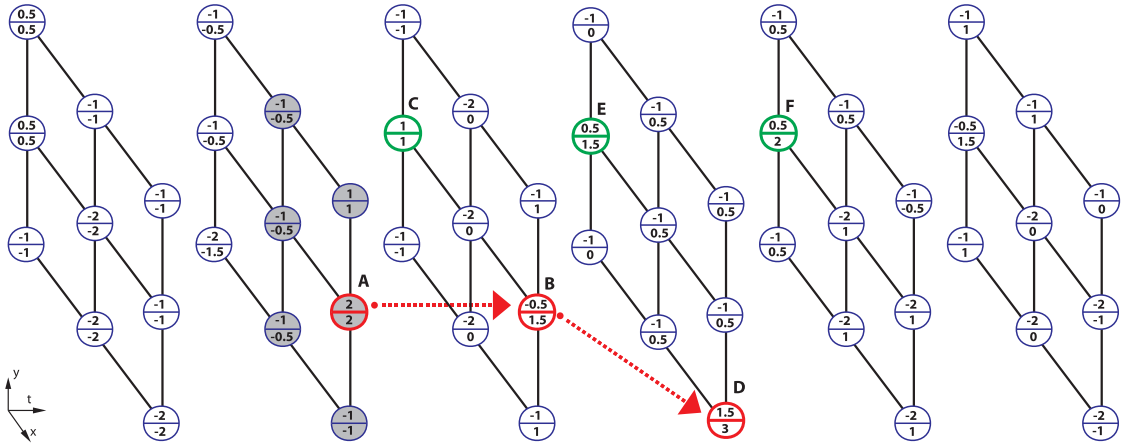

Fig. 3. *A message propagation example*: An example of Max-Path algorithm applied to a $3 \times 3 \times 6$ video. Each node is denoted with a local discriminative score (upper number), and the best accumulated score (lower number). In the first frame, all the best accumulated scores are initialized by their corresponding local discriminative scores. In the third frame, B can grow further from A, which has the best accumulated score among B's neighbors (shaded nodes), while C needs to start a new path. The final best path is A-B-D (red nodes), and C-E-F is the second best path (green nodes).

also true. If a node u at frame k has m directly connected neighbors, then there are $m+1$ possible paths leading to it. These paths include m paths going through its neighbors with an accumulated scores of $S(v,k-1)+M(u,k), v \in N(u)$ and another one starting by itself with a score of $M(u,k)$. From Algorithm 1, we have

$$v_0 = \arg\max_{v \in N(u)} S(v,k-1) \tag{5}$$

$$\Rightarrow S(v_0,k-1) \geq S(v,k-1), \forall v \in N(u) \tag{6}$$

$$\Rightarrow S(v_0,k-1)+M(u,k) \geq S(v,k-1) + M(u,k), \forall v \in N(u). \tag{7}$$

And also from Algorithm 1, the *If* statement for assigning values to S indicates two cases that

$$S(u,k) = \begin{cases} S(v_0,k-1)+M(u,k), & S(v_0,k-1) > 0 \\ M(u,k), & \text{otherwise}, \end{cases} \tag{8}$$

$$\Rightarrow S(u,k) = \max\{S(v_0,k-1)+M(u,k), M(u,k)\}. \tag{9}$$

From (7) and (9), we have shown that $S(u,k)$ is always the best accumulated sum compared to all $m+1$ paths that can lead to u. This confirms that $\mathcal{Q}(k)$ is true. □

Lemma 1 confirms the correctness of Algorithm 1. Algorithm 1 will result in the best path value S^* and the ending point of the best path l^*. The localization of the best path is straightforward by looking at the values stored in P and tracing back until reaching a *null* node. Overall, it takes $O(whn)$ to compute S, $O(n)$ to trace back the path, and uses $O(whn)$ memory to store S and P. The algorithm gives exactly the same results as the dynamic programming algorithm but reduces both computational and storage requirement.

As the size of the trellis is $w \times h \times n$, and one cannot find the maximum path sum without reading every element, $O(whn)$ is the lowest complexity we can expect. Together with *Lemma 1*, we thus have the following theorem.

Theorem 1. *Algorithm 1 results in the global optimal solution with a complexity of $O(whn)$, which is the lowest complexity of the Max-Path problem.*

4.3 Further Extensions of the Algorithm

4.3.1 Handling Multiple Scales and Shapes

When the events appear across a wide range of scales, we can extend the sliding window scheme to multiple scales. Instead of sliding a fixed scale window, at the same location v, one can use windows $W(v)$ with different scales. As a result, since each node is coupled with multiple windows with different scales, the trellis $\mathcal{G}_M$ becomes a 4D array with a size of $w \times h \times m \times n$ (m is the number of scales). The problem is now posed in 4D, but still can be solved by the same Algorithm 1. One difference is that the trellis is changed because each node now has neighbors not only from its same scale but also from its two nearest scales. More specifically, if a node has up to nine neighbors for the single scale setting, it now may have 27 neighbors including 9 from its same scale and two other 9s from its two adjacent scales. In general, the algorithm's complexity and space cost will both be increased to $O(whmn)$.

We previously assumed that the object does not change its shape during the event. For clarification, we first define the object shape ratio. We define the shape ratio as the ratio between the width and the height of the main object performing the event. For example, for a human actor, we normally use the ratio of 1:2, for the side car object this ratio is 2:1, for human face object is about 1:1. With this observation, we argue that there are events that the shape ratio may change and cannot fix. For example, in the dancing scenario, the actor performs a spinning action. During the spinning event, the shape ratio can be varying significantly. More specific, when the actor's two arms are parallel to the floor like a T shape, this ratio is about 1:1, but when those two arms are both orthogonal to the ground, this ratio is 1:2. Similarly to scales, we can extend the trellis to another dimension of shape, or both shape and scale. As a result, the trellis becomes 4D if using multiple shapes, and even 5D if we use both multiple scales and shapes. The same algorithm is still applicable; however, the complexity

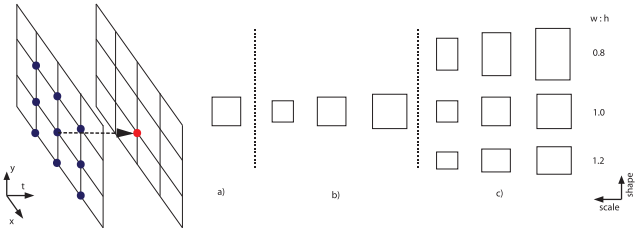

Fig. 4. *Possible extensions of Max-Path Search*: a) Single scale and single shape ratio results in a 3D trellis. b) Multiple scales with a fixed shape ratio result in a 4D trellis. c) Multiple scales and shape ratios will result in a 5D trellis.

is increased to $O(whmln)$, where m and l are the number of quantized scales and shape ratios. The simple visualization of those extensions is showed in Fig. 4.

4.3.2 Discovery of Multiple Paths

Similar to nonmaximum suppression or branch-and-bound [6], [8], this algorithm can also be applied repeatedly to locate multiple instances. After obtaining p^*, one can remove it from M and restart the process to search for the next best Max-Path.

4.3.3 Moving Speed Adaptation

The algorithm is also adaptable to the event moving speed. Instead of using 3×3 local neighbors, one can use a larger neighborhood region (e.g., 5×5 local neighbors) to accommodate fast motions of the event, or 1×1 for static events (e.g., hand-waving). Edges of neighbors can be weighted by a Gaussian mask to control the smoothness of the spacial movement.

5 Application 1: Anomaly Event Detection

5.1 Data Sets

We use the UCSD abnormal event detection data set [40] for evaluation. The data set consists of two sections of two different scenarios. We use the videos in Section 2 that consists of 16 training and 12 test sequences. Each sequence has about 180 frames. The training videos capture only normal motions of walking crowd, while the testing ones have abnormal motions such as bikers, skaters, and small carts. Only eight of 12 testing sequences are provided with pixel-level binary mask ground truth. As our target is to discover and localize the abnormal events, only sequences with pixel-level ground truth are evaluated.

5.2 Training

We first extract features at locations with notable motions in the training data set because abnormal events cannot happen without movement. The features we used are histogram of oriented gradients (HOG) [35] and histogram of oriented flows (HOF) [41] with 16×16 patches. Feature quantization is then applied by k-means clustering. These cluster centers are used as codewords.

5.3 Testing

On testing, at any location with motions, we compute features and the distance to its nearest codeword. These distances are then used as prediction values for the local pixels. The far distance implies the high likelihood of being abnormal. The pixels with no motion are assigned zero distances. To introduce negative values, we subtract these distances by a threshold. This distance map is now a 3D array of positive and negative scores that can be passed to the subvolume search [8] for event localization. For our approach, we assume that the abnormal events will occur across m different scales (e.g., $d_1 \dots d_m$). We use *Integral Image* [9] to compute the sum scores of different local windows of different scales (e.g., squares with d_i-long sides) at each frame. This process will result in a 4D discriminative score map, which is then input to our Max-Path algorithm for discovering abnormal events. The Max-Path algorithm used in this experiment goes with multiple scale extension and the local neighbors of 5×5. The spacial step is 1 pixel.

TABLE 1
Abnormal Event Localization Results

Algorithm	Subvolume[8]	**Our Max-Path**
Average Accuracy	23.98	**60.20**

Our Max-Path algorithm significantly improves the localization accuracy compared to subvolume search [8] thanks to the constraint relaxation.

5.4 Results

For evaluations, we build ground truth by drawing bounding boxes around the provided masks of abnormal events. We use PASCAL metric (e.g., overlap area divided by union area of predicted and ground-truth boxes) to evaluate the localization accuracy. At every frame, if both prediction and ground truth are positive, then the PASCAL metric is applied to compute the localization score. If both of them are negative, then the score is assigned 1, otherwise this score is 0. Table 1 shows the average accuracy of abnormal event detection and localization. Our Max-Path algorithm significantly outperforms subvolume search more than 35 percent as a result of relaxing the 3D bounding box constraint. Fig. 5 compares the results of our Max-Path search and the subvolume search [8]. The first two rows are from a relatively simple sequence, while the last two rows are from another more difficult one due to the noisy motions of the walking crowd. In both cases, subvolume search predicts large volumes covering most of the video. Even though with a very noisy confidence map, our Max-Path search can localize event accurately. This is true because the false positives appear randomly at inconsistent spacial locations, in a long run, their accumulated scores cannot compete to those of the true event paths. On the other hand, the short-run missed or weak detections caused by occlusions can be resolved and linked to the main path as long as the final score can be further improved after the drops. Finally, experimental results showed that Max-Path search can automatically discovers the starting and ending points of events (see Fig. 5).

6 Application 2: Walking Person Localization

6.1 Data Sets

We use two data sets: *TUD-MotionPairs* [42] for training and our *NTU-UIUC YouTube Walking* for testing. *TUD-MotionPairs* is a fully annotated data set contains image pairs of outdoor walking pedestrians for evaluating pedestrian detection algorithms that employ motion information.

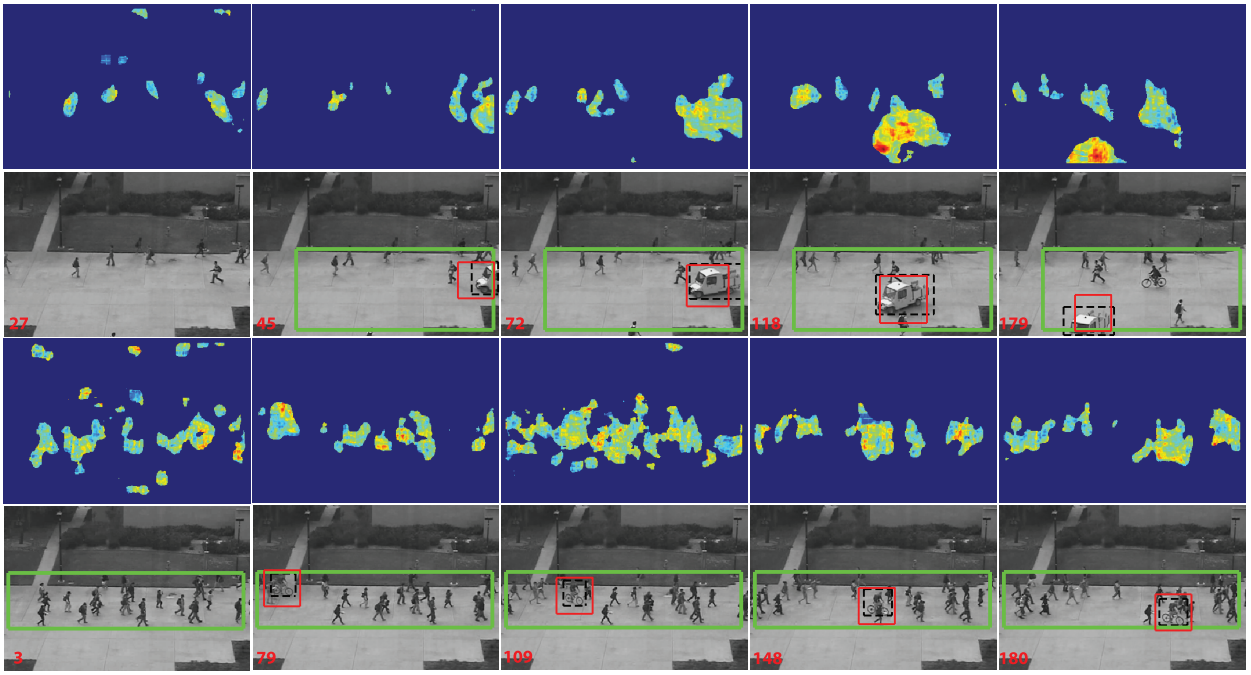

Fig. 5. *Abnormal event detection*: Demonstration of abnormal event detection. The odd rows are confidence maps; even rows are localization results. The results of subvolume search [8] are visualized in green boxes, the results of Max-Path search are in red, and ground truth is in dashed-black. The subvolume search covers a large portion of the video. Max-Path locates events more accurately by relaxing the constraint, and it can automatically discover the starting and ending points of events.

These image pairs include 1,092 positive pairs, 192 negative pairs, and 26 additional negative pairs for further bootstrapping training. *NTU-UIUC YouTube Walking* data set contains two long video sequences (800-900 frames per sequence) and 25 short video sequences (100-150 frames per sequence) downloaded from YouTube making a total of 4,083 annotated bounding boxes. These videos are real-world sequences including outdoor walking and indoor fashion shows of catwalk models performance. The sequences are very challenging due to their low quality with compression artifacts, appearing in crowded scenes, many partial and full occlusions, significant scale changes, different lighting conditions, noisy background motions, camera shaking motions (Fig. 6).

6.2 Training a Walker Detector

We use a global representation of pedestrian by HOG [35], IMHd2 [36] which is a variation of HOF [41], and simple self-similarity [36]. These features are then trained on a linear support vector machine (SVM) with one more additional bootstrapping round on hard negative set of *TUD-MotionPairs*.

6.3 Walking Localization Algorithms

We slide the trained detector over the test sequence at multiple scales. The sliding process results in a 4D output prediction map, which will be passed to a localization algorithm to process. This map does often contain false positives and missed detections due to the imperfect base detector. For quantitative evaluations, we implement two baseline algorithms for walking localization. The first one is to simply choose the maximum detection score at every frame over all scales. It is actually a variant of nonmaximum suppression, and it is more reasonable than nonmaximum suppression provided that there is one walking person in the sequence. We call this algorithm *Greedy-Suppression*. Another baseline algorithm is the *Spatiotemporal-Smooth* which is straightforwardly averaging k-consecutive boxes that are results from the Greedy-Suppression algorithm.

Besides baseline algorithms, we also compare our framework to an online tracking algorithm. We use the

Fig. 6. *NTU-UIUC Walking data set*: Twenty-seven realistic video sequences downloaded from YouTube. The two upper rows are snapshots of outdoor sequences. The last two rows are those from indoor fashion shows. These realistic videos are low quality, captured in crowded scenes, occlusions, complex background motions.

TABLE 2
Walking Localization Accuracy

Algorithm	Average accuracy
Incremental Learning Tracking [43]*	30.30
Greedy-Suppression + [36]	50.11
Spatiotemporal-Smooth + [36]	47.47
Offline Tracking [31]	63.63
Our Max-Path	**73.98**

*Our Max-Path algorithm improves 10-27 percent of accuracy compared to the other algorithms. *The incremental learning tracking algorithm [43] is not directly comparable.*

Incremental Learning Tracking (IL-Tracking) with the source code provided by Ross et al. [43]. The IL-Tracking algorithm is initialized by the ground-truth bounding box of the first frame. We note that the IL-Tracking is not directly comparable to the other algorithms because first it requires initialization and second it does not use the prediction map. For a complete evaluation, we further compare our method to an offline tracking algorithm. We use Berclaz et al. [31] with the source code obtained directly from the authors of [31]. This method formulates offline tracking as a flow optimization problem with the inputs as confident maps. Their inputs are similar to ours in the way that they take confident maps as inputs. However, there are two main differences between theirs and ours. First, they take confident maps of probabilities (e.g., in $[0, \ldots, 1]$) while our method should take discriminative scores which are possible negatives (e.g., outputs of an SVM, mutual information scores). Second, they just deal with single scale that requires the inputs have to be 3D trellises. For a fair comparison, we experiment this method using the same trellises of discriminative scores as our Max-Path. In this experiment, our trellises are 4D, which are obtained from sliding a window detector at multiple scales (i.e., 10-12 scales). We test the algorithm [31] with two different input maps. In the first evaluation, we use a preselected scale, which is the scale that most frequently appears in the data set (the mean scale). In the second evaluation, we greedily choose the scale with the maximum score at each node. These steps will reduce our 4D trellises to 3D ones. These trellises are then normalized to $[0, \ldots, 1]$ values to make them probabilities. Finally, those 3D probability maps are inputted into [31] to find the best paths. The greedy-scale version degrades the accuracy of [31] about 3-4 percent; therefore, we only report their accuracy with scale preselection evaluation.

These algorithms are then compared to our proposed Max-Path algorithm to demonstrate the effectiveness and robustness of our algorithm. In this experiment, we use the Max-Path algorithm with the multiple scales extension. The node's neighbors are its nine-connected neighbors. The used spacial stepsize is 4 pixels, both temporal and scale stepsize are 1.

6.4 Results

We evaluate the localization accuracy at every frame by PASCAL metric, and report the average accuracy in Table 2. We also visualize the behaviors of different algorithms on two long outdoor walking sequences in Figs. 7 and 13. Our Max-Path algorithm improves 10-27 percent from the baselines and tracking algorithms. The IL-Tracking algorithm works only a short time at the beginning, then loses the tracks when occlusions occur. It works better on some other higher quality sequences but still loses the tracks when partial occlusions are presented (see Fig. 13). The greedy-suppression algorithm suffers from false positives and missed detections. The spatiotemporal smooth cannot make any difference from greedy-suppression if not making it worse, due to highly noisy false detections. The offline tracking (magenta curve) misses some early frames due to the small size of the

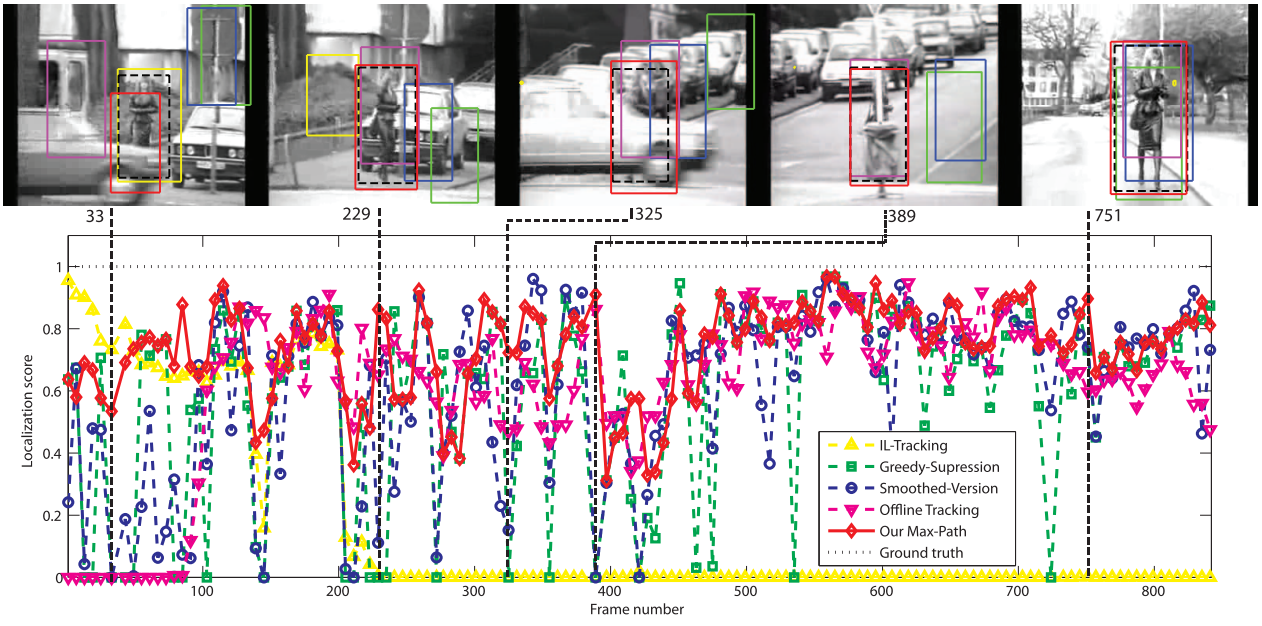

Fig. 7. *Walking localization*: The plots of localization scores from different algorithms on an outdoor walking sequence with visualized snapshots. IL-Tracking [43] works only at the beginning, then loses the tracks when occlusions occur. Offline tracking [31] misses early and last frames due to scale changing of the pedestrian. Greedy-Suppression and Spatiotemporal-Smooth perform poorly due to false positives and missed detections. Max-Path significantly improves the other algorithms with global optimized solution. The data points are dropped by ratio 1:7 for better representation (best viewed in color).

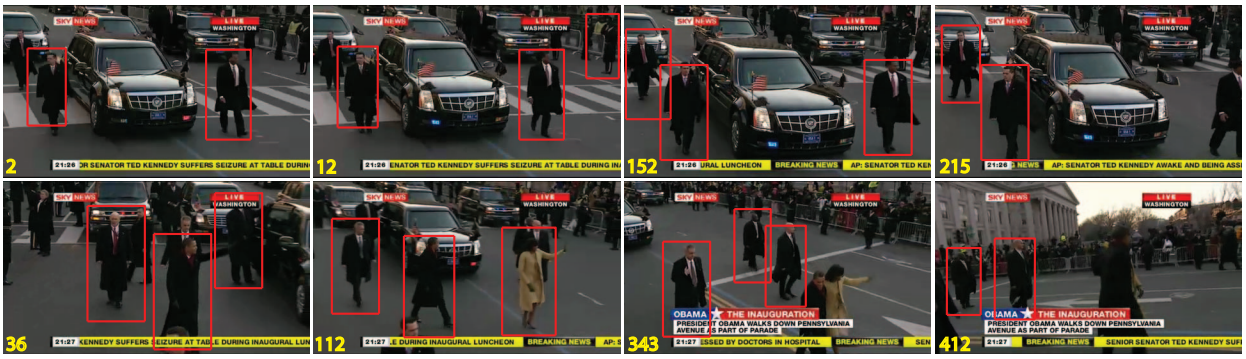

Fig. 8. *Pedestrian detection in videos*: The sequences are challenging due to complex camera and background motions.

pedestrian. We note that its localization score also drops in the last 80 frames due to the larger size of the walker when she approaches closer to the camera. On the other frames where the preselected scale is about the true scale, the offline tracking locates walker with similar accuracy as our Max-Path (red curve). In overall data set evaluation, Max-Path improves 10.35 percent from [31] thanks to its efficient optimization over multiple scales. In summary, Max-Path algorithm provides significant improvements over the other methods thanks to its global optimal solution over all spatiotemporal and scale spaces.

6.5 Detecting of Multiple Walking Pedestrians

We collect two sequences from YouTube which consist of 672 frames for evaluating this extension. These realistic sequences are the television news of the event that President Barack Obama walking to the White House on his inauguration day, which contains many walking people. We apply the detector in Section 6.2 with our proposed algorithm repeatedly to find the top $k = 5$ paths. Results are shown in Fig. 8. It is worth noting that unlike multiple object tracking, we do not identify different people due to the lack of appearance modeling. Instead, we only discover the top 5 best paths in the video.

7 Application 3: Running Detection and Localization

7.1 Data Sets

In this experiment, we use the *KTH* data set [44] for training and *NTU Running* data set for testing. KTH is a good data set for human action recognition which is widely used in the computer vision community. The data set consists of six types of actions. However, we only use walking and running sequences for training. We also exclude the sequences from Section 2 of KTH because this section contains videos captured with zooming-in and -out motions that are not suitable to our application. NTU Running data set consists of 15 positive sequences and 22 negative sequences captured at public and crowded scenes. The total number of frames is 8,096. Each positive video has exactly one running event while negative ones have no running motion. This data set is difficult because of the crowded scenes with many people which cause many partial and full occlusions. The positive videos are manually annotated with bounding boxes around the actors for each frame where the running events present. Some examples of the running data set are presented in Fig. 9.

7.2 Discriminative Scores

We use the interest point action representation in [8], which will be briefly described as follows. We first extract the STIPs [38] and compute the features HOG [35] and HOF [41] at these local interest points for both training and testing sequences. Assuming the class priors are equal, the local discriminative score of each interest point is then estimated by

$$s^{c+}(d) = MI(c+, d) = \log \frac{2}{1 + \frac{P(d|c-)}{P(d|c+)}}. \tag{10}$$

Then, the likelihood ratio is estimated as

$$\frac{P(d \mid c-)}{P(d \mid c+)} \approx \exp^{-\frac{1}{2\sigma^2}(\|d - d_{NN}^{c-}\|^2 - \|d - d_{NN}^{c+}\|^2)}. \tag{11}$$

In (11), d_{NN}^{c-} and d_{NN}^{c+} are nearest neighbors of d in negative and positive point set, respectively, and $\frac{1}{2\sigma^2}$ is adaptively estimated as suggested in [8] because it is more discriminative than a fixed bandwidth. We refer the reader to [8] for more detailed information about pointwise mutual information and adaptive bandwidth search. It is worth noting that the approximation notation is used because we use locality sensitive hashing (LSH) [45] to perform an approximate ϵ-nearest neighbors (ϵ-NN) search with a confident probability of p. In this experiment, we use $\epsilon = 3$ and $p = 0.9$ for all settings.

7.3 Detection and Localization Algorithms

On testing, given a sequence $\mathcal{Q} = \{d_i\}$, once the pointwise mutual information for each STIP $d_i \in \mathcal{Q}$ is computed, $\mathcal{Q}$

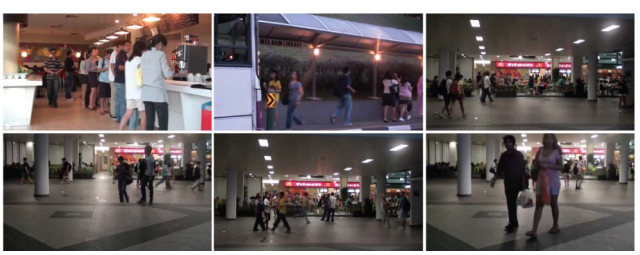

Fig. 9. *NTU Running data set*: The sequences are challenging due to complex background motions. The upper row images are from negative sequences, while the lower ones are from positive sequences.

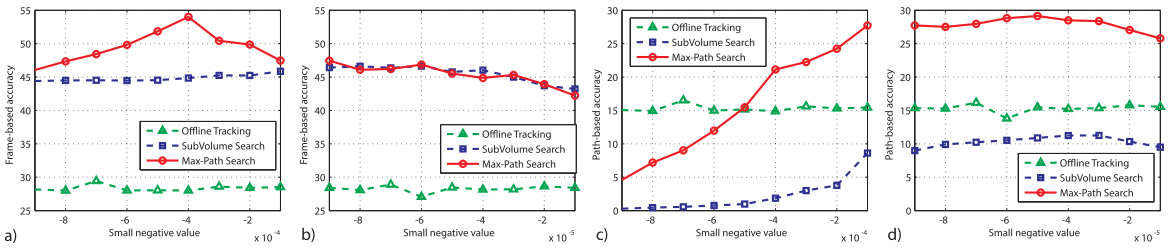

Fig. 10. *Parameter search*: The localization accuracy of subvolume search [8], offline tracking [31], and Max-Path search with different values of s_0. a) and b) Frame-based localization accuracy for different s_0 in 10^{-4} and 10^{-5} ranges, respectively. c) and d) Path-based localization accuracy for different s_0 in 10^{-4} and 10^{-5} ranges, respectively. Max-Path search outperforms the other methods by a large margin in both frame-based and path-based metrics.

becomes a sparse volume $\mathcal{V}$ with discriminative scores that are either positive or negative. The discriminative sparse volume $\mathcal{V}$ is then passed to algorithms such as subvolume search [8] and our Max-Path search [46] for detecting and localizing the running events. The subvolume search code provided by the authors is used to evaluate and compare to our Max-Path. In this experiment, we use Max-Path with multiple scale extension and a local neighborhood of 5×5. The spatial stepsize is 3 pixels. There is one difference in this experiment compared to the abnormal event detection. When building the trellis, at each local voxel, we compute the summations of multiple scale squares using *Integral Image* [9] in our abnormal detection experiment. However, in this experiment, because the running actors appear to be rectangle shapes rather than squares, we instead use multiple scale rectangles with ratio of width to height of 1:2. We also evaluate the offline tracking algorithm [31] using the same trellis as our Max-Path. The trellis of mutual information scores is first normalized to make it probabilities, then passed to [31] to detect and localize running.

7.4 Evaluations and Results

7.4.1 Evaluation Metrics

We evaluate the running localization by three different metrics: temporal localization accuracy, frame-based localization accuracy, and path-based localization accuracy. The temporal localization score is computed as the length (measured in frames) of the intersection divided by the union of detection and the ground truth. The frame-based metric is similar to the one used in two previous experiments, while the path-based metric is more difficult and more precisely measure the correctness of localization. Without loss of generality, we assume that the testing sequence has a length of n with the ground-truth path $P = \{b_i\}_{i=1}^n$. Here, we denote the annotated ground truth as a path P consists of n boxes b_i. It is worth noting that if the path starts and ends somewhere in the middle of the sequence then the first and the last boxes may be empty (e.g., a zero vector in $\mathbb{R}^4$). We also denote $\hat{P} = \{\hat{b}_i\}_{i=1}^n$ as an output predicted path. Frame-based accuracy is evaluated by

$$FA(P, \hat{P}) = \frac{1}{n} \sum_{i=1}^{n} BA(b_i, \hat{b}_i), \tag{12}$$

where $BA(b, \hat{b})$, the box accuracy between the ground-truth box b and the predicted box $\hat{b}$, is computed by

$$BA(b, \hat{b}) = \begin{cases} 1, & \text{if } b = 0 \text{ and } \hat{b} = 0, \\ \frac{b \cap \hat{b}}{b \cup \hat{b}}, & \text{otherwise.} \end{cases} \tag{13}$$

The path-based accuracy is basically the overlapped volume of two paths divided by the union volume of those two paths, which is approximated by

$$PA(P, \hat{P}) = \frac{\sum_{i=1}^{n} b_i \cap \hat{b}_i}{\sum_{i=1}^{n} b_i \cup \hat{b}_i}. \tag{14}$$

We note that the path-based metric is more difficult and more precise for evaluating the localization. This score will approach to $1 = 100\%$ when the predicted path approaches to the ground-truth path, and will be 0 when the two paths have no overlap. On the other hand, the frame-based metric is easier. The accuracy may be positive even in the cases that there is no overlap between two paths due to the negative frames. It is also worth noting that temporal localization is less challenging compared to the other two metrics because only temporal information is evaluated while ignoring the spatial information.

7.4.2 Results

Running Localization. Similarly to [8], we also subtract a small negative value s_0 to the sparse volume before passing it to the localization algorithms. For a fair comparison, we search a wide range of value of s_0, then compare the best accuracies of the two algorithms. The values for searching s_0 are 18 values which are $\{-1, \ldots, -9 \times 10^{-4}\}$ and $\{-1, \ldots, -9 \times 10^{-5}\}$. The behavior of the accuracies versus the parameter s_0 are visualized in Fig. 10. Compared to subvolume search [8] and Max-Path, offline tracking [31] is less sensitive to s_0 because they have a preprocessing step in which they smooth (apply a logarithm function) the probability map before their flow optimization algorithm. The best accuracies of all algorithms are reported in Table 3. Our Max-Path search significantly outperforms subvolume search [8] and offline tracking [31] on all three different evaluation metrics. More specifically, Max-Path improves

TABLE 3
Running Localization Results

Evaluation metric	[8]	[31]	Our Max-Path
Temporal localization	49.79	36.57	**76.66**
Frame-based localization	45.85	29.43	**54.00**
Path-based localization	11.25	16.51	**29.14**

Our Max-Path algorithm significantly outperforms subvolume search [8] and offline tracking [31] in all three different evaluation metrics.

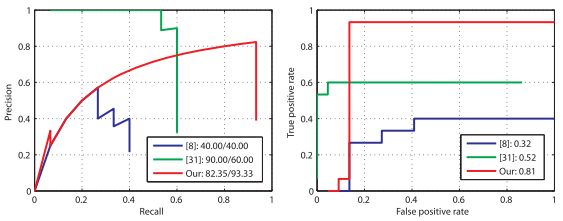

Fig. 11. *Running detection results*. Precision/recall and ROC curves of subvolume search [8], offline tracking [31], and Max-Path search. The numbers beside the legend items are the precision at the equal rate (left) and the area under ROC curves (right).

26.87 percent on temporal localization metric, 8.15 percent on frame-based metric, and 17.89 percent on path-based metric from subvolume search [8]. It improves 40.09 percent on temporal localization, 24.57 percent frame-based metric, and 12.63 percent path-based metric from offline tracking [31]. It is worth noting that for testing the running localization algorithms, only positive sequences are used.

Fig. 12 visualizes our Max-Path localization results compared to subvolume search on two different sequences. The subvolume search covers a large portion of the video due to its hard constraint of 3D bounding box; thus, its localization scores (visualized in blue curves) are relatively low around 0.2-0.4. Offline tracking misses the whole running event in the first sequence, and works better in the second sequence. However, it still does not well detect the starting and ending frames. Our Max-Path provides more flexible and precise localizations as an advantage of the constraint relaxation. Max-Path also precisely locates the running both spatially and temporally.

Running Detection. We further test the algorithm in the detection task. As suggested in [8], we count a prediction as a correct detection if the path-based localization score is at least $1/8$. The precision and recall are then evaluated on three algorithms with their best parameters s_0, which is -3×10^{-5} for subvolume search, -7×10^{-4} for offline tracking, and -5×10^{-5} for Max-Path. The precision is computed as the number of correct detections divided by total number of detections, and the recall is evaluated as the number of correct detections divided by total number of action instances. The precision/recall and ROC curves of all algorithms are shown in Fig. 11. We note that the detection curves of Subvolume search and Max-Path search are almost similar around 0 to 25 percent of recall. This is because the two algorithms use the same mutual information scores of local spatiotemporal interest points with different detection hypothesizes: subvolume and spatiotemporal path. Subvolume detections normally cover larger portions of video than spatiotemporal paths. In fact, it not only covers the running action but also other background motions. For video sequences with less noisy background motions, the difference between the subvolume and Max-Path is not notable. However, for videos with crowded background motions, many more interest points' scores from the background will be added to subvolume detections; hence its precision significantly drops compared to our Max-Path. This observation shows that our Max-Path is more appropriate for "dynamic" actions than subvolume search, especially with videos containing complex background motions.

We also report the precision at the equal rate and the area under the ROC curve of those three algorithms. The equal-rate precision is defined as the maximum of the minimum of precision and recall over the curve points (e.g., $\max_{p_i \in curve}\{\min\{precision_{p_i}, recall_{p_i}\}\}$, where p_i is a point on the curve). Subvolume search reaches equal rate at precision and recall at 40 percent, offline tracking is at 60 percent, while Max-Path reaches the equal rate at precision is 82.35 percent. The area under ROC curve of subvolume search is 0.32, that of offline tracking is 0.52, and our Max-Path is 0.81.

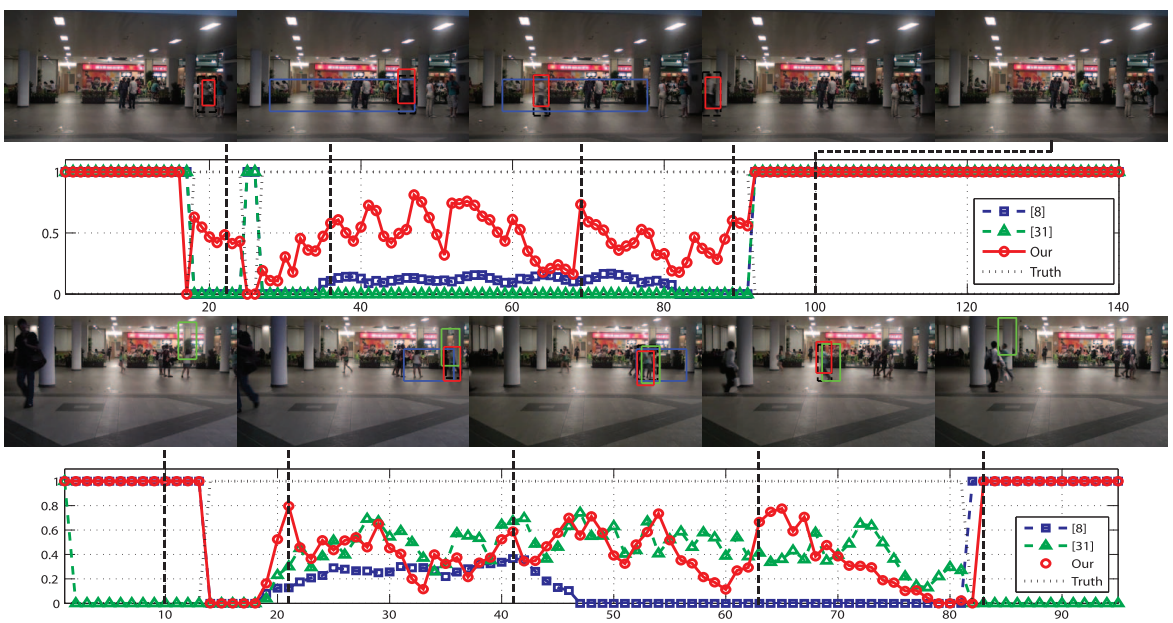

Fig. 12. *Running localization*: Demonstration of running event localization. The results of subvolume search [8] are visualized in blue boxes, offline tracking [31] are green, Max-Path search are red, and ground truth is in dashed-black. The ground truth starting and ending frames for the first sequence are 18 and 91, those of the second sequence are 14 and 81.

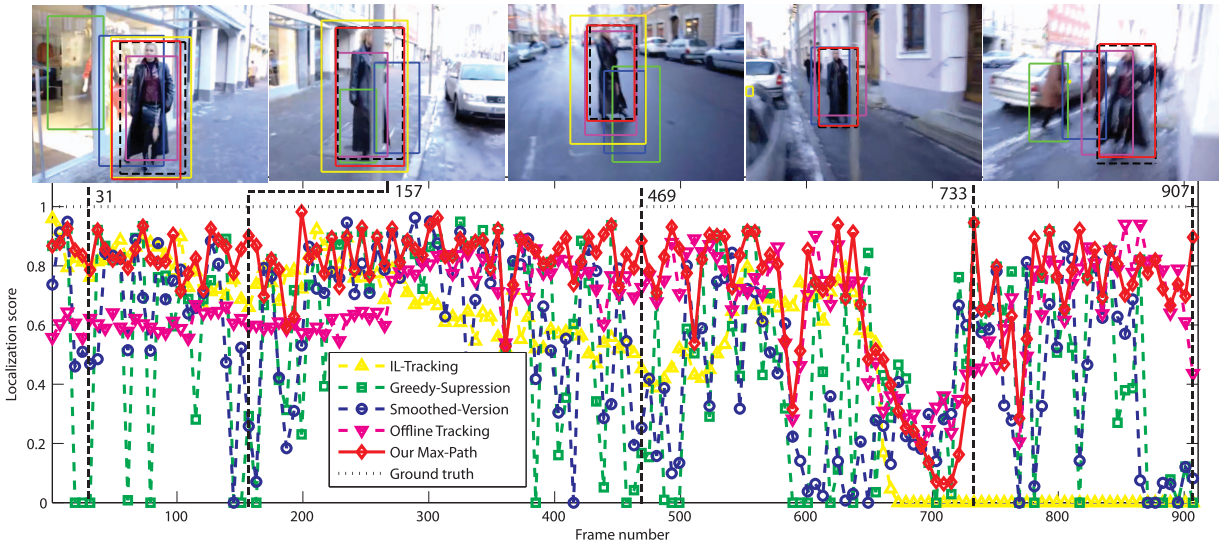

Fig. 13. *Walking localization*: Demonstration of walking localization on a simpler sequence. The online tracking algorithm works at early frames, but still loses the target when partial occlusions present. The offline tracking works reasonably; however, it still cannot handle different scales. The data points are dropped by ratio 1:7 for better representation (best viewed in color).

8 Conclusions

We have proposed a novel approach for detecting complex and dynamic events. The relaxation from video subvolume to spatiotemporal path makes the method more flexible and, hence, well addressed to complex events that could not be precisely covered by subvolume. The global optimal solution of our Max-Path algorithm improves the smoothness of the event, thus eliminating the false positives and alleviates missed or weak detections due to occlusions and the image low quality. In addition, Max-Path's lowest complexity makes it efficient to search for spatiotemporal paths in a large 5D space of spatiotemporal, scale, and shape. Finally, our experiments on three different types of events, using different features, with both local and global object representations proved that our proposed method is general and flexible enough to be applied to a wide class of events.

In conclusion, this paper contributes to the computer vision literature a novel approach for video event detection and localization with significant improvements over the state-of-the-art methods. It will strongly benefit a class of problems in video event detection and localization, especially for the complex and dynamic events. In practice, thanks to its low complexity, Max-Path search has been used as an efficient method for fast inference and complex structured learning problem [33] or applied to improve the speed performance [47].

Acknowledgments

This work was supported in part by Nanyang Assistant Professorship SUG M4080134.040.

References

[1] A. Efros, A. Berg, G. Mori, and J. Malik, "Recognizing Action at a Distance," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 726-733, 2003.

[2] Y. Ke, R. Sukthankar, and M. Hebert, "Volumetric Features for Video Event Detection," *Int'l J. Computer Vision*, vol. 88, pp. 339-362, 2010.

[3] M.D. Rodriguez, J. Ahmed, and M. Shah, "Action Mach: A Spatio-Temporal Maximum Average Correlation Height Filter for Action Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2008.

[4] K. Derpanis, M. Sizintsev, K. Cannons, and P. Wildes, "Efficient Action Spotting Based on a Spacetime Oriented Structure Representation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2010.

[5] C.H. Lampert, M.B. Blaschko, and T. Hofmann, "Beyond Sliding Windows: Object Localization by Efficient Subwindow Search," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2008.

[6] C.H. Lampert, M.B. Blaschko, and T. Hofmann, "Efficient Subwindow Search: A Branch and Bound Framework for Object Localization," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2129-2142, Dec. 2009.

[7] J. Yuan, Z. Liu, and Y. Wu, "Discriminative Subvolume Search for Efficient Action Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 2442-2449, 2009.

[8] J. Yuan, Z. Liu, and Y. Wu, "Discriminative Video Pattern Search for Efficient Action Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1728-1743, Sept. 2011.

[9] P. Viola and M. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple Features," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2001.

[10] N. Ikizler and D. Forsyth, "Searching Video for Complex Activities with Finite State Models," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2007.

[11] N. Ikizler and D. Forsyth, "Searching for Complex Human Activities with No Visual Examples," *Int'l J. Computer Vision*, vol. 80, no. 3, pp. 337-357, 2008.

[12] I. Laptev and P. Perez, "Retrieving Actions in Movies," *Proc. 11th IEEE Int'l Conf. Computer Vision*, 2007.

[13] Y. Ke, R. Sukthankar, and M. Hebert, "Event Detection in Crowded Videos," *Proc. 11th IEEE Int'l Conf. Computer Vision*, 2007.

[14] H. Jiang, M. Drew, and Z. Li, "Successive Convex Matching for Action Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2006.

[15] H. Jiang, M. Drew, and Z. Li, "Action Detection in Cluttered Video with Successive Convex Matching," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 20, no. 1, pp. 50-64, Jan. 2010.

[16] H. Seo and P. Milanfar, "Detection of Human Actions from a Single Example," *Proc. 12th IEEE Int'l Conf. Computer Vision*, 2009.

[17] O. Boiman and M. Irani, "Detecting Irregularities in Images and in Video," *Int'l J. Computer Vision*, vol. 74, pp. 17-31, 2007.

[18] Y. Hu, L. Cao, F. Lv, S. Yan, Y. Gong, and T.S. Huang, "Action Detection in Complex Scenes with Spatial and Temporal Ambiguities," *Proc. 12th IEEE Int'l Conf. Computer Vision*, 2009.

[19] J. Gall, A. Yao, N. Razavi, L. van Gool, and V. Lempitsky, "Hough Forests for Object Detection, Tracking, and Action Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 33, no. 11, pp. 2188-2202, Nov. 2011.
[20] G. Yu, J. Yuan, and Z. Liu, "Unsupervised Random Forest Indexing for Fast Action Search," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2011.
[21] G. Yu, A. Norberto, J. Yuan, and Z. Liu, "Fast Action Detection via Discriminative Random Forest Voting and Top-k Subvolume Search," *IEEE Trans. Multimedia,* vol. 13, no. 3, pp. 507-517, June 2011.
[22] A. Gaidon, Z. Harchaoui, and C. Schmid, "Actom Sequence Models for Efficient Action Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2011.
[23] L. Cao, Z. Liu, and T. Huang, "Cross-Data Set Action Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2010.
[24] Z. Zhang, Y. Cao, D. Salvi, K. Oliver, J. Waggoner, and S. Wang, "Free-Shape Subwindow Search for Object Localization," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2010.
[25] M. Everingham, J. Sivic, and A. Zisserman, "'Hello! My Name Is... Buffy'—Automatic Naming of Characters in TV Video," *Proc. British Machine Vision Conf.,* pp. 899-908, 2006.
[26] B. Leibe, K. Schindler, and L. Gool, "Coupled Detection and Trajectory Estimation for Multi-Object Tracking," *Proc. 11th IEEE Int'l Conf. Computer Vision,* 2007.
[27] C. Huang, B. Wu, and R. Nevatia, "Robust Object Tracking by Hierarchical Association of Detection Responses," *Proc. European Conf. Computer Vision,* 2008.
[28] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Gool, "Robust Tracking-by-Detection Using a Detector Confidence Particle Filter," *Proc. 12th IEEE Int'l Conf. Computer Vision,* 2009.
[29] S. Stalder, H. Grabner, and L.V. Gool, "Cascaded Confidence Filtering for Improved Tracking-by-Detection," *Proc. European Conf. Computer Vision,* 2010.
[30] M. Andriluka, S. Roth, and B. Schiele, "People-Tracking-by-Detection and People-Detection-by-Tracking," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2008.
[31] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua, "Multiple Object Tracking Using K-Shortest Paths Optimization," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 33, no. 9, pp. 1806-1819, Sept. 2011.
[32] H. Pirsiavash, D. Ramanan, and C. Fowlkes, "Globally-Optimal Greedy Algorithms for Tracking a Variable Number of Objects," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2011.
[33] D. Tran and J. Yuan, "Max-Margin Structured Output Regression for Spatio-Temporal Action Localization," *Proc. Neural Information Processing Systems,* 2012.
[34] B. Jon, "Programming Pearls: Algorithm Design Techniques," *Comm. ACM,* vol. 27, no. 9, pp. 865-873, 1984.
[35] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2005.
[36] S. Walk, N. Majer, K. Schindler, and B. Schiele, "New Features and Insights for Pedestrian Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2010.
[37] P. Dollr, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior Recognition via Sparse Spatio-Temporal Features," *Proc. IEEE Int'l Conf. Computer Vision Second Int'l Workshop Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS),* pp. 65-72, 2005.
[38] I. Laptev and T. Lindeberg, "Space-Time Interest Points," *Proc. Ninth IEEE Int'l Conf. Computer Vision,* 2003.
[39] J. Yuan, J. Meng, Y. Wu, and J. Luo, "Mining Recurring Events through Forest Growing," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 18, no. 11, pp. 1597-1607, Nov. 2008.
[40] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, "Anomaly Detection in Crowded Scenes," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2010.
[41] N. Dalal, B. Triggs, and C. Schmid, "Human Detection Using Oriented Histograms of Flow and Appearance," *Proc. European Conf. Computer Vision,* 2006.
[42] C. Wojek, S. Walk, and B. Schiele, "Multi-Cue Onboard Pedestrian Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* 2009.
[43] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental Learning for Robust Visual Tracking," *Int'l J. Computer Vision,* vol. 77, pp. 125-141, 2008.
[44] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing Human Actions: A Local SVM Approach," *Proc. Int'l Conf. Pattern Recognition,* 2004.
[45] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, "Locality Sensitive Hashing Scheme Based on P-Stable Distribution," *Proc. 20th Ann. Symp. Computational Geometry,* pp. 253-262, 2004.
[46] D. Tran and J. Yuan, "Optimal Spatio-Temporal Path Discovery for Video Event Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),* pp. 3321-3328, 2011.
[47] G. Yu, J. Yuan, and Z. Liu, "Real-Time Human Action Search Using Random Forest Based Hough Voting," *Proc. ACM Multimedia Conf.,* 2011.

Du Tran received the BS and MS degrees in computer science from Ho Chi Minh City University of Science and the University of Illinois at Urbana-Champaign, respectively. He is currently working toward the PhD degree in the Department of Computer Science at Dartmouth College, Hanover, New Hampshire. Before arriving at Dartmouth College, he was a research associate at Nanyang Technological University, Singapore. His research interests include computer vision, machine learning, and computer graphics, with specific interests in human activity and video event analysis. He received the Vietnam Education Foundation Fellowship in 2006 and the Dartmouth Presidential Fellowship in 2012. He is a student member of the IEEE.

Junsong Yuan received the PhD degree from Northwestern University, Evanston, Illinois, and the MEng degree from the National University of Singapore. Before that, he graduated from the Special Class for the Gifted Young at Huazhong University of Science and Technology and received the BEng degree in communication engineering. He is currently a Nanyang Assistant Professor at Nanyang Technological University (NTU), Singapore. His research interests include computer vision, video analytics, large-scale visual search and mining, human computer interaction, biomedical image analysis, etc. He is serving as area chair for the IEEE Winter Conference on Computer Vision (WACV '14) and IEEE Conference on Multimedia Expo (ICME '14). He is also serving as the organization cochair and area chair for the Asian Conference on Computer Vision (ACCV '14). He is an associate editor for the *Visual Computer Journal* (Springer) and *Journal of Multimedia* (Academy). He has filed three US patents and two provisional US patents. He is a member of the IEEE and ACM.

David Forsyth received the BSc and MSc degrees in electrical engineering from the University of the Witwatersrand, Johannesburg, South Africa, and the DPhil degree from Balliol College, Oxford, United Kingdom. He was a professor at the University of California, Berkeley. He is currently a professor at the University of Illinois at Urbana-Champaign. He has published more than 100 papers on computer vision, computer graphics, and machine learning. He is a coauthor (with J. Ponce) of *Computer Vision: A Modern Approach* (Prentice-Hall, 2002). He was a program cochair for the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) in 2000, a general cochair for CVPR '06, and a program cochair for the European Conference on Computer Vision '08 and CVPR '11. He received the IEEE Technical Achievement Award in 2005. He is the editor in chief of the *IEEE Transactions on Pattern Analysis and Machine Intelligence*. He is a fellow of the IEEE.