J Sign Process Syst (2011) 65:49–62
DOI 10.1007/s11265-010-0540-3

# Automatic Detection of Object of Interest and Tracking in Active Video

**Jiawei Huang · Ze-Nian Li**

Received: 1 April 2010 / Revised: 30 June 2010 / Accepted: 20 September 2010 / Published online: 9 October 2010

**Abstract** We propose a novel method for automatic detection and tracking of Object of Interest (OOI) from actively acquired videos by non-calibrated cameras. The proposed approach benefits from the object-centered property of Active Video and facilitates *self-initialization* in tracking. We first use a *color-saliency weighted Probability-of-Boundary* (cPoB) map for key-point filtering and salient region detection. *Successive Classification and Refinement* (SCR) is used for tracking between two consecutive frames. A strong classifier trained on-the-fly by AdaBoost is utilized for key-point classification and subsequent Linear Programming solves a maximum similarity problem to reject outliers. Experiments demonstrate the importance of Active Video during the data collection phase and confirm that our new approach can automatically detect and reliably track OOI in videos.

**Keywords** Saliency detection · Feature matching · Visual attention · Tracking · Active video

## 1 Introduction

Multimedia is one of the most fascinating and fastest growing areas in current entrainment, education, training, simulation, and research. Texts, pictures, animations, sounds, and videos are often seamlessly blended, resulting in dazzling and interactive presentations. Before the advent of computers, all these varied media were difficult to put together. Computers enabled us to combine them into applications so that they could be efficiently stored or reused.

Various types of image and video processing software are prevalent in our daily lives and attract the attention of many Computer Vision (CV) researchers. We benefit greatly from new and cutting-edge multimedia applications. For example, *Seam Carving* [3], an image resizing algorithm developed by Avidan and Shamir, is now adopt by Adobe into their new release of PhotoShop. *AutoCollage* [30, 31] helps users to construct a visually appealing collage from a collection of input images. Microsoft *Surface* is another emerging multimedia system, which uses cameras and image recognition in the infrared spectrum to do the recognition task.

Object detection and tracking is the key to many efficient, accurate, and user friendly implementations of machine vision applications. For example, object detection and tracking techniques can be used in surveillance [2, 35], transportation systems [24, 38], driver assistance [1, 14], smart rooms [19, 39], or visual data summarizing [25, 33]. However, due to the great complexities such as pose, lighting, or clutter, and insurmountable amount of data that each digital image and video carries, it remains a challenge to detect and track objects accurately.

*Object detection* deals with detecting instances of semantically meaningful objects of certain classes, such as humans, buildings, or cars in digital images and videos. Object detection is a relatively broad domain in the CV literature and it attracts great interests in the CV research community. *Saliency Detection* or *Object*

J. Huang (✉) · Z.-N. Li
Vision and Media Lab, School of Computing Science,
Simon Fraser University, Burnaby, BC, V5A 1S6, Canada
e-mail: jha48@cs.sfu.ca

Z.-N. Li
e-mail: li@cs.sfu.ca

*of Interest* (OOI) *Detection* in this paper is a specific type of object detection based on the Human Visual Attention (HVA) model [20, 21], which is inspired by intelligent vision systems. *Saliency* is the distinct subjective perceptual quality which makes some items in the scene stand out from their neighbors and immediately grab our attention. It can appear in the form of color, texture, shape and motion, for example. Human beings have a remarkable ability to interpret complex scenes in real time using intermediate and higher levels of visual processes [36]. These vision processes appear to select a subset of the available sensory information before further processing for reducing the complexity of scene analysis. This is known as "focus of attention" [29] in the psychological literature.

*Tracking* is the process of locating one or more moving objects in video frames and yielding the location of the moving objects. The main difficulty in video tracking is to associate target locations in consecutive frames, especially when objects are moving fast or occluded by other visual targets. The goal of visual object tracking is to repeatedly localize an object in successive frames. Most object trackers search for the target locally in new frames and they consist of several key components: (1) an object representation, such as an appearance model by color histogram [10] or bag of samples for classification [2]; (2) a similarity measure between the reference model and candidate targets, such as Bhattacharya coefficient [13], Earth Mover's Distance [32]; and (3) a local mode-seeking method for finding the most similar location in new frames, such as the mean-shift [8, 12] or Lucas-Kanade algorithm [4].

In order to find "meaningful" features [23], or to represent useful and reliable contexts, we develop a *color-saliency weighted Probability-of-Boundary* (cPoB) confidence map [18] which combines color and edge cues. To tackle the detection and tracking problem, we formulate the task as a self-initialization and learning problem in Active Video. we propose *Successive Classification and Refinement* (SCR) [17] for automatic detecting and tracking OOI in Active Video. Our image processing methods that employ color difference, edge information, and motion cues in visual data are *bottom-up* and data driven. Meanwhile, the applications of the HVA model and Machine Learning techniques in tracking are considered *top-down* and model based CV processes. We combine the bottom-up and top-down vision procedures in our research to extract and track OOI from images and videos. An overall picture of proposed algorithm is shown in Fig. 1.

The organization of this paper is as follows. Section 2 introduces Saliency Context analysis in Active Video using cPoB to detect OOI automatically. We describe our SCR tracking algorithm in Section 3. Results and comparison between our algorithm and existing ones will be reported in Section 4. Section 5 concludes the paper and we outline the potential future work in Section 6.

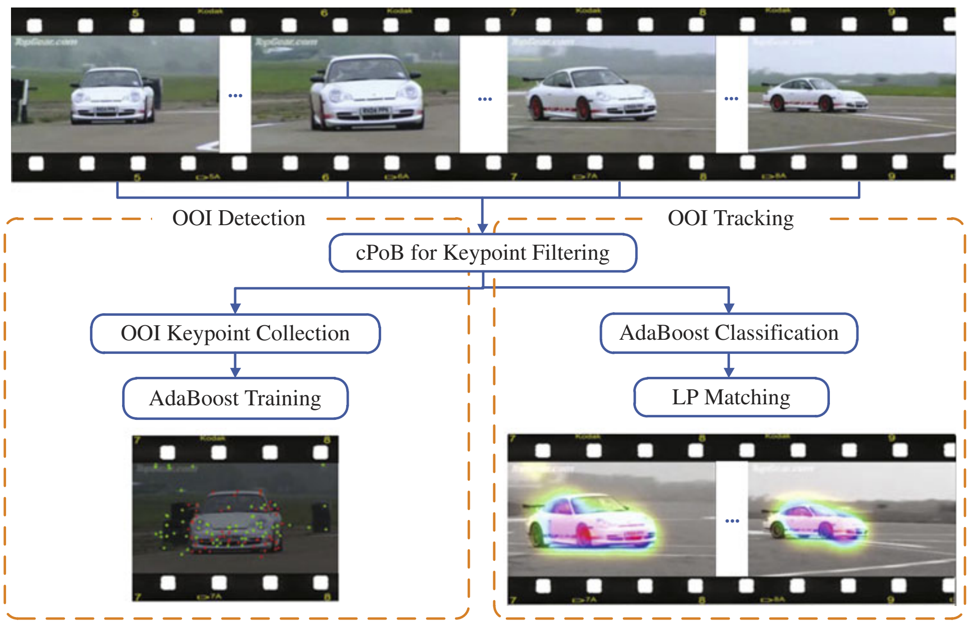


**Figure 1** Illustration of automatic detection of OOI and tracking algorithm. The first several frames from the very beginning of each video clip are used for OOI detection and the following frames are served for tracking.

## 2 Saliency Context

Kadir and Brady [23] summarized *visual saliency* with three characteristics: (1) distinct geometric features, (2) rarity, and (3) local signal complexity (or unpredictability)—this is to argue that saliency can be more reliably detected locally, similar to the "pop-out" features [22] that are observed in the pre-attentive stage. However, it is known that the context information is vital in the subsequent attentive stage. We will explore the following descriptions for saliency context in video analysis.

### 2.1 Active Video

In general, videos can be classified into two types: *passive video* and *active video*. A video produced by a static surveillance camera is a good example of the former where the camera's function is to passively record all objects passing by in front of it and the background. However, videos generated by an active vision system, such as our eyes, will not look like that. Digital videos taken by human operators are more purposive—aimed to depict interesting features of the OOI and capture their actions. Typical examples will be filming, professional video cameramen covering sporting events, or an amateur shooting at a tourist scene. As introduced by Lu and Li in [26], this type of videos can be characterized as *Active Video*. They are very much object-centered and often exhibit prominent *catching and holding* behaviors of the video camera operators. In order to catch the OOI, it is common for the operator to initiate pan/tilt camera movements. The movement can be rapid that resembles the saccadic human eye movements [7]. Afterwards, the camera stays still (holding) so later the viewers will have opportunity to examine the details of the OOI. When dealing with moving objects, smooth (usually not so rapid) pan/tilt movements are used for *smooth pursuit*. It should be apparent that Active Video is object-based and full of actions.

Figure 2 shows the most common camera motions in the traditional filming terminology.

(1) **Panning and tilting**. A pan is a horizontal camera movement in which the camera rotates left or right about a central axis. The panning movement usually has a definite direction and a start and end point. In Active Video, panning is used to initially catch the OOI. If the object is moving, it can be used to follow the moving OOI, i.e., smooth pursuit. In film making, a pan usually begins and

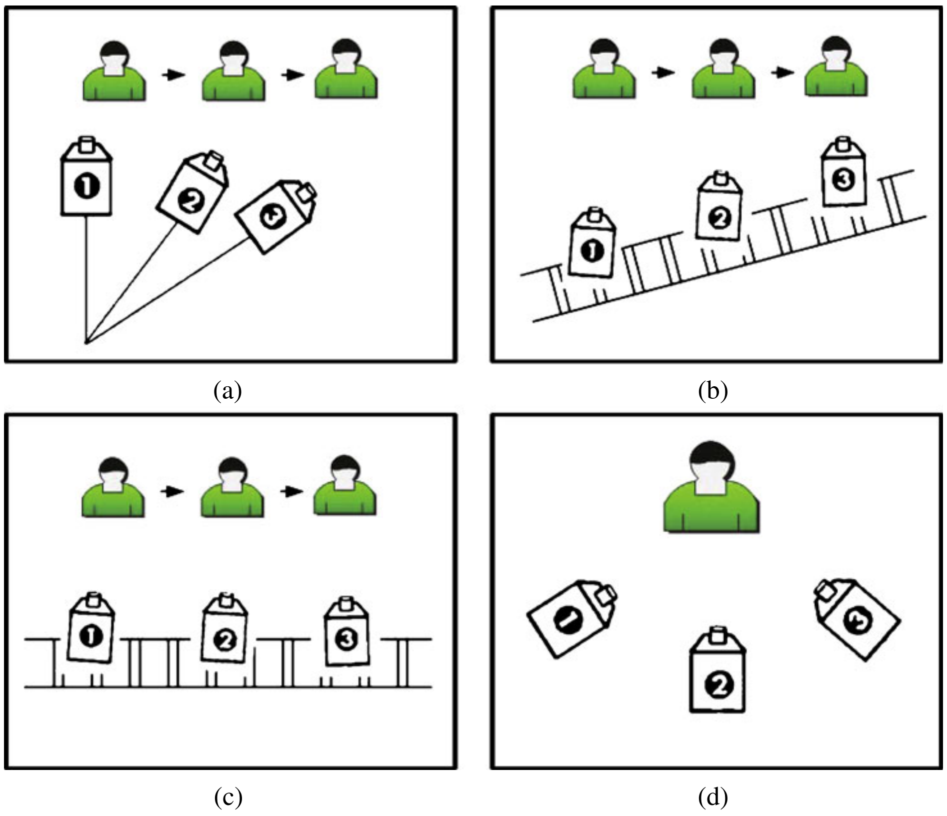


**Figure 2** Different types of camera motions: **a** pan/tilt, **b** dolly, **c** track, and **d** revolve.

ends with a few seconds of still picture to give greater impact. The speed of a pan across a subject creates a particular mood as well as establishing the viewer's relationship with the subject. A tilt is a vertical camera movement in which the camera points up or down from a stationary location. Tilting can often be used in combination with panning. It is not used as frequently as panning because humans tend to look left and right more than we do up and down.

(2) **Dollying and tracking**.[1] A dolly is a cart which travels along tracks. The camera is mounted on the dolly and records the shot as it moves. The term *track shot* is widely considered to be synonymous with *dolly shot*. However, some professionals prefer the more rigid terminology which defines *dolly* as in-and-out camera movement, whereas *track* is the camera movement parallel to object movement in order to keep the OOI in the central area of the picture.

(3) **Revolving**. The camera moves around the OOI to capture multiple views of it from different angles. This is usually adopted when the OOI is static and after the catching and holding cycle has completed. A typical example will be a tourist attempting to capture more details of a sculpture from multiple views.

Different camera motions are utilized in Active Video to convey or emphasize certain OOI. They provide useful cues for *video segmentation*, i.e., dividing a long video into shorter *shots*. After this is achieved, the object-centered property of these videos can further facilitate OOI tracking which will be shown in detail in Section 3.

### 2.2 Saliency Map and Keypoint Filtering

Human beings have a remarkable ability to interpret complex scenes in real time. The vision processes appear to select a subset of the available information for reducing complexity before further processing. This pre-process step helps in extracting the saliency regions from an unknown background and is extremely fast. Although it is still an open question how humans achieve this, detecting saliency based on some existing probabilistic models trained by certain machine learning techniques would not be the answer. For example, if a cat is considered the OOI in one image, a corresponding cat model could be trained so that we can detect it. This raises a question: What if we have a cat, a person, and a car in the same image? Are we going to use three different models to check the image separately and output the salient region? Facing unpredictable and innumerable categories of visual objects appearing in photos and videos, a general purpose saliency region detection system is required. The saliency region detection task, different from object detection and localization, attempts to imitate the human visual system for locating interest regions in images. It can be considered as a pre-process and complexity reduction step for more complex vision tasks.

As the basis of our proposed saliency detection algorithm, color difference, edge responses, and *key feature points*, or *keypoints* for short in our paper, play a central role. This algorithm involves using a number of keypoints to create a support region for the area of interest. We know that for each image, depending on its size, we may find more than one thousand keypoints. This is because new digital cameras tend to generate (very) high resolution images. The key question is how to select the representative keypoints while discarding all irrelevant keypoints. To tackle this problem, a new saliency map, *color-saliency weighted Probability-of-Boundary* (cPoB) confidence map is deployed. Our cPoB is built based on two types of features: (1) Color difference is used as an important cue for saliency detection. If the color of the object is significantly different from its surroundings, i.e., the background, we can get a response, representing the object, in our cPoB map. (2) Edge information is another cue for saliency detection because it represents the boundary of an object. We extract local features from each image and with the help of boundary information, we can focus more on keypoints that are on the contour of the object only. Notice that our cPoB is used for extracted SURF (Speeded Up Robust Features) [6] keypoint filtering, another emerging local feature descriptor for feature selection. The cPoB is not used directly to determine the interest region since the filtering step is used for selecting SURF keypoints that are located at the high response areas of our cPoB. Remaining collection of SURF keypoints is the final support region that we need.

For color saliency map, we want to find the most promising color features that best discriminate object classes from background ones as in [11]. Let $I$ be an image indexed by the location vector $l \in L \subset \Re^2$. We define two windows: a neighborhood $\mathcal{W}_l^{+1}$ denoted as the *inner* window, and a surrounding window $\mathcal{W}_l^{-1}$ denoted as the *outer* window. The union of the two

[1]The term "tracking" here refers to one of the traditional camera motions in filming, whereas in other parts of this paper it refers to the action of following moving OOI as it is generally used in Computer Vision literatures.

windows is denoted as the *total* window, $\mathcal{W}_l = \mathcal{W}_l^{-1} \cup \mathcal{W}_l^{+1}$. Figure 3a shows an example. Let $\mathbf{y}$ be the vector of feature responses, e.g., in the color space, at location $l \in L$. We intend to label every pixel, positioned at location $l$, within the area of interest with probability density $p(\mathbf{y}|+1)$ of being label $C(l) = +1$, and probability density $p(\mathbf{y}|-1)$ of being label $C(l) = -1$. We hope that pixels inside the inner window would have higher probabilities $p(\mathbf{y}|+1)$ than $p(\mathbf{y}|-1)$ and vice versa. Our feature pool $\mathbf{y}$ is of three channels of RGB pixel values. We use histograms of raw pixel values within this local area $\mathcal{W}_l$ for appearance representation. This is computed efficiently by integral images [37] and distributive histogram [34]. We normalize them to get probability density functions (pdfs). In our work, we discretized raw RGB values, ranging from 0 to 255, into 100 buckets. This discretization is performed for efficiency. Those pdfs are quantified by the mutual information [9] between features $\mathbf{Y}$ and class label $C$ over three scales.

$$S(l) = I_l(\mathbf{Y}; C) = \sum_c \int p(\mathbf{y}, c) \log \frac{p(\mathbf{y}, c)}{p(\mathbf{y})\,p(c)} d\mathbf{y} \tag{1}$$

which can also be written as

$$S(l) = \sum_c p(c) \mathrm{KL}(p(\mathbf{y}|c)||p(\mathbf{y})) \tag{2}$$

where $\mathrm{KL}(p(\mathbf{y}|c)||p(\mathbf{y}))$ represents the Kullback–Leibler (KL) divergence [27]. In information theory, the mutual information of two random variables is a quantity that measures the mutual dependence of the two variables. Hence, a large $S(l)$ implies that the inner and outer windows have a large difference of feature responses, i.e., large local feature contrast [27]. We multiply the color saliency maps found in each scale, similar to the binary OR operation, to capture the maximum responses from every level. Figure 3b shows an example of color saliency map. The procedure is summarized in Algorithm 1.

We adopt the Probability-of-Boundary (PoB) operator proposed in [28] which combines local brightness gradient (BG), color gradient (CG) and texture gradient (TG). PoB has been shown to outperform gradient-based approaches at boundary detection because a classifier is trained [28] using human labeled images as ground truth to combine the local feature information in an optimal way. The output of this classifier provides the posterior probability of a boundary at each image location and orientation.

In our experiments, we use BG and TG for detecting soft edges in one image because color information will be combined later via color saliency map. Subsequently, the boundary contrast map, shown in Fig. 3c,

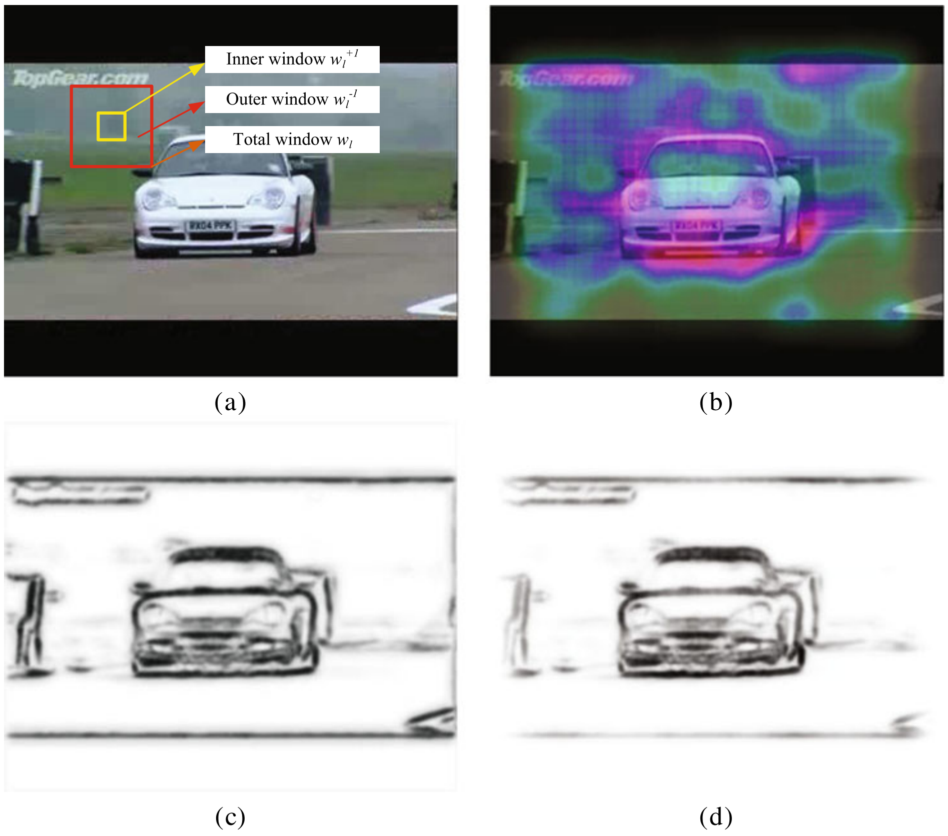


**Figure 3** **a** An example of inner, outer, and total window. **b** Using different color distributions to get color saliency map. **c** The soft boundaries detected by Probability-of-Boundary (PoB) operator. **d** Combining color saliency map and edge map for cPoB confidence map.

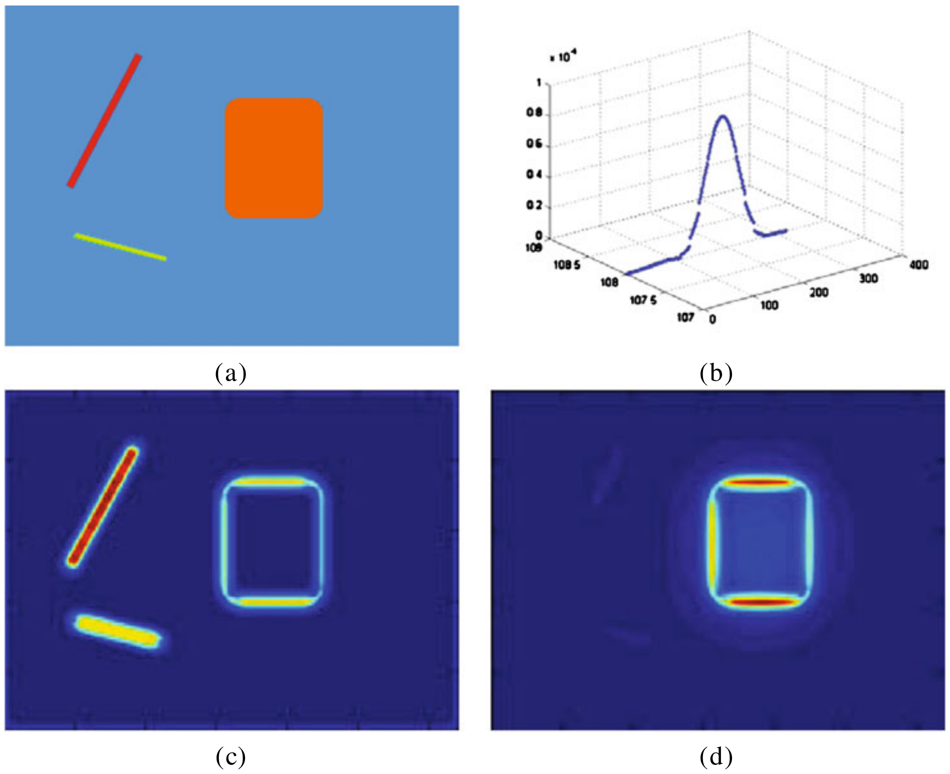


**Figure 4** **a** One simple example shows the effectiveness of our color saliency map. **b** Color saliency response at scan line 108, **c** soft edge map, **d** cPoB confidence map. All results are shown in heat map, i.e., color indicates the intensity of responses. *Blue* means a low or no response while *red* indicates a high value.

is weighted by color saliency map for cPoB confidence map, in Fig. 3d, which enables us to filter keypoints to select more important ones. The combination method used here is pixel by pixel multiplication between two maps. The intuition behind it is that some irrelevant or even unnecessary edges responses detected by PoB should be down weighted.

**Algorithm 1** Color Salience Map

```
1:  Input: Given image $I$ , initial inner window size $w_i$
    and outer window size $w_o$, and scale count $s_{\text{count}}$,
    e.g., 3.
2:  Output: Color saliency map $S(l)$.
3:
4:  for $s = 1$ to $s_{\text{count}}$ do
5:      $w_i = w_i \times \sqrt{2}$ and $w_o = w_o \times \sqrt{2}$.
6:      Identify inner window $\mathcal{W}_l^{+1}$ with width $w_i$, and
        outer window $\mathcal{W}_l^{-1}$ with width $w_o$.
7:      Create color histograms and then normalize
        them for inner $p(\mathbf{y}|+1)$, outer $p(\mathbf{y}|-1)$, and
        total $p(\mathbf{y})$ probability densities.
8:      Compute the mutual information $S(l)$ between
        $p(\mathbf{y}|+1)$, $p(\mathbf{y})$ and $p(\mathbf{y}|-1)$, $p(\mathbf{y})$.
9:  end for
10: Sum over all scales and re-normalize color salience
    map $S(l)$.
11: return $S(l)$
```

In Fig. 4, we demonstrate a simple example to show the effectiveness of our color saliency map. Provided the orange square as the OOI in Fig. 4a while two line segments as distracters, PoB will detect the edge of the square as well as two line segments as shown in Fig. 4c. However, the responses from the line segments are unnecessary, and sometimes it is even harmful for keypoint filtering because keypoints detected on the line segments would be included in our support region. By multiplying the color saliency map, the final cPoB is shown in Fig. 4d. Responses on two line segments are suppressed and edge responses from the object stand out.

## 3 Object Tracking

Automatic OOI detection and tracking in actively acquired videos are two problems considered in this section. Objects as well as background may contain certain appearance changes which makes it difficult to determine OOI by motion cues only. In addition, as introduced in Section 2.1, actively acquired videos, or Active Video, may contain one or several different types of camera movements. Thus, we deploy the OOI detection algorithm discussed in Section 2.2. The proposed method offers two advantages. (1) The system can determine OOI by itself, i.e., self-initialization, and track it in successive frames. This is one significant difference between other existing tracking methods like [2, 11] which need initial user interaction in the first (several) frame(s). (2) The appearance change in OOI is captured by the Successive Classification and Refinement (SCR) procedure because classification and matching is performed over the whole video sequence.

### 3.1 Self Initialization

We refer most of the existing tracking systems that require human assistance in initial labeling as *human-aided trackers*. Such kind of trackers have been well studied. Major research interests are focusing on how to develop a more robust algorithm to tackle the problems, such as color/shape change, occlusion, or undesired noises. However, in the self-initialization part, we mainly focus our attention in the first several frames of video. How to detect OOI and initialize tracking automatically? For static cameras, background subtraction algorithm [41] gives us satisfactory results in extracting moving objects. However, the result will be quite different for a moving camera. Both foreground and background objects possess certain movements in the resulting video frames. The background subtraction algorithms may not help and it is very likely to create noisy responses.

Borrowing the idea from Section 2.2, OOIs should be within or form a color salient region to capture human attentions. We use a group of SURF keypoints as our OOI representation. The major problem is how to distinguish the foreground keypoints from the background ones. Here, the important *object-centered* property of *Active Video* reviewed in Section 2.1 plays a central role. OOI is caught by the video camera operators by prominent catching and holding behaviors. A *data collection* procedure is deployed throughout the whole video sequence. The data collection step is done somewhat differently in the *initial* frames and the *tracking* frames.

During the initial frames, foreground and background keypoints are distinguished only by the cPoB confidence map. It is observed that the cPoB results are quite noisy. However, we are still able to hypothesize that most of the select ones belonging to OOI because of the object-centered property. We train a "strong classifier" based on the keypoints collected in the initial stage and use it for classification during the tracking. The data collection procedure continues operating while throwing away obsolete keypoints to capture changes in object appearance. During tracking, we have one more tool to tell the foreground keypoints from background ones. Linear Programming (LP) is used to find maximum similarities between two sets of keypoints from consecutive frames. This can be consider as an outlier rejection process. Therefore, we coined the acronym SCR for *Successive Classification and Refinement* and it will be discussed in detail in the following section.

### 3.2 Successive Classification and Refinement (SCR)

Objects in videos often follow a smooth trajectory, which is usually referred to as the temporal smoothness property. In particular, the object-centered property of Active Video makes the trajectory even smoother. In the phase of holding or smooth pursuit, the OOI actually tends to show little motion, staying at the center of the video frame. This is an important and useful property in that keypoints from the background will appear and disappear because of the background change caused by changing occlusion and/or camera movements. However, keypoints from the object tend to be more stable. Similar to the Earth Mover's Distance (EMD) [32], we formulate the keypoints matching between two consecutive frames as an LP problem.

Given two consecutive frames $I_1$ and $I_2$, let $P = \{\mathbf{p_1}, \ldots, \mathbf{p_m}\}$ and $Q = \{\mathbf{q_1}, \ldots, \mathbf{q_n}\}$ be the collection of keypoints extracted from $I_1$ and $I_2$, respectively; and $K = [k_{ij}]$ be the similarity matrix where $k_{ij}$ is the similarity between keypoints $\mathbf{p_i}$ and $\mathbf{q_j}$ with $\chi^2$ distance measurement. We define their maximum keypoint similarity as follows.

$$S(P, Q) = \max_{\alpha} \sum_{i,j} \alpha_{ij} k_{ij} \tag{3}$$

$$\text{s.t.} \quad \forall i,\ j,\ \sum_{i} \alpha_{ij} \leq 1,\ \sum_{j} \alpha_{ij} \leq 1,\ 0 \leq \alpha_{ij} \leq 1.$$

where $\alpha_{ij} \in \alpha$ denotes the flow value between keypoint $\mathbf{p_i}$ and $\mathbf{q_j}$ indicating how likely they can be matched so that $S(P, Q)$ is maximized. LP approach presents a general framework for global optimization. Even when the number of keypoints to be matched is large, LP can still be solved in polynomial time.

When we get the matched pair of keypoints between two frames by LP, it is very likely that some false matching has occurred and background keypoints are also included. Displacement clustering is designed to reject false matching. For the matched keypoints, we calculate the displacement in $x$ and $y$ directions for each pair. This displacement information helps us to cluster keypoints into two groups. One group is the keypoints which possess similar displacement for OOI and another group is the outliers which is caused by false matching. We want to keep the former group and

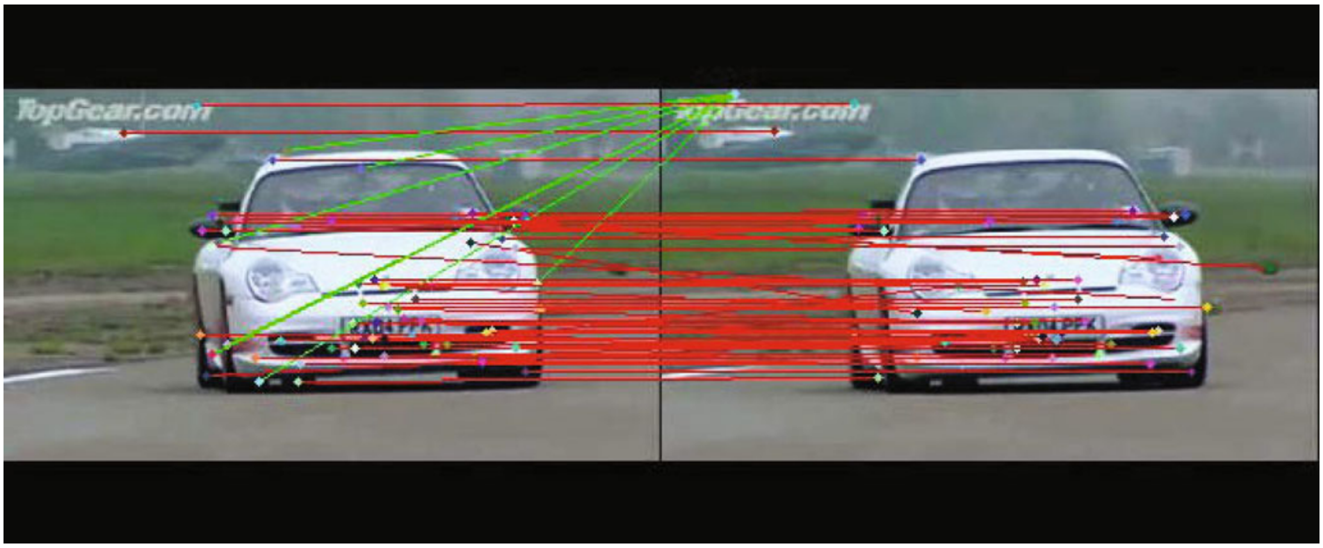


**Figure 5** Demonstration of LP soft matching between two frames. Matching results are shown in *red* and *green lines*. *Red lines* indicate the larger group of keypoint pairs that we want to keep and green ones show the false matching. In our experiments, we use K-means as our clustering algorithm. Notice that all keypoints shown here are filtered ones, which is discussed in Section 2.2.

eliminate the latter. Figure 5 shows one example. Red lines indicate the larger group of keypoint pairs that we want to keep and green ones show the false matching.

After keypoints are filtered by cPoB and matched to the next frame via LP, we consider that the support region created by remaining ones approximates the position of OOI in each frame, shown in Fig. 6a. A convex hull is found from them using [5] (Fig. 6b). *Candidate* keypoints within the polygonal region are assumed to belong to the object (positive examples), while others are assumed to belong to the background (negative examples).

Tracking is now treated as a binary classification problem as is also addressed by Avidan [2] and Comaniciu [13]. The strong classifier, calculated using AdaBoost, short for "Adaptive Boosting" [15], is then used to classify the keypoints in the next frame. Boosting can give good results even if the weak classifiers

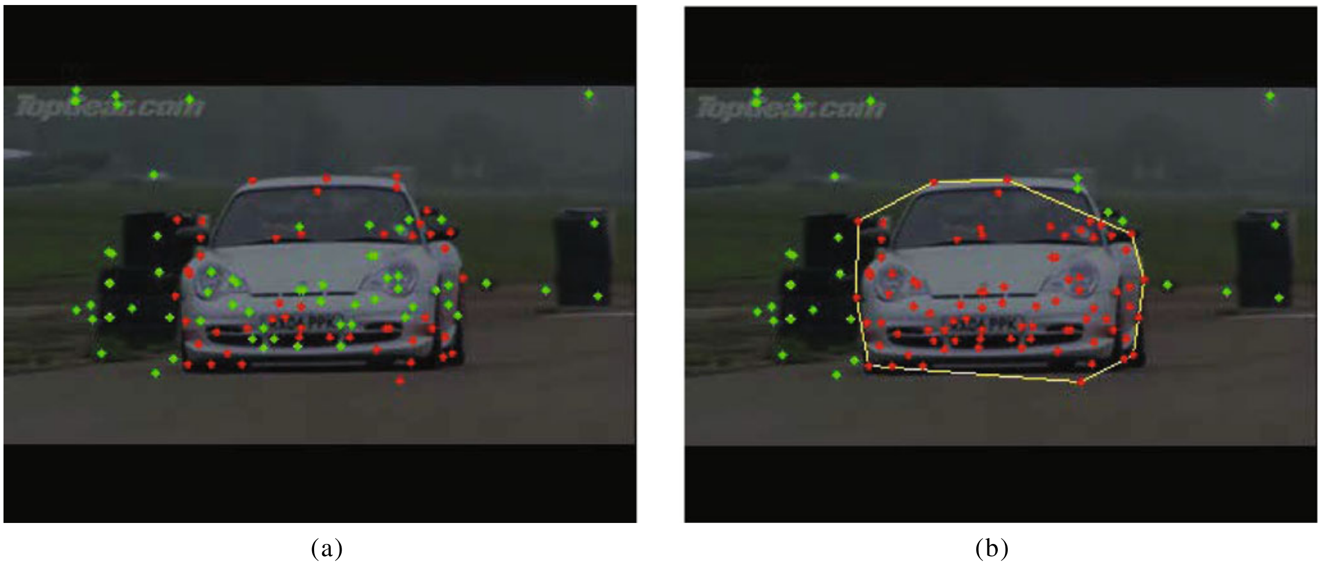


**Figure 6** **a** *Red candidate keypoints* are used to approximate the position of OOI in each frame. **b** A convex hull, in *yellow*, is found and all inside keypoints are labeled as positive examples (*red ones*) while others are negative examples (*green ones*).

have a performance that is only slightly better than random. In our experiments, we use Real AdaBoost from GML AdaBoost MATLAB Toolbox,[2] which uses a classification tree as a weak learner. The details of the Automatic OOI Detection and Tracking algorithm are given in Algorithm 2.

**Algorithm 2** Automatic OOI Detection and Tracking

```
1: Input: $n$ video frames $I_1, \dots, I_n$, number of frames for OOI detection ciDet, and number of frames for dataset collection ciTr.
2: Output: OOI confidence map $r_1, \dots, r_{n-1}$.
3: Initialize: Positive dataset PosSet $= \emptyset$ and negative dataset NegSet $= \emptyset$.
4:
5: for iFrm = 1 to iTotalFrame - 1 do
6:     Extract $P = \{\mathbf{p_i}\}_{i=1}^m$, $Q = \{\mathbf{q_j}\}_{j=1}^n$ keypoints from $I_{\mathtt{iFrm}}$ and $I_{\mathtt{iFrm+1}}$.
7:     Calculate cPoB confidence map for $I_{\mathtt{iFrm}}$ and $I_{\mathtt{iFrm+1}}$ respectively.
8:     Filter keypoints $P$, $Q$ using cPoB and get *potential* dataset $P_{pot} \subseteq P$, $Q_{pot} \subseteq Q$.
9:     if iFrm > ciDet then
10:        $P_{pot} =$ Classify($P$) $\cap P_{pot}$.
11:    end if
12:    Do LPMatching($P_{pot}$, $Q$) and outliers rejection procedure to get $P'_{pot} \subseteq P_{pot}$
13:    Find convex hull from $P'_{pot}$.
14:    Keypoints inside the polygonal region are *candidate* dataset $P_{can} \subseteq P_{pot}$ and severed as positive dataset $P^+$.
15:    Negative dataset is the remaining keypints $P^- = P - P^+$.
16:    PosSet = PosSet $\cup P^+$.
17:    NegSet = NegSet $\cup P^-$.
18:    Generate OOI confidence map $r_{\mathtt{iFrm}}$ from $P^+$.
19:    if !Mod(iFrm, ciTr) then
20:        Do AdaBoost(PosSet, NegSet) for a new strong classifier $Y_M(\mathbf{x})$.
21:    end if
22: end for
```

Intuitively, this automatic OOI detection and tracking algorithm works in two phases. (1) The OOI detection or self-data collection step is performed at the very beginning of each video shot for $n$ frames, e.g., first 30 frames. In this step, OOI candidate keypoints are selected using the cPoB confidence map from each frame and collected into a positive keypoint pool. Meanwhile, all other keypoints are considered as negative and kept in a negative pool. (2) After the first $n$ frames, we proceed with the SCR step. In the successive classification phase, a strong classifier is trained using AdaBoost by the positive and negative datasets and it is used in the subsequent classification step. Refinement is finished using LP feature matching to maximize the similarity between two sets of keypoints. In order to capture the appearance change of OOI, we continue collecting positive and negative datasets while throwing away obsolete ones. For a larger amount of $m$ frames, e.g., after 150 frames, a new strong classifier is trained again for later use.

[2]We use the MATLAB code from http://research.graphicon.ru/machine-learning/gml-adaboost-matlab-toolbox.html.

### 3.3 Global Motion Compensation

Global Motion (GM) compensation [16] is another useful tool for foreground and background keypoints selection. When smooth pursuit is undertaken by the operator, the OOI is centered in *Active Video* and tends to not move. This will result in (very) small magnitudes of motion vectors. However, motion vectors belonging to the background are different from OOI and their magnitudes are relatively large. We use a relatively new diamond search algorithm proposed in [40] for fast block-matching motion estimation. The global motion compensation is performed for cleaning remaining keypoints which are moving along the same direction of the global motion vector. That is, the reverse direction of the global motion vector is added to all motion vectors and only keypoints with high magnitude of motion vectors are kept.

## 4 Experimental Results

We tested our algorithm on three video clips which contain one or several different types of camera movements as described in Section 2.1. Video sequences are from car racing and sports games with well defined OOIs, and they range from 10 to 20 s in length. Each video is at 25 frames per second and the frames have visible compression artifacts. For all the video clips, first 25 frames, i.e., 1 s, are used for self-initialization. The training interval is set to 125 frames, that is, every 5 s to do AdaBoost again for a new strong classifier.

### 4.1 General Performance

*Car sequence* In this clip, one white car is detected and tracked with a dynamic moving background. This low quality video contains 428 frames and are of the resolution 352 × 288. Subject being tracked has scale change, shot cut, and partial occlusion. Tracking results are shown in the top row of Fig. 7. From the result we can see that our proposed algorithm can handle view point and scale changes pretty well. The reason is that we focus on the group of keypoints belonging to OOI in our algorithm and the distribution of them will vary naturally according to the view point or scale change of the subject. Also, with the help of the strong classifier trained via AdaBoost, our algorithm is robust to shot cut and occlusion. It can re-detect and locate the subject again.

*Skate sequence* This is a higher resolution clip, 720 × 240, with about 200 frames. Same as the "car" sequence, only one subject is detected and tracked in this clip. However, one of the noticeable differences is that the athlete we tracked is rich in motions and shape changes.

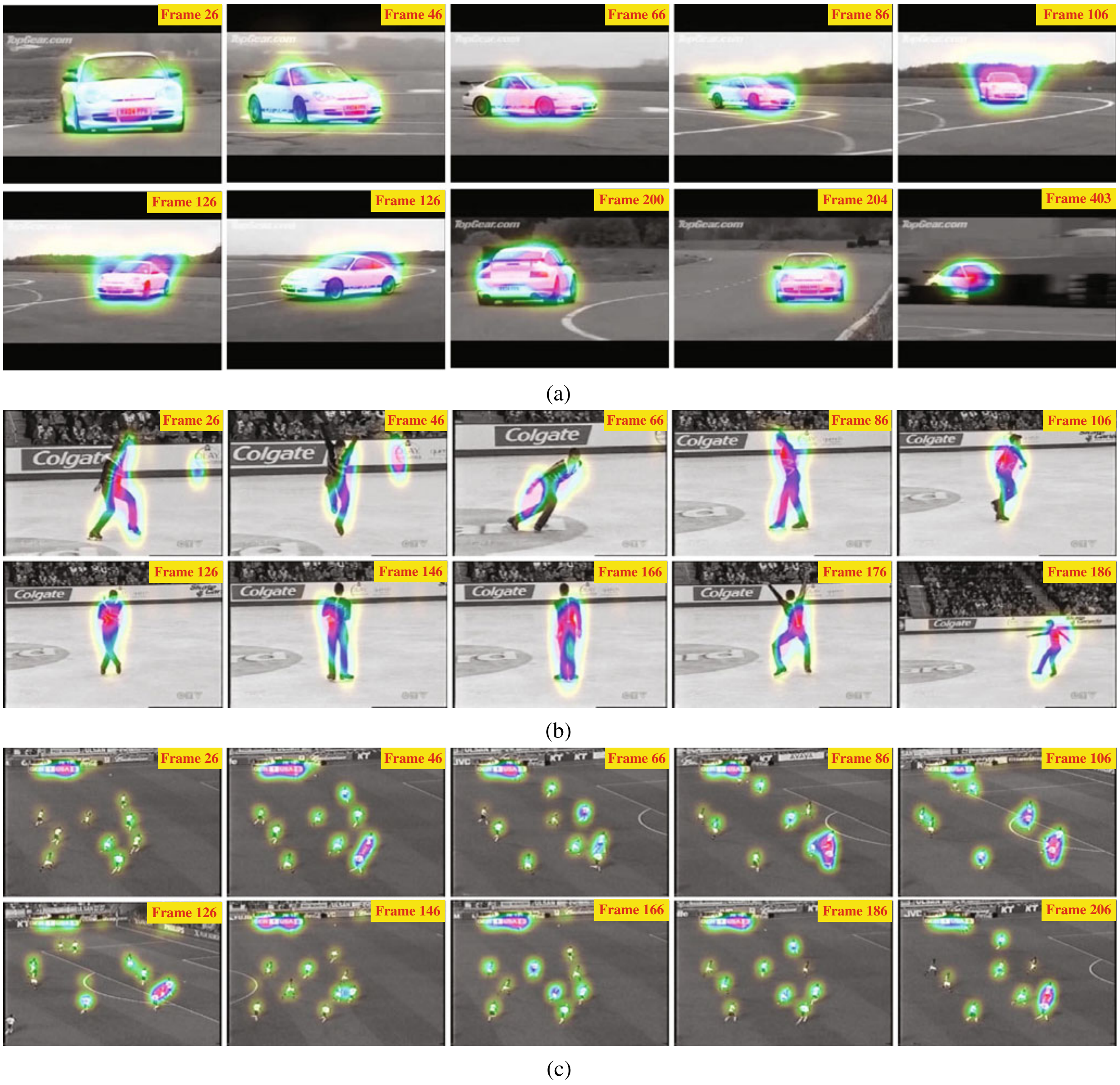


**Figure 7** Tracking results for **a** "car", **b** "skate", and **c** "soccer" video clips. For each frame, a probability map is shown overlaying on it which shows the position of the OOI.

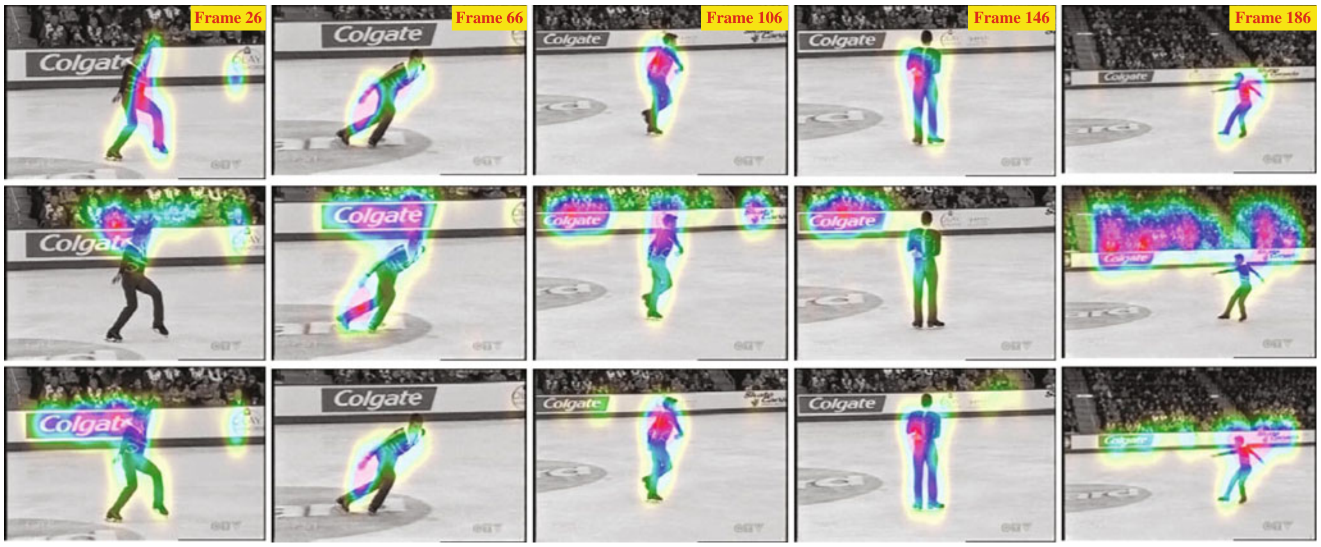


**Figure 8** Comparative tracking results for "skate" sequence by our proposed algorithm (*top row*), without cPoB keypoints filtering (*middle row*), and without keypoints classification (*bottom row*).

Moreover, the audience section as the background causes an additional problem because corners or T-junctions are captured by the SURF keypoints detector as well. To tackle this problem, cPoB is very useful to filter out those useless keypoints because audience section is relatively textureless comparing to the foreground, i.e., the athlete on the ice. Sample results are shown in the middle row of Fig. 7.

*Soccer sequence* The cameraman did the OOI "tracking" in this clip. It has about three hundred 720 × 480 frames. This video contains multiple OOIs, i.e., the players, although it is a very simple case because all objects are actually moving in more or less the same direction. Our algorithm can detect and track meaningful OOIs successfully, shown in the bottom row of Fig. 7. In general, tracking multiple objects in *Active Video* will involve more sophisticated mechanisms. While humans will likely focus our attention to where the ball or the key player is, we do have our peripheral vision that enables the coverage of a large visual field, and alerts us when a shift of attention is needed, e.g., when the ball is passed from one player to the other. To emulate these in a machine vision system, more advanced algorithms will need to be developed.

## 4.2 Comparisons

Based on the above discussions, we can see that cPoB confidence map for keypoint filtering and keypoint classification by the strong classifier are the two important steps in our algorithm. Indeed, they are useful and

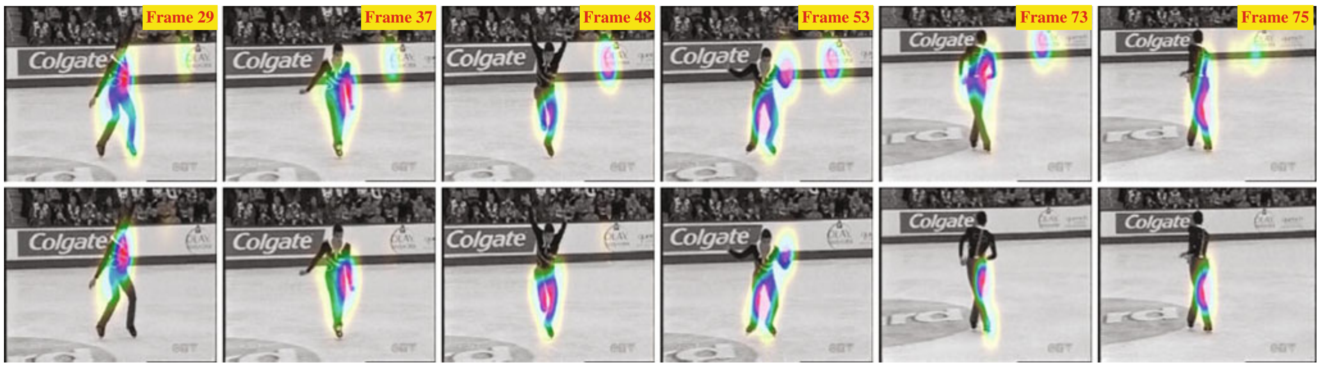


**Figure 9** Comparative tracking results for "skate" sequence by keypoint filtering and classification only (*top row*), and with additional global motion cleaning (*bottom row*).

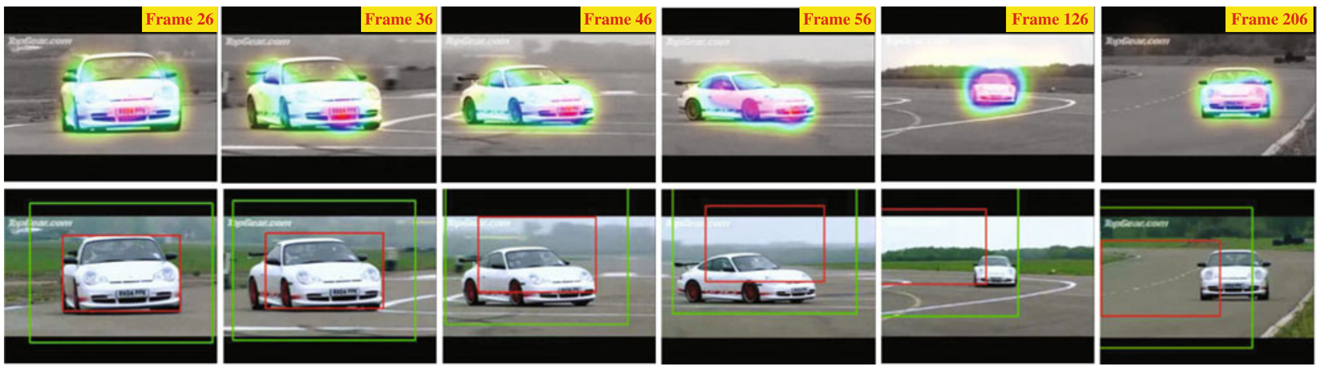


**Figure 10** Comparative tracking results for "car" sequence by our proposed algorithm (*top row*) and online feature selection (OFS) tracker [11] (*bottom row*). For frame 26, our algorithm automatically detects OOI and starts tracking, while in the *bottom row* the *red rectangle* is labeled by user as the input for OFS tracker.

crucial which can be verified from the following two experiments.

*Importance of cPoB keypoint filtering* cPoB is used in our algorithm for giving confidence score to each keypoint belonging to OOI or background. When this step is removed, the only way to tell whether a keypoint is filtered out or not is by LP soft matching. Those matched keypoints are selected and the polygonal region is found based on them. This will introduce some problems because the background keypoints will also be "correctly" matched between two consecutive frames. In the data collection phase, our algorithm will collect keypoints from OOI as well as those from the background, which results in misclassification. We can see from the middle row of Fig. 8 where background has higher responses than foreground OOI.

*Effectiveness of keypoint classification* We get better results than those without cPoB, however, some false responses are also generated from the background as shown in the bottom row of Fig. 8. This is because color difference or edge information will give a high response in our cPoB confidence map. Keypoints will not be filtered out by cPoB as long as they possess either or both properties. It is the strong classifier trained using AdaBoost that tells us if it is an OOI keypoint or not. Without classification, background keypoints pop up.

*Feasibility of GM keypoint cleaning* In the "skate" video sequence, the advertisement around the playground pops up sometimes as shown in the top row of Fig. 9. The reason is that advertisement is usually rich in color and have strong edges. Our cPoB will not filter out those keypoints due to high response in the confidence map. By using the Global Motion keypoint cleaning, we keep keypoints with high magnitude of motion vectors. Results are shown in the bottom row of Fig. 9.

In addition, we compare our results on "car" sequence to the Online Feature Selection (OFS) tracker[3] by Collins and Liu [11]. OOI is labeled by user at frame 26 because our tracker also starts to track at that time after the first 25 frames for data collection and learning. Sample results are shown in Fig. 10. We can see that after tracking for about 20 frames, OFS starts to drift away from the OOI because the change in scale of OOI will introduce more background pixels to the inner window (red rectangle). This contaminates the foreground color histogram with background pixel values and reduces the discriminability of the two-classes variance ratio measurement. Besides, OFS cannot handle sudden frame changes caused by the shot cut.

## 5 Conclusion

We proposed an automatic object of interest (OOI) detection and tracking algorithm. Unlike most existing tracking algorithms, this method requires no initial user labeling and it is able to detect moving subjects automatically. The only constraint is that in the data collection procedure, OOIs should be the major part within the first several frames. In Active Video, this is usually achieved after the initial catching and during the subsequent holding actions. In this paper, We introduced a *color-saliency weighted Probability-of-Boundary* (cPoB) map for salient region detection and *Successive Classification and Refinement* (SCR) for tracking. Keypoints classification is done by

[3]We implement Online Feature Selection (OFS) tracker in MATLAB.

training a strong classifier using AdaBoost. The temporal smoothness property of the videos helps Linear Programming to reject outliers. We tested our algorithm on three actively acquired videos and performed comparative experiments in Section 4.2. Experiments confirm that the proposed scheme can automatically detect and reliably track OOIs in Active Video.

## 6 Future Work

In addition to the tracking algorithm, one interesting question we may ask is whether our cPoB and SCR can be applied to the following new problems.

(1) *Image thumbnail*. "Thumbnail" is the term used to describe a miniature version of a picture. We use image thumbnails everyday in modern operating systems. However, current image thumbnails are only a scaled down version of larger images. Many important details are lost during the procedure. This is a well defined problem and a popular topic in multimedia research. Our cPoB is capable of finding OOI within an image, which can be considered as a preliminary step of image thumbtacking.

(2) *Video summarization*. Video summarization is a method for removing irrelevant frames from a video and outputting a smaller number of frames containing the OOI. In our SCR, the self-initialized OOI can be reliably tracked throughout the video and it is robust to viewpoint changes and OOI scaling. The object model built in the first several frames are useful in the successive tracking. Is the model effective in extracting relevant frames in the whole video? This can be another interesting topic for future exploration.

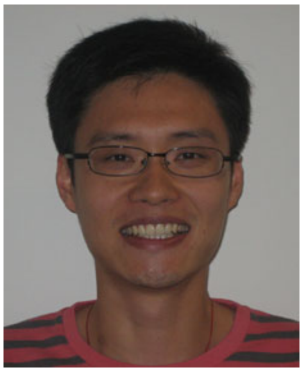
**Acknowledgements** This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under the grant RGP36726.

## References

1. Avidan, S. (2004). Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 26*(8), 1064–1072.
2. Avidan, S. (2005). Ensemble tracking. In *Proceedings of computer vision and pattern recognition* (pp. 494–501).
3. Avidan, S., & Shamir, A. (2007). Seam carving for content-aware image resizing. *ACM Transactions on Graphics, 26*(3). doi:10.1145/1276377.1276390.
4. Baker, S., & Matthews, I. (2004). Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision, 56*(1), 221–255.
5. Barber, C. B., Dobkin, D. P., & Huhdanpaa, H. (1995). The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software, 22*, 469–483.
6. Bay, H., Tuytelaars, T., & Gool, L. V. (2006). Surf: Speeded up robust features. In *Proceedings of European conference on computer vision* (pp. 404–417).
7. Carpenter, R. (1977). *Movements of the eyes*. London: Pion.
8. Cheng, Y. (1995). Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 17*, 790–799.
9. Chouinard, J. Y., Fortier, P., Gulliver, T. A. (Eds.) (1996). *Information theory and applications II, 4th Canadian workshop*. Lac Delage, Québec, Canada, May 28–30, 1995. *Selected papers. Lecture notes in computer science* (Vol. 1133). Springer.
10. Collins, R. T. (2003). Mean-shift blob tracking through scale space. In *Proceedings of computer vision and pattern recognition*.
11. Collins, R. T., & Liu, Y. (2003). On-line selection of discriminative tracking features. In *Proceedings of international conference on computer vision* (pp. 346–352).
12. Comaniciu, D., & Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*, 603–619.
13. Comaniciu, D., Ramesh, V., & Meer, P. (2003). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 25*, 564–577.
14. Enkelmann, W. (2001). Video-based driver assistance: From basic functions to applications. *International Journal of Computer Vision, 45*(3), 201–221.
15. Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences, 55*(1), 119–139.
16. Ghanbari, M. (1999). Video coding: An introduction to standard codecs. Stevenage: Institution of Electrical Engineers.
17. Huang, J., & Li, Z. N. (2009). Automatic detection of object of interest and tracking in active video. In *Proceedings of Pacific rim conference on multimedia* (pp. 368–380).
18. Huang, J., & Li, Z. N. (2009). Image trimming via saliency region detection and iterative feature matching. In *Proceedings of international conference on multimedia expo* (pp. 1322–1325).
19. Intille, S. S., Davis, J. W., & Bobick, A. F. (1997). Real-time closed-world tracking. In *Proceedings of computer vision and pattern recognition* (pp. 697–703).
20. Itti, L., & Koch, C. (1999). A comparison of feature combination strategies for saliency-based visual attention systems. In *Proceedings of SPIE. Human vision and electronic imaging IV. (HVEI'99)* (Vol. 3644, pp. 473–482). San Jose: SPIE.
21. Itti, L., Koch, C., & Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 20*(11), 1254–1259.
22. Julesz, B. (1995). *Dialogues on perception*. Cambridge: MIT Press.
23. Kadir, T., & Brady, M. (2001). Saliency, scale and image description. *International Journal of Computer Vision, 45*(2), 83–105.
24. Kim, Z. (2008). Real time object tracking based on dynamic feature grouping with background subtraction. In *Proceedings of computer vision and pattern recognition* (pp. 1–8).
25. Liu, D., Hua, G., & Chen, T. (2008). Videocut: Removing irrelevant frames by discovering the object of interest. In

*Proceedings of European conference on computer vision* (Vol. I, pp. 441–453).

26. Lu, Y., & Li, Z. N. (2008). Automatic object extraction and reconstruction in active video. *Pattern Recognition, 41*(3), 1159–1172.
27. Mahadevan, V., & Vasconcelos, N. (2008). Background subtraction in highly dynamic scenes. In *Proceedings of computer vision and pattern recognition* (pp. 1–8).
28. Martin, D. R., Fowlkes, C. C., & Malik, J. (2004). Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 26*(5), 530–549.
29. Niebur, E., & Koch, C. (1998). Computational architectures for attention. In R. Parasuraman (Ed.), *The attentive brain* (pp. 163–186). MIT Press.
30. Rother, C., Bordeaux, L., Hamadi, Y., Blake, A. (2006). Autocollage. *ACM Transactions on Graphics, 25*(3), 847–852.
31. Rother, C., Kumar, S., Kolmogorov, V., & Blake, A. (2005). Digital tapestry. In *Proceedings of computer vision and pattern recognition* (pp. 589–596).
32. Rubner, Y., Tomasi, C., & Guibas, L. J. (2000). The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision, 40*(2), 99–121.
33. Simakov, D., Caspi, Y., Shechtman, E., & Irani, M. (2008). Summarizing visual data using bidirectional similarity. In *Proceedings of computer vision and pattern recognition* (pp. 1–8).
34. Sizintsev, M., Derpanis, K. G., & Hogue, A. (2008). Histogram-based search: A comparative study. In *Proceedings of computer vision and pattern recognition* (pp. 1–8).
35. Stauffer, C., Eric, W., & Grimson, W. E. L. (2000). Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22*, 747–757.
36. Tsotsos, J. K., Culhane, S. M., Winky, W. Y. K., Lai, Y., Davis, N., & Nuflo, F. (1995). Modeling visual attention via selective tuning. *Artificial Intelligence, 78*(1–2), 507–545.
37. Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of computer vision and pattern recognition* (Vol. I, pp. 511–518).
38. Yin, Z., & Collins, R. T. (2008). Object tracking and detection after occlusion via numerical hybird local and global mode-seeking. In *Proceedings of computer vision and pattern recognition* (pp. 1–8).
39. You, W., Jiang, H., & Li, Z. N. (2008). Real-time multiple object tracking in smart environments. In *Proceedings of international conference on robotics and biomimetics* (pp. 818–823).
40. Zhu, S., & Ma, K. K. (2000). A new diamond search algorithm for fast block-matching motion estimation. *IEEE Transactions on Image Processing, 9*(2), 287–290.
41. Zivkovic, Z. (2004). Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of international conference on pattern recognition* (Vol. 2, pp. 28–31).



**Jiawei Huang** received the B.Eng. degree in electrical engineering from Zhejiang University, Hangzhou, China in 2006 and the M.Sc. degree in computing science from Simon Fraser University, Vancouver, British Columbia, Canada, in 2009.

He is currently a research assistant at the School of Computing Science, Simon Fraser University, and the video technology group of Scientific Atlanta, Vancouver, British Columbia, Canada. His research interests include computer vision, image and video processing, multimedia, and machine learning.

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

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