The CSI Journal on
Computer Science and Engineering
Vol. 12, No. 2 & 4 (b), 2014
Pages 16-28
Regular Paper

Online Visual Object Tracking Using Incremental Discriminative Color Learning

Alireza Asvadi Hami Mahdavinataj Mohammadreza Karami Yassar Baleghi

Department of Electrical and Computer Engineering, Babol University of Technology, Babol, Iran

Abstract

This paper presents a method for tracking an object in a sequence of images given its location in the first frame. Recently, a class of techniques called discriminative methods has shown promising results. These methods are based on training a classifier to distinguish the object from surrounding background. However, discriminative methods do not explicitly model the object. Therefore, noisy samples are likely to interfere and cause visual drift. In this paper, 3D joint RGB histograms of the object and surrounding background are used to develop an object model. An incremental color learning scheme with a forgetting factor is applied to evolve the object model during tracking. It is shown the proposed method can handle visual drift effectively. Evaluated against five state of the art methods, experiments demonstrate superior results of the proposed tracking algorithm. Implemented in MATLAB, the algorithm runs at 17.2 frames per second, including image input/output time.

Keywords: Visual Object Tracking, 3D Joint RGB Histogram, Log-Likelihood Ratio, Incremental Learning, Mean-Shift Localization.

1. Introduction

Visual object tracking is one of the most active research areas in computer vision, with numerous applications, including automated surveillance, motion-based recognition, human-computer interaction and robotics. The goal of object tracking is to estimate the trajectory of an object in an image sequence given the initialized position in the first frame [1].

There is a rich literature in visual object tracking [1-4]. The object tracking algorithms mainly differ in the way they use image features and model the motion, appearance and shape of the object. Here, we focus on the most relevant online visual object tracking methods that operate directly on color images and also some state of the art methods. Generally, online object tracking algorithms can be categorized as either generative methods or discriminative methods [3].

The generative method builds a model to describe the appearance of an object and then finds the object by searching for the region most similar to the reference model in each frame. One of the most influential approaches in this category is the Mean-Shift Tracker [5] which uses a color histogram regularized by a spatially smooth isotropic kernel for object representation. Using the Bhattacharyya coefficient as a similarity metric, the mean-shift procedure is performed for object localization by finding the basin of attraction of the local maxima. In [6] it is shown that background-weighted histogram algorithm in the mean-shift tracking is not optimal. They assigned new weights to pixels in the target's candidate region to reduce background's interference in target localization. It leads to faster convergence and more accurate localization than the usual target representation in mean-shift tracking. One of the limitations of rectangular or ellipsoidal regions for object representation is that the object model encompasses many background pixels in addition to the true target pixels. Consequently, the resulting histogram can be corrupted by background pixels, and the tracking result degrades accordingly. To address this problem, a fragment-based appearance model has been proposed [7]. Multiple local histograms are used to represent an object. A vote map is

used to combine the votes from all the regions in the target frame. However, the computation of multiple local histograms and the vote map can be time consuming even with the use of integral histograms. In a similar vein, in [8], each target object is modeled by a small number of rectangular blocks, which their positions within the tracking window are determined adaptively. Recently, [9] proposed a histogram that takes into account contributions from pixels adaptively in a multi-region manner along with an illumination invariant feature. However, parts of the background may reside inside the object model. Therefore, the object model encompasses many background pixels in addition to the true target pixels which can degrade the object model. The advantage of generative methods is that it does not require a large dataset for training. However, the imperfection of the generative model is that it only focuses on the foreground without considering the background information.

Discriminative methods pose object tracking as a binary classification problem in which the task is to determine a decision boundary that distinguishes the object from the background without the need to a complex model characterizing the object. Since the discriminative model considers both the object and background information, it usually achieves better performance than the generative model. In [10] pixel's colors are mapped into 49 one-dimensional lines in RGB color space to build a set of 1D color histograms. Next, the log-likelihood ratio of a feature value is used to select discriminative color features for tracking. In a similar manner [11] and [12] adopt 13 different linear combinations of R, G, B pixel values to approximate 3D RGB color space. Approximating RGB color space using a set of 1D histograms is cheaper. However the major imperfection is it loses considerable color information that lies in 3D joint RGB histogram. In [13] a tracking-by-detection approach, called CSK tracker, is proposed. It is based on optimal selection of samples in target's neighborhood. They showed that the process of taking sub-windows of an image induces circulate structure. They used Fast Fourier Transform (FFT) to incorporate information from all sub-windows. Recently, in [14] the CSK tracker has been extended by incorporating color attributes and defining a kernel. In [15] a description-discrimination collaborative frame work has been proposed. The descriptive component forms a robust object model from all the tracked frames. Meanwhile, the discriminative component differentiates the targets from its surrounding background in recent frames. In [16] a deep learning tracker is proposed that works based on learning discriminative saliency map. They pre-trained Convolution al Neural Network (CNN) on a dataset offline. They employed outputs from the hidden layers of the network as feature descriptors of an object. The features are employed to learn discriminative target appearance models using an online Support Vector Machine (SVM).

The appearance of the object usually changes during the tracking. Therefore, it is inevitable to update the appearance of the object model during tracking. It has been shown that an adaptive appearance model is the key to a good performance, and much attention has been paid in recent years to address this issue. The most straightforward method is to replace the current appearance model (e.g. template, color) with the visual information from the most-recent tracking result [17]. However, simple update with recently obtained tracking result can lead to drift. In [18] Ad a Boost algorithm is proposed to classify pixels belonging to foreground and background and update weak classifiers with new ones to adapt to the changes of the object and background during the tracking. In [19] the Boosting method is extended to an online manner. In [20] an incremental subspace learning method is proposed, which presents an adaptive probabilistic tracking algorithm that updates the models using an incremental update of Eigen-basis. In [21] the object is represented by a set of Hear-like features that are computed for each image patch. They used Multiple Instance Learning (MIL) to handle ambiguously labeled positive and negative data obtained online to reduce visual drift caused by classifier update.

The discriminative methods generally do not model the object explicitly and only define the boundary between the target and background. Here we utilize a discriminative method to separate the object from surrounding background in each frame, and an incremental learning scheme is incorporated to learn and evolve the object color model during the tracking process [1]. In addition, the proposed method attempts to solve the unwanted effect of background pixels in the object rectangle.

The contributions of this paper are summarized as follows:

- A discriminative 3D joint RGB histogram based representation of the object.
- Incremental color tracking with a forgetting factor to account for appearance variation of the object.
- A scale adaptation scheme to compensate the scale and ratio variation of the object.

The remaining part of this paper is organized as follows. Section 2 describes the details of the proposed algorithm that presents an efficient incremental color object tracking. The results of the experiments and performance evaluations are presented in Section 3, and the conclusions and the future prospects are given in Section 4. The source code and videos corresponding to this work are available[2].

2. Proposed Method

Object tracking starts with selecting an object in the first frame approximately. A surrounding area is computed automatically such that the number of background pixels in the region surrounding the object is approximately the same as the number of pixels within the object region. Therefore, the width and height of outer rectangle are defined by $W = \sqrt{2} \times w$ and $H = \sqrt{2} \times h$. Where w and h are the width and height of the selected object region.

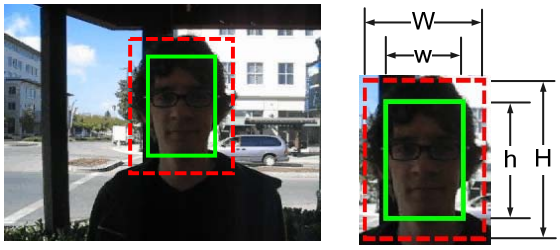


Figure 1. Object and background rectangles. The selected object region is inside the solid green rectangle and the surrounding background region is the area between the dashed red and solid green rectangles

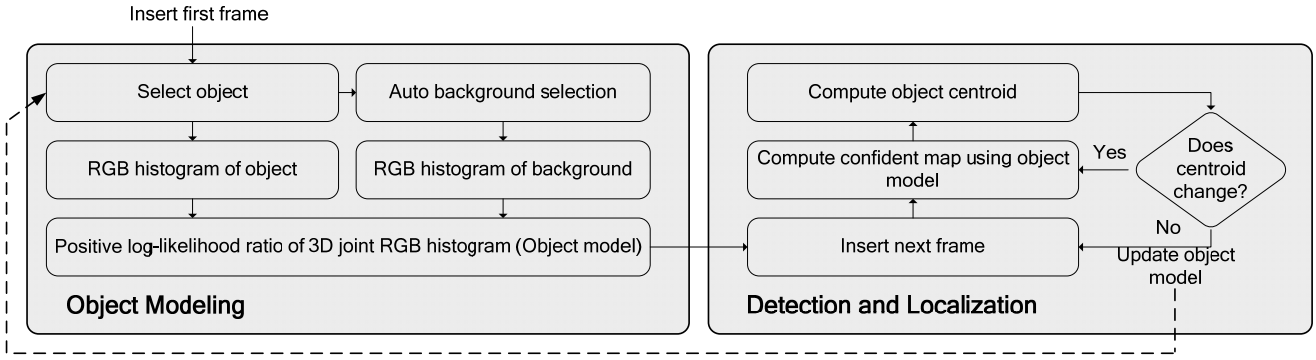


Figure 2. Architecture of the proposed object tracking algorithm

The selected object and background rectangles are shown in figure 1 In object tracking, first, one needs to develop a model for the object of interest.

The aim of the object model is to identify the object from the background in subsequent frames accurately. Then, the object localizer estimates the target location in the subsequent frames using the object model. Next, an efficient method that incrementally updates the object model is employed to learn the appearance of the target while tracking progresses. Architecture of the proposed tracking algorithm is shown in the figure 2 Each step is described in the following sections.

2.1. Object Modeling

The object model is built and evolved based on discriminative 3D joint RGB histogram of the object and background. The quantized 3D joint RGB histograms of the region within the inner rectangle and the region between the inner and outer rectangles are computed. Positive part of the log-likelihood ratio of the object region and background region surrounding the object is used to determine the object model using (1),

$$L_s = \max\left\{\ln \frac{\max\{H_o(s),\varepsilon\}}{\max\{H_b(s),\varepsilon\}}, 0\right\} \quad (1)$$

where $H_o(s)$ the histogram is computed within the object rectangle, and $H_b(s)$ is the histogram for the region surrounding the object. The object color seeds have higher values in the computed log-likelihood ratio while background color seeds have negative values and colors that are shared by both the object and background tend towards zero. To model the object individually, negative values are rejected. Here, 8 bins for each channel are used for histogram quantization. Therefore, the index s ranges from 1 to 8^3 and 8^3 is the total number of histogram seeds. ε Is a small non-zero value to avoid numerical instability that prevents dividing by zero or taking the logarithm of zero? Here ε is set to 1.

The quantized 3D joint RGB histograms of the object, background and the computed positive log-likelihood ratio are shown in figure 3 In this figure, each square represents a non-empty seed in quantized 3D joint RGB histogram, and its area is proportional to the value of the corresponding seed in the histogram.

2.2. Detection and Localization

The positive log-likelihood ratio provided in previous step is used for object detection, and the mean-shift on the confident map is used for object localization. Steps are described in the following subsections.

2.2.1. Detection

The positive log-likelihood ratio L_s is used as a mapping to provide a confident map, $M(x_i, y_i)$ from the object region, $I(x_i, y_i, c_j)$, i.e.

$$L_s: I(x_i, y_i, c_j) \longmapsto M(x_i, y_i) \quad (2)$$

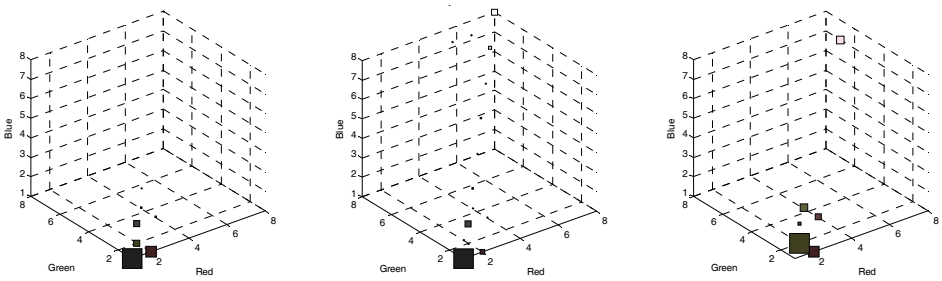
where $[x_i, y_i]$ is the pixel location in the image coordinate. Index i ranges from 1 to N and N is the total number of pixels in the object region. c_j Is the color channel of image. Index j ranges from 1 to 3 that stand for R-G-B color channels. The mapping result is a confident map image, and its pixels with higher values are more likely to belong to the object. The procedure is shown in figure 4 for the first frame of Trellis sequence.

2.2.2. Localization

Object localization for the next frame starts at the centric of the confident map of the object in the current frame. Since object movement is not ballistic, the mean-shift object localization on the confident map of the object provides satisfactory performance. The displacement of the object is given by the shift in the centric of the pixel values in the confident map. In each iteration, the center of the object rectangle is shifted to the centric of the current confident map of the object computed at the same iteration. The object rectangle is iteratively shifted until the object is placed inside the rectangle completely (mean-shift convergence). Mean-shift object localization on the confident map for the second frame is shown in figure 5 At each iteration, the center of the object rectangle is relocated using Equations (3) and (4),

$$x_{new} = \frac{\sum_{i=1}^{N}(M_i \times x_i)}{\sum_{i=1}^{N} M_i} \quad (3)$$

$$y_{new} = \frac{\sum_{i=1}^{N}(M_i \times y_i)}{\sum_{i=1}^{N} M_i} \quad (4)$$



(a) (b) (c)

Figure 3. Quantized 3D joint RGB histograms of (a) the object, (b) the background and (c) the computed positive log-likelihood ratio

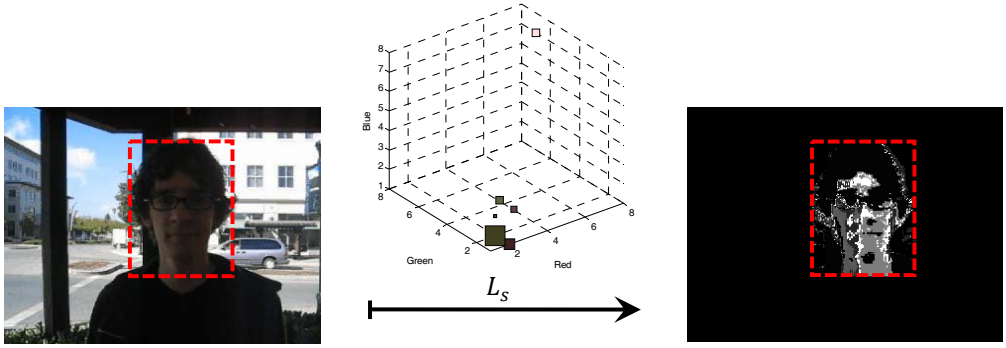


Figure 4. Generating the confident map of the object

where x_i and y_i show the location of pixels in the frame coordinate. M_i Is the confidence value, x_{new} and y_{new} show the relocated center of the object rectangle in each iteration and N is the total number of pixels in the inner rectangle. In the presented work, the maximum number of mean-shift iterations is limited to 3 and the centric movement less than 2 pixels are considered as a complete convergence.

2.3. Model Update

To account for appearance variation of the object, it is inevitable to adapt the object model to the recent observations. Here, once the object location at the current frame is provided by the mean-shift, the positive log-likelihood ratio, L_p^t, is computed and used to update the previous object model, L_p^{t-1}, by (5),

$$L_p^{t+1} \leftarrow (1-\gamma) \times L_p^{t-1} + \gamma \times L_p^t \quad (5)$$

where $t+1$, t and $t-1$ are indexes for the next, current and previous frames respectively. p Indicates the randomly α percent selection of the positive log-likelihood ratio seedss. Here α is set to 5%. γ Is a forgetting factor to moderate the balance between the old and new observations? Although the forgetting factor depends on the amount of changes in object appearance, here we set it to 0.1 for all video sequences. L_p^{t+1} Is the updated object model which will be used for the object detection in the next frame.

Figure 6 shows the effectiveness of the proposed model update procedure for the trellis sequence. Columns with left to right flow, show frames number 90, 250 and 355 in order. First and second rows show the object and its model respectively with no updating on the object model. The remaining rows show the object and corresponding model when the incremental learning is applied. Confident maps are shown in the upper left corner of each frame. As it is shown in the first raw when there is no update in the object model, the provided confident map degrades gradually; however, the proposed method incrementally learns the new colors of the object and provides a more precise confident map of the object.

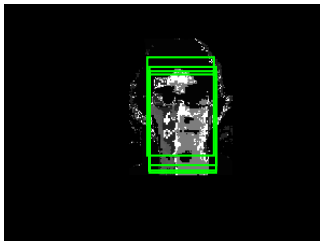


Figure 5. Mean-shift localization on the confident map

3. Experimental Results

In order to evaluate the performance of the proposed tracking algorithm, experiments were carried out using a core 2 Duo 2 GHz processor with 1 GB RAM under MATLAB R2011a on the Windows XP platform. The tracker has been tested on a variety of challenging sequences, and the most representative sequences are reported in this paper.

3.1. Data Sets

We have used the sequences which are publicly available. The sequences are Basketball, Crossing, David outdoor, Girl, Face occlusion, Tiger and Trellis available at Tracker Benchmark v1.0[3] [23, 24], Sunshade and Torus available at VOT2013[4] [25], Caviar1 and Caviar2 from Caviar dataset[5]

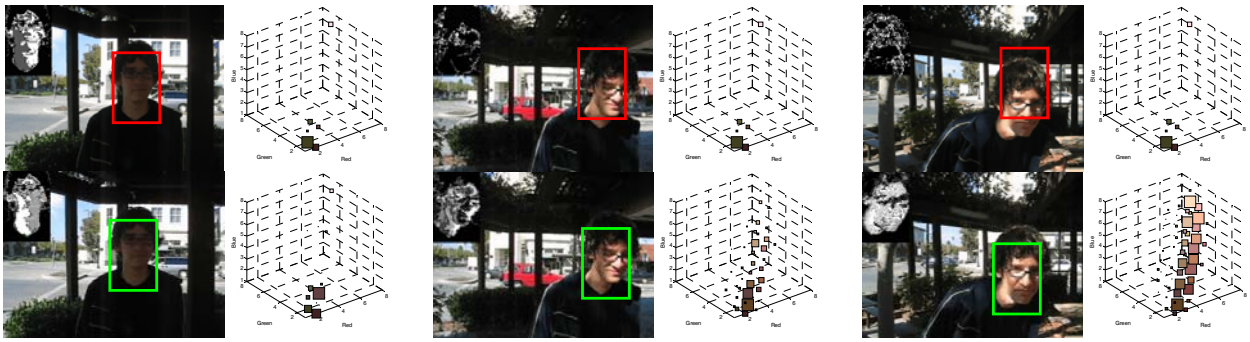


Figure 6. Effect of the model update procedure on the object model and corresponding confident map

and Board from Prost Dataset[6]. Seven of the sequences (Basketball, Board, David, Girl, Face, Sunshade, Trellis) were taken by non-stationary cameras and five of them (Caviar1, Caviar2, Crossing, Tiger and Torus) were taken by a stationary camera. The information and challenging factors for each sequence are described in table 1.

3.2. Evaluated Methods and Speed Comparison

The proposed method (Incremental Discriminative Color Tracking, called IDCT) is evaluated against five methods which their codes are publicly available, Mean Shift tracker (MS) [5], mean-shift tracker with Corrected Background Weighted Histogram (CBWH)[7] [6], Variance Ratio tracker (VR)[8] [10], Fragment based Tracker (Frog)[9] [7] and Locality Sensitive Histogram Tracker (LSHT)[10] [9]. We compared the proposed method with the most relevant visual object tracking methods that operate directly on histogram or color images. MS uses a color histogram regularized by a kernel to describe the appearance of an object. CBWH uses a color histogram with new weights to pixels in the target candidate region to reduce background's interference in object localization. VR uses 49 candidates of 1D line color features in RGB color space as a feature pool and chooses the best N features at each frame. Frog uses multiple local histograms for a fragment-based object representation, and a vote map is used to combine the votes from all the regions in the target frame. LSHT works based on a locality sensitive histogram that was computed at each pixel location along with a floating-point value corresponds with each bin to save the occurrence of an intensity value.

Object model is one of the essential components for visual tracking. Generally, tracking algorithms use generative or discriminative methods to model the object appearance. Generative methods build a reference model of an object and use this model to search for the object in the next frames. On the other hand, discriminative methods distinguish an object from surrounding background in every frame without using a definitive model of the object. As explained, each of the compared method is generative (Ms, CBWH, Frog, and LSHT) or discriminative (VR). In this paper, a combination of generative and discriminative methods is used to model the object appearance. It is used to provide a right balance between being adaptive and ability to re-track the object after losing the object due to occlusion. The quantized 3D joint RGB histograms of the region within the inner rectangle and the region between the inner and outer rectangles are used as a discriminative component. Generative component is composed of the positive part of the log-likelihood ratio of the computed 3D joint RGB histograms along with an incremental color learning scheme. In every frame, after localization, discriminative component differentiates the object from its surrounding background and adds the detected object colors to the generative model. The generative model is evolved during tracking and is used to detect the object in the next frame.

The parameters of the proposed algorithm are set as follows: in the object detection and localization parts, we use 8 bins for histogram quantization and the maximum number of mean-shift iterations is limited to 3 as empirical values. In the model update part, α, which is the percentage of randomly selected non-empty seeds is set to 5% and the forgetting factor, γ, is set to 0.1. For all other algorithms, we use the parameters provided by the corresponding authors and run them with the adjusted parameters. When it was possible, we run evaluated algorithms in different modes and report them. For example, MS and CBWH are run with 8 and 16 bins, and LSHT is run in intensity and illumination modes. Parameters of the algorithms were fixed for all the experiments. All trackers were run on a single scale. All algorithms assume the knowledge of the object position only in the first frame. The object position is estimated in the next frames using the corresponding algorithm. The average speeds of the evaluated algorithms are computed for Trellis (as a typical video with object size of 101×68 pixels) and together with the detailed information of each algorithm are reported in table 2.

Our tracker (IDCT) works at 17.2 frames per second and runs faster than the other algorithms except Variance Ratio (VR) method. It should be noted VR method is implemented in C++, which is intrinsically more efficient than MATLAB.

3.3. Results Comparison

Tracking algorithm evaluation is itself a challenging task. Therefore, in this section extensive quantitative and qualitative evaluations with different criterions are performed to compare the proposed method against other evaluated methods.

3.3.1. Quantitative Evaluation

For evaluating the performance of trackers, three criterions are used. The average center location error, the average overlap rate and the success rate (tracking percentage) which

Table 1. The detailed information of each sequence

Sequence name	No. of frames	Resolution	Challenge			
			Background clutter	Rotation and deformation	Illumination variation	Occlusion
Basketball	725	576×432	✓	✓	✓	✓
Board	698	640×480		✓		
Caviar1	382	384×288				✓
Caviar2	500	385×289	✓			✓
Crossing	120	360×240	✓			
David outdoor	252	640×480	✓	✓		✓
Face occlusion	892	352×288				✓
Girl	500	128×96		✓		✓
Sunshade	172	352×288	✓		✓	
Tiger	354	641×481		✓	✓	✓
Torus	264	320×240	✓	✓		
Trellis	569	320×240	✓	✓	✓	

Table 2. A comparison of tracking speed (including image I/O time) and specification of each algorithm. The best result is shown by the underline. The best three results are printed in bold faced letters

Algorithm	Feature description	Approach	Code style	Speed (fps)
MS 8 bin	Color histogram	Generative	MATLAB code	8.6
MS 16 bin				8.3
CBWH 8 bin	Color histogram	Generative	MATLAB code	8.9
CBWH 16 bin				8.6
VR	Line color feature	Discriminative	C++ code with OpenCV	**<u>17.4</u>**
Frog	Local gray histogram	Generative	C++ code with OpenCV	1.7
LSHT intensity	Local gray histogram	Generative	C++ code with OpenCV	**12.5**
LSHT illumination				11.1
IDCT (proposed)	Color histogram	Hybrid	MATLAB code	**17.2**

are common criterions. The average center location error (CLE) [21] is defined by,

$$\mathrm{CLE} = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\left(X_i - X_i^g\right)^2 + \left(Y_i - Y_i^g\right)^2} \tag{6}$$

where $[X_i, Y_i]$ shows the center location of the object determined by an algorithm. The center of the object is defined by the central point of the object window. $[X_i^g, Y_i^g]$ Denotes the center of the ground truth rectangle. i Ranges from 1 to n and n is the total number of frames. The total average center location error for all sequences is defined by,

$$\text{Total average} = \frac{\sum_{k=1}^{m}\text{error(k)} \times \text{number of frames (k)}}{\sum_{k=1}^{m}\text{number of frames (k)}} \tag{7}$$

where k ranging from 1 to m denotes each of the sequences. m Is the total number of sequences, which are 12 The average overlap rate (score) [26, 27] is defined by,

$$\text{Overlap rate} = \frac{\text{area}(G \cap T)}{\text{area}(G \cup T)} \tag{8}$$

where $G \cap T$ denotes the intersection of the tracked and ground truth rectangles and $G \cup T$ is their union. This criterion takes into account the different size of the objects in different scenarios. The third criterion which is success rate (tracking percentage) [26, 27] is extracted by exploiting the average overlap rate criterion. The range of the overlap rate is from 0 to 1. To be considered a success, the overlap ratio must exceed 0.5. The percentage of correctly tracked frames is the success rate. Therefore, the success rate is defined by,

$$\text{success rate} = \frac{\sum_{i=1}^{n}\text{success}}{n} \tag{9}$$

where i ranges from 1 to n and n is the total number of frames. The success is defined by,

$$\text{success} = \begin{cases} 1 & \text{if overlap rate} > 0.5 \\ 0 & \text{if overlap rate} \leq 0.5 \end{cases} \tag{10}$$

Table 3 demonstrates the tracking results in terms of average center location errors (AC), average overlap rate (AO), and success score (SR). It can be seen that the proposed method provides superior results in most individual scenarios and the best total result in comparison with evaluated methods. The details of center location errors and overlap rates for the sequences are shown in figure 7 and 8 respectively.

3.3.2. Qualitative Evaluation

In this section, the qualitative comparison on video sequences is presented. Figure 9 shows sample screenshots of the tracking results for the video sequences where each row corresponds to one video sequence.

In the Basketball sequence, our method and LSHT-illumination track the object successfully. The object trajectories provided by MS 8 and 16, CBWH 8 and 16 and LSHT-intensity methods were hijacked by the other cluttered basketball players (e.g. frames #600 and #700). The Frog and Variance Ratio methods lost the object. For the Board sequence, only the Variance Ratio method loses the object in some middle frames (e.g. frame #575) and the other methods perform well.

Table 3. The average center location errors (AC), average overlap rate (AO), and success score (SR) of the evaluated methods on the sequences. The best result is shown by the underline. The best three results are printed in bold-faced letters. The entry '-' for Frog indicates that the value is not available as the algorithm loses track of the object and provides error

Seq. \ Algorithm		Basketball	Board	Caviar1	Caviar2	Crossing	David	Face	Girl	Sunshade	Tiger	Torus	Trellis	Total
MS 8	AC	57.0	34.7	12.7	11.6	14.0	68.6	17.1	18.5	72.6	**25.4**	8.2	54.1	32.1
	AO	0.57	0.65	0.55	**0.56**	0.48	0.40	0.73	0.43	0.27	**0.52**	0.66	0.28	0.54
	SR	0.71	0.84	0.59	**0.62**	0.37	0.47	**0.92**	0.48	0.31	**0.55**	0.81	0.23	0.64
MS 16	AC	58.4	30.5	13.5	13.8	12.6	55.6	18.4	19.0	71.2	26.0	36.2	38.3	31.3
	AO	0.55	0.68	0.53	0.53	0.51	0.45	0.73	0.43	0.31	**0.51**	0.47	0.33	0.54
	SR	0.69	0.88	0.58	0.60	0.48	0.56	**0.99**	**0.54**	0.39	**0.54**	0.54	0.30	0.66
CBWH 8	AC	18.3	**21.0**	16.2	6.7	6.7	**25.2**	23.9	26.9	**9.8**	27.3	**4.2**	23.6	**19.4**
	AO	**0.74**	**0.77**	0.55	0.54	**0.66**	**0.61**	0.69	0.34	**0.60**	0.49	**0.78**	0.46	**0.61**
	SR	**0.87**	**0.92**	0.55	0.50	0.83	**0.97**	0.90	0.41	**0.85**	0.51	**0.95**	0.51	**0.73**
CBWH 16	AC	**18.1**	**19.8**	14.7	6.8	**5.8**	26.6	17.2	27.6	**9.4**	**25.8**	**3.4**	**18.5**	**17.4**
	AO	**0.74**	**0.79**	0.57	**0.58**	**0.67**	**0.59**	0.75	0.35	**0.61**	**0.51**	**0.81**	**0.51**	**0.64**
	SR	0.86	**0.91**	0.56	**0.64**	**0.85**	0.95	**0.99**	0.46	**0.93**	0.53	**0.95**	**0.57**	**0.77**
VR	AC	116.1	74.3	51.9	**4.9**	**6.5**	81.8	59.7	62.4	13.9	27.2	7.6	60.0	57.5
	AO	0.51	0.57	0.25	0.53	**0.66**	0.45	0.43	0.09	0.48	0.47	0.70	0.29	0.43
	SR	0.65	0.77	0.29	0.50	**0.89**	0.63	0.53	0.12	**0.40**	0.51	**0.82**	0.37	0.52
Frog	AC	-	66.7	**5.5**	5.9	47.6	-	**10.5**	**6.6**	29.5	105.3	25.9	71.6	35.9
	AO	-	0.49	**0.67**	**0.61**	0.34	-	**0.82**	**0.63**	0.34	0.12	0.49	0.27	0.53
	SR	-	0.48	**0.93**	**0.65**	0.41	-	**1.00**	**0.75**	0.35	0.10	0.50	0.35	0.62
LSHT int.	AC	18.5	23.4	7.2	**5.1**	27.5	101.2	**11.8**	**11.4**	81.8	109.4	**3.8**	**19.9**	26.7
	AO	**0.66**	**0.76**	**0.69**	**0.58**	0.39	0.31	**0.81**	**0.55**	0.14	0.13	**0.79**	**0.55**	0.60
	SR	0.86	0.73	**0.95**	0.61	0.51	0.35	**1.00**	**0.60**	0.12	0.15	**0.94**	**0.73**	0.71
LSHT illus.	AC	**9.0**	**19.6**	**3.3**	7.5	28.9	**5.6**	**12.6**	36.1	61.5	70.9	37.0	50.2	24.6
	AO	**0.66**	**0.79**	**0.71**	**0.58**	0.38	**0.73**	**0.80**	0.17	0.26	0.19	0.48	0.40	0.57
	SR	**0.94**	0.89	**0.98**	0.61	0.44	**0.98**	**1.00**	0.17	0.23	0.13	0.51	**0.59**	0.70
IDCT	AC	**5.4**	22.8	**7.1**	**4.8**	**5.1**	**23.1**	20.4	**12.8**	**12.5**	**19.7**	**4.2**	**15.2**	**13.8**
	AO	**0.82**	**0.76**	**0.67**	0.55	**0.60**	**0.61**	0.72	**0.49**	**0.56**	**0.58**	**0.78**	**0.56**	**0.66**
	SR	**0.99**	**0.96**	**0.98**	0.55	**0.84**	**0.96**	**1.00**	0.45	**0.85**	**0.63**	**0.94**	**0.73**	**0.83**

For the Board sequence, only the Variance Ratio method loses the object in some middle frames (e.g. frame #575) and the other methods perform well. For the Caviar1 sequence, all methods except the Variance Ratio have a good performance and track the object reasonably. In the Caviar2 sequence, MS 8 and 16 and CBWH 16 lose the object when another person gets into the object background region as shown in the frame #470, while other methods have a better tracking result. For the Crossing sequence, when a car passes in front of the object, the Frog and LSHT-intensity and illumination methods lose the object. Almost the other methods perform well. In the David outdoor sequence our method and CBWH 8 and 16 have a good performance. MS 8 and 16 lose the object from frame #100 to #180. Variance Ratio, Frog and LSHT-intensity drifts when the target object is occluded by a tree (it can be seen in frames #125 and #230). In the Face occlusion sequence, all methods except the Variance Ratio have a reasonable result and track the object to the end of the sequence. For the Girl sequence, since our object model evolves during tracking, our method successfully tracked the object. The LSHT-intensity and Frog methods have a good performance too, while the other methods lose the object (as shown in frames #100 and #425). In the Sunshade sequence MS 8 and 16 and LSHT-illumination and intensity did not perform well due to illumination variation. However, almost the other methods have reasonable results. Although the LSHT-illumination method is for the environments with illumination variation, it fails in some frames as shown in frames #70 and #140. For the Tiger sequence which has occlusion, LSHT-illumination and intensity and Frog did not have a good performance while other methods tracked the object reasonably. In the Torus sequence, MS 16, Frog and LSHT-illumination drift to background clutter. Almost other methods track the object to the end of the sequence. In this sequence, there is no illumination variation, so the LSHT-illumination method is not supposed to have a good performance. For the Trellis sequence the object appearance changes drastically due to the changes in the illumination, only the proposed method achieves good performance and is able to adapt to the illumination variation well. After our method, CBWH 16 performed better than the other methods. It is shown that in frames #420 and #490, the LSHT-illumination failed to track the object, although this type of LSHT is for the sequences which have illumination variation.

3.4. Scale Handling

The aspect ratio of an object change over time. The proposed method allows tracking of objects with changing aspect ratio and scale. For an object's bounding box, the aspect ratio denotes the ratio of longer side to shorter side of the rectangle. Here, the scale change and aspect ratio change of the object is addressed by the resulting confident map of the object. The confident map within the object window at first frame is used as a reference model to compute a map ratio. The map ratio is defined as an average of the confident map pixel values within the tracked rectangle,

$$\text{Map Ratio} = \frac{\sum_{i=1}^{n} \text{confident map(i)}}{n} \quad (11)$$

where i ranges from 1 to n and n is the total number of pixels in the tracked rectangle. To compensate all the possible changes in the aspect ratio of the object, five candidate scales are considered: $(w - \Delta w, h - 2 \times \Delta h)$, $(w - 2 \times \Delta w, h - \Delta h)$, (w, h), $(w + \Delta w, h + 2 \times \Delta h)$ and $(w + 2 \times \Delta w, h + \Delta h)$ where w and h indicate the width and height of the object window

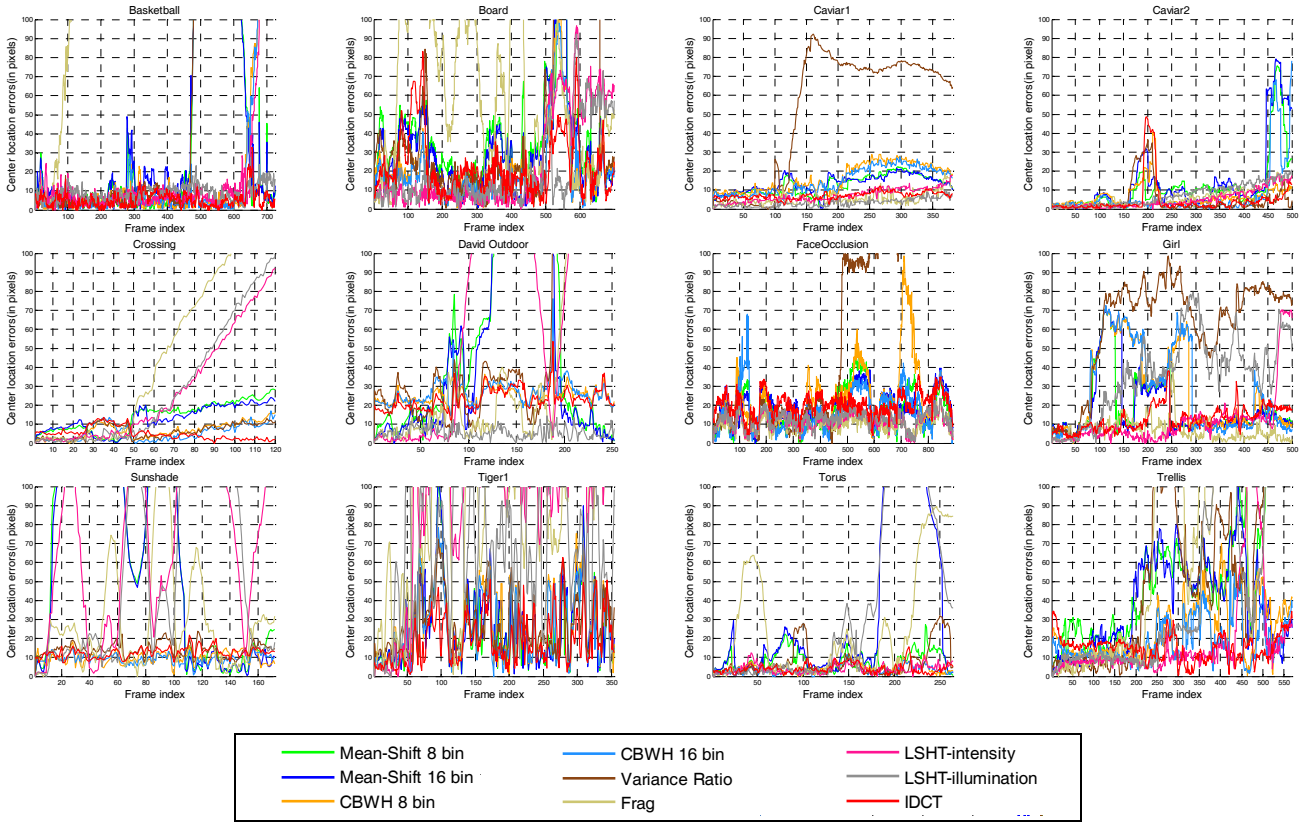


Figure 7. The center location errors of the sequences

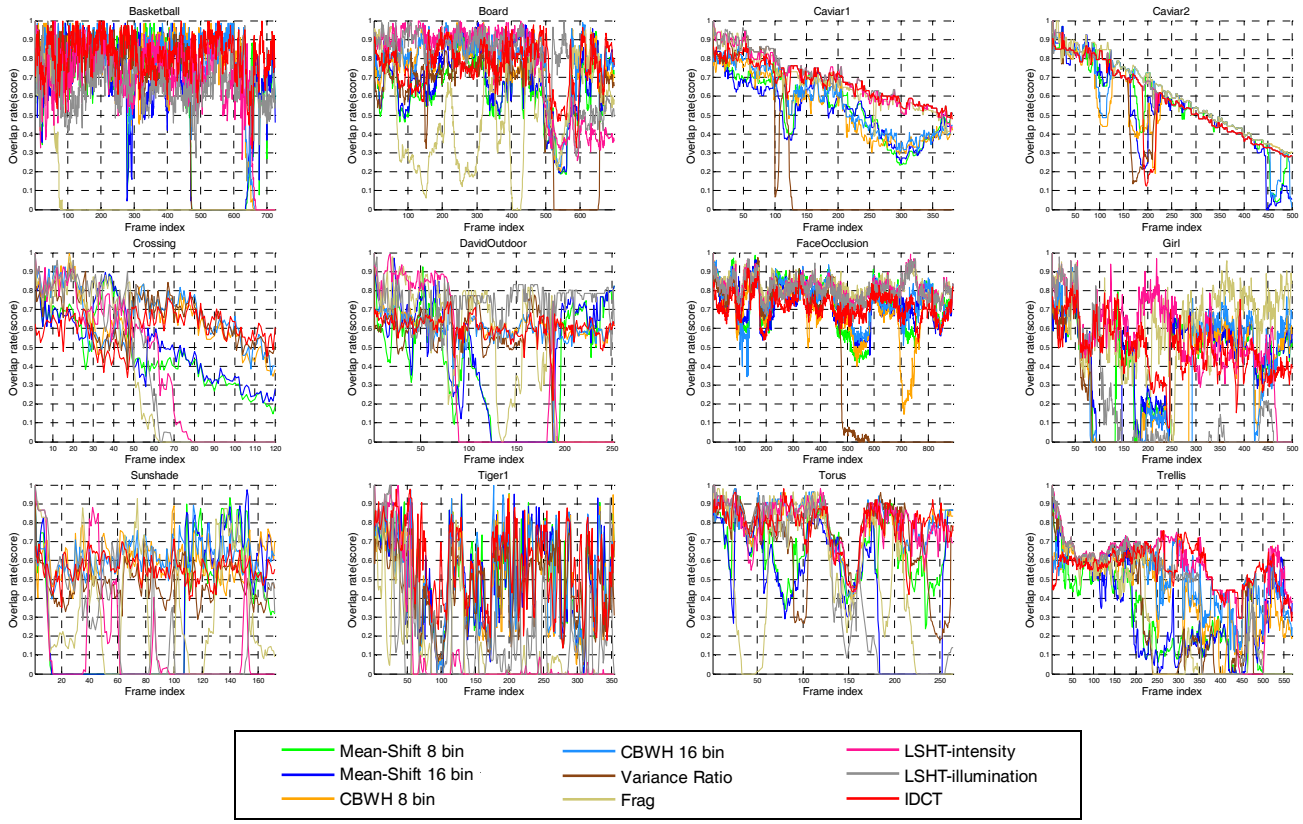


Figure 8. The overlap rate of the sequences

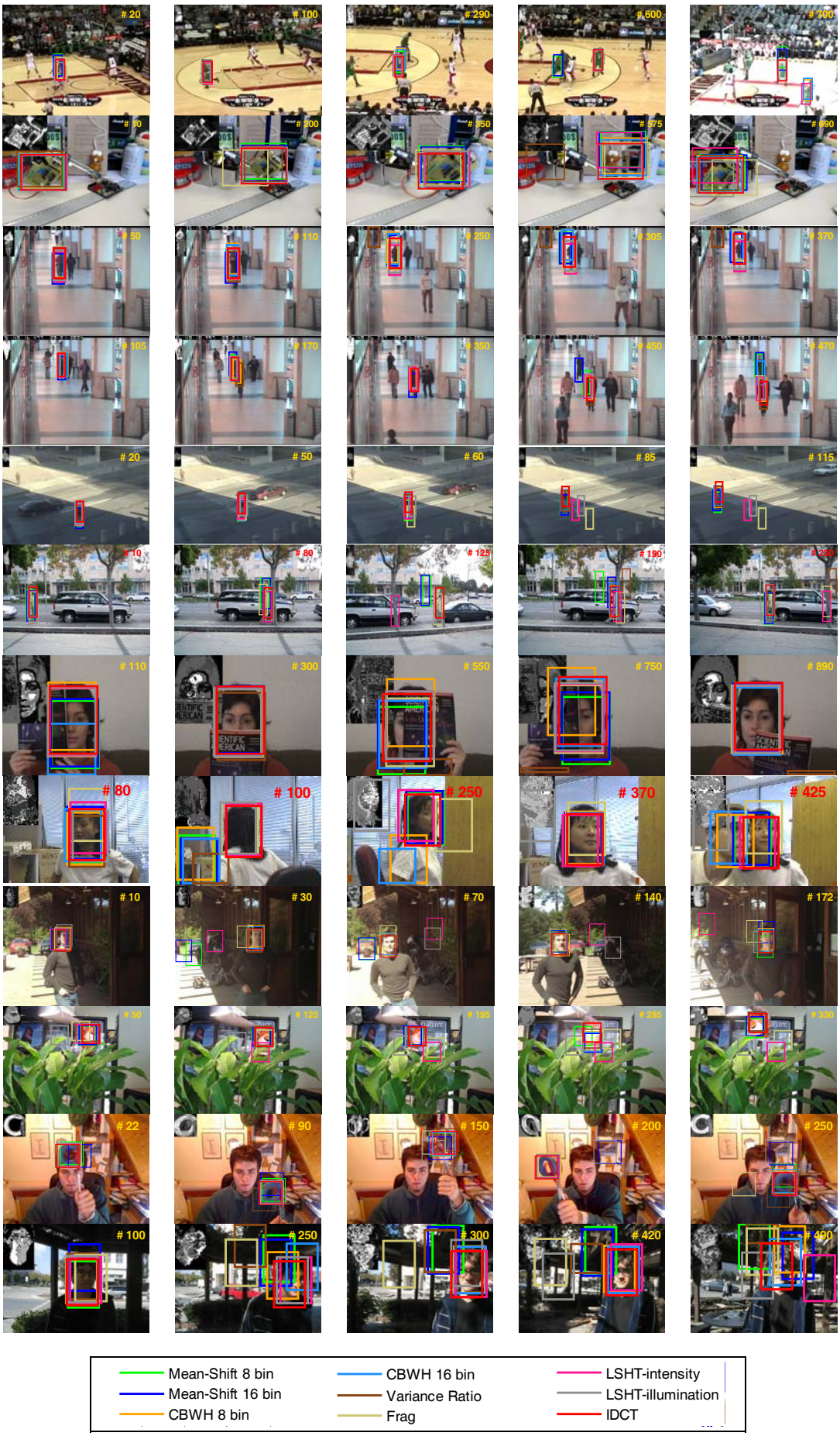


Figure 9. Screenshots of tracking results presented for sequences as listed in Table 1. The object confident map provided by the proposed method (IDCT) is shown at the top-left corner of the frames

in the previous frame respectively, $\Delta h = 1.25\%\ h$ and $\Delta w = 1.25\%\ w$. The five candidate scales are shown in figure 10. It covers the possible changes in width and height of the object's rectangle. For the next incoming frame after object localization, the map ratio for five candidate scales is computed. If the computed map ratio does not experience much change, the previous scale will be applied for the next frame. If the computed map ratio is reduced, it is assumed the object size is reduced, therefore, rectangle of the object must become smaller and the choices for object scale will be limited to red rectangles and the one which best fits is selected as an object new scale. A similar strategy is used when the map ratio is increased where the choice will be limited to blue rectangles. The details of the process of selecting the appropriate scale for the incoming frames are shown in figure 11 In the depicted algorithm in figure 11, $\text{Map Ratio}(w, h)$ is computed using (11). w And h are denote the current scale and ratio of the object. w_0 And h_0 are the width and height of the object in the first frame. w_{new} And h_{new} are the new scales of the object that will be used to determine the rectangle of the object for the next frame. The fault is computed by,

$$\text{fault}(w_{cand}, h_{cand}) = \text{abs}(\text{Map Ratio}(w_{cand}, h_{cand}) - \text{Map Ratio}(w_0, h_0)) \quad (12)$$

where w_{cand} and h_{cand} are the candidate scale. Minimum fault shows the scale that better fits the object. The candidate scale with minimum fault is used to determine the new height and width of the object.

Figure 12 shows an example of applying the proposed method to an object in PETS 2000[11] with changing scale and ratio. It presents a comparison between computed width and height using proposed method against ground truth. The result shows the proposed method closely follows the actual changes in width and height of the object's bounding box. Screenshots of the result is shown in figure 13 Proposed method is shown by solid green rectangle. Confident map obtained by the proposed method is shown in the upper left corner of the images.

3.5. Multiple-Object Tracking

With some minor modifications, proposed tracker successfully applied to track multiple objects with different appearances. The test sequence is 50 frames of Subway sequence [23] where the objects are two walking passengers. The result is presented in figure 14. The first row of figure 14 shows the result provided by the proposed method. Second row shows the confident map corresponding to the selected frame. Third and Forth rows show the models of red and white passengers respectively.

3.6. Failure Case Analysis

Figure 15 shows a failure case on Stone sequence [23]. It can be seen in frame #140 the tracker drifts gradually to the neighborhood background. The reason is the color of the target stone is similar to the neighborhood stones. Therefore the proposed method may fail when the neighborhood area near the target contains a region larger than the object with the similar color to the object.

4. Concluding Remarks and Future Work

In this paper, we proposed a fast effective object tracking algorithm with incremental discriminative object color modeling. The algorithm efficiently addresses tracking of an object which undergoes variation in illumination and color. We propose two ways for the future work. First, the motion and object model we used here are fairly simple, in future, prediction-based methods such as a Kaman filter [28] can be used. Second, inherent in the use of histograms is the complete lack of spatial information which is undesirable and cause the failure case demonstrated in this paper. Template patches can be incorporated in the object model to preserve spatial information which will be our future work.

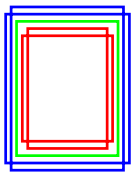


Figure 10. Candidate scales. The original scale is shown by green. The candidate scales with decrease in size are shown by red and the candidate scales with increase in size are shown by blue

Scale Handling Algorithm Outline

if $\text{Map Ratio}(w, h) < 0.9 \times \text{Map Ratio}(w_0, h_0)$

$(w_{new}, h_{new}) = \min(\text{fault}(w - 2 \times \Delta w, h - \Delta h), \text{fault}(w - \Delta w, h - 2 \times \Delta h))$

elseif $\text{Map Ratio}(w, h) > 1.1 \times \text{Map Ratio}(w_0, h_0)$

$(w_{new}, h_{new}) = \min(\text{fault}(w + 2 \times \Delta w, h + \Delta h), \text{fault}(w + \Delta w, h + 2 \times \Delta h))$

else

$(w_{new}, h_{new}) = (w, h)$

end

Figure 11. Scale handling algorithm

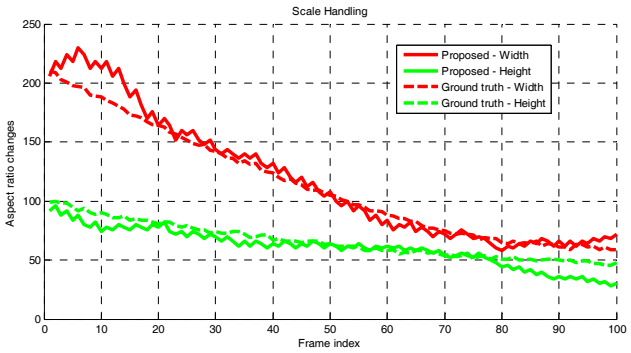


Figure 12. Aspect ratio handling's result. It shows the actual change in aspect ratio of object extracted by an expert (dashed) and aspect ratio computed using proposed approach

References

[1] A. Yilmaz, O. Javed, and M. Shah, "Object Tracking: A Survey," *ACM Computing Surveys*, vol. 38, no. 13, pp. 28-49, 2006.

Figure 13. Screenshots of scale handling results and corresponding confident map

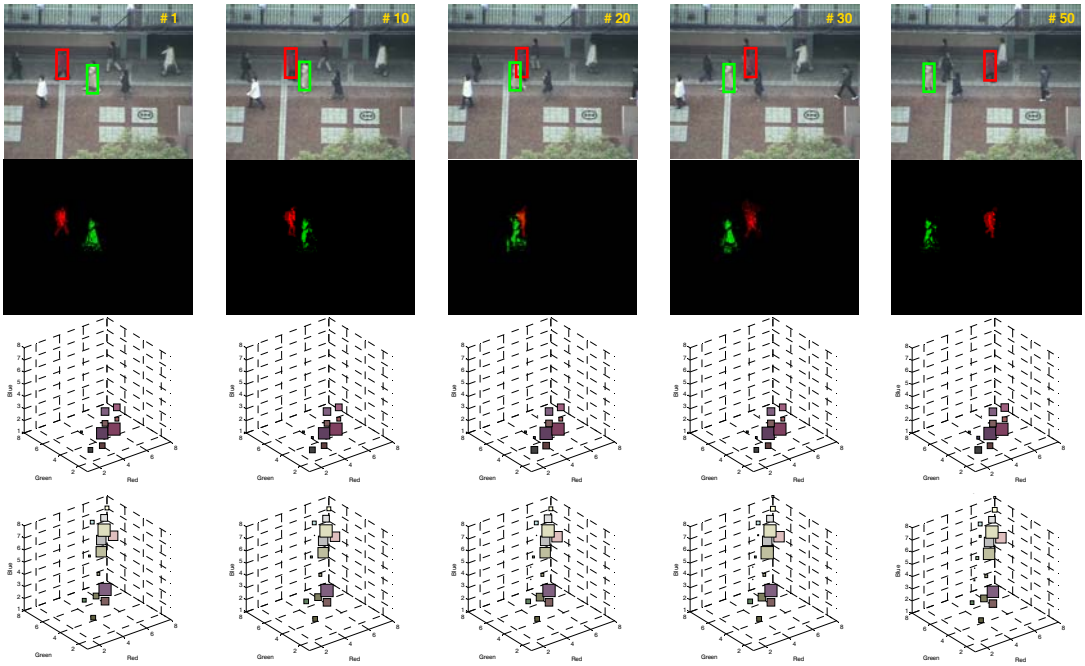


Figure 14. Screenshots, confident maps and object color models for multiple object tracking results

Figure 15. Screenshots for the failure case

[2] K. Cannons, *A Review of Visual Tracking*, Technical Report, York University, Toronto, Canada, 2008.

[3] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent Advances and Trends in Visual Tracking: A Review," *Neurocomputing Journal*, vol. 74, no. 18, pp. 3823-3831, 2011.

[4] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel, "A Survey of Appearance Models in Visual Object Tracking," *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 4, pp. 321-342, 2013.

[5] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-Based Object Tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564-577, 2003.

[6] J. Ning, L. Zhang, D. Zhang, and C. Wu, "Robust Mean-Shift Tracking with Corrected Background-Weighted Histogram," *IET Computer Vision Journal*, vol. 6, no. 1, pp. 62-69, 2012.

[7] A. Adam, E. Rivlin, and I. Shimshoni, "Robust Fragments-based Tracking using the Integral Histogram," *Proc. IEEE Intl Conf. on Computer Vision and Pattern Recognition*, pp. 798-805, 2006.

[8] S. S. Nejhum, J. Ho, and M. H. Yang, "Online Visual Tracking with Histograms and Articulating Blocks," *Journal of Computer Vision and Image Understanding*, vol. 114, no. 8, pp. 901-914, 2010.

[9] S. He, Q. Yang, R. W. Lau, J. Wang, and M. H. Yang, "Visual Tracking via Locality Sensitive Histograms," *Proc. IEEE Intl Conf. on Computer Vision and Pattern Recognition*, pp. 2427-2434, 2013.

[10] R. T. Collins, Y. Liu, and M. Leordeanu, "Online Selection of Discriminative Tracking Features," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631-1643, 2005.

[11] Y. Wei, J. Sun, X. Tang, and H. Y. Shum, "Interactive Offline Tracking for Color Objects," *Proc. IEEE Intl Conf.*

on Computer Vision, pp. 1-8, 2007.

[12] D. Wang, H. Lu, Z. Xiao, and Y. Chen, "Fast and Effective Color-based Object Tracking by Boosted Color Distribution," *IEEE Trans. on Pattern Analysis and Applications*, vol. 16, no. 4, pp. 647-661, 2013.

[13] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the Circulant Structure of Tracking-by-Detection with Kernels," *Proc. IEEE European Conf. on Computer Vision*, pp. 702-715, 2012.

[14] M. Danelljan, F. S. Khan, M. Felsberg, and J. V. D. Weijer, "Adaptive Color Attributes for Real-Time Visual Tracking," *Proc. IEEE Intl Conf. on Computer Vision and Pattern Recognition*, pp. 1090-1097, 2014.

[15] D. Chen, Z. Yuan, G. Hua, Y. Wu, and N. Zheng, "Description-Discrimination Collaborative Tracking," *Proc. IEEE European Conf. on Computer Vision*, pp. 345-360, 2014.

[16] S. Hong, T. You, S. Kwak, and B. Han, "Online Tracking by Learning Discriminative Saliency Map with Convolutional Neural Network," *arXiv preprint arXiv:1502.06796*, 2015.

[17] A. Asvadi, M. Karami, and Y. Baleghi, "Object Tracking using Adaptive Object Color Modeling," *Proc. IEEE Conf. on Information and Knowledge Technology*, pp. 848-852, 2012.

[18] S. Avidan, "Ensemble Tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 261-271, 2007.

[19] H. Grabner, C. Leistner, and H. Bischof, "Semi-Supervised On-Line Boosting for Robust Tracking," *Proc. IEEE European Conf. on Computer Vision*, pp. 234-247, 2008.

[20] D. A. Ross, J. Lim, R. S. Lin, and M. H. Yang, "Incremental Learning for Robust Visual Tracking," *International Journal of Computer Vision*, vol. 77, no. 3, pp. 125-141, 2008.

[21] B. Babenko, M. H. Yang, and S. Belongie, "Robust Object Tracking with Online Multiple Instance Learning," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619-1632, 2011.

[22] A. Asvadi, H. Mahdavinataj, M. Karami, and Y. Baleghi, "Incremental Discriminative Color Object Tracking," *Proc. IEEE Intl Symp. on Artificial Intelligence and Signal Processing*, pp. 71-81, 2013.

[23] Y. Wu, J. Lim, and M. H. Yang, "Online Object Tracking: A Benchmark," *Proc. IEEE Intl Conf. on Computer Vision and Pattern Recognition*, pp. 2411-2418, 2013.

[24] Y. Wu, J. Lim, and M. H. Yang, "Object Tracking Benchmark," *To Appear in IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[25] M. Kristan, and et. al, "The Visual Object Tracking vot 2013 Challenge Results," *Proc. IEEE Intl Workshop on Computer Vision*, pp. 98-111, 2013.

[26] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof, "Prost: Parallel Robust Online Simple Tracking," *Proc. IEEE Intl Conf. on Computer Vision and Pattern Recognition*, pp. 723-730, 2010.

[27] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (voc) Challenge," *Intl Journal of Computer Vision*, vol. 88, no. 2, pp. 303-338, 2010.

[28] H. Jahandide, K. Mohamedpour, and H. A. Moghaddam, "A Hybrid Motion and Appearance Prediction Model for Robust Visual Object Tracking," *Pattern Recognition Letters*, vol. 33, no. 16, pp. 2192-2197, 2012.

Alireza Asvadi received Bachelor degree in Electrical Engineering from University of Isfahan, Isfahan, Iran in 2009 and Master degree in Electrical Engineering from Babol (Noushirvani) University of Technology, Babol, Iran in 2012. His research interests include the field of Computer Vision, with an emphasis on object tracking.
E-mail: alireza.asvadi@gmail.com

Hami Mahdavinataj received the B.Sc. degree in Electrical Engineering in 2012 from Amirkabir University of Technology (Tehran Polytechnic), Iran. He received the M.Sc. degree in Electrical Engineering in 2015 from Babol Noshirvani University of Technology, Iran. His research interests include image processing and distributed video coding.
E-mail: mahdavinataj@stu.nit.ac.ir

Mohammad Reza Karami received the Bs.c in electronic engineering in 1992, Ms.c of signal processing in 1994, and PhD in 1998 in biomedical engineering from I.N.P.L d'Nancy of France. He is now the Associate professor with the Department of Electrical and Computer Engineering, Babol University of Technology. Since 1998 his research is in signal and speech processing. He publishes 80 articles in journals and conferences.
E-mail: mkarami@nit.ac.ir

Yasser Baleghi is an Assistant Professor of Electronic Engineering at Babol University of Technology. He holds a PhD degree in Electronic Engineering from Iran University of Science & Technology. His research interests are evolvable and adaptive hardware, image processing and fault tolerant system design.

E-mail: y.baleghi@nit.ac.ir

Paper Handling Data:

Submitted: 07.06.2014
Received in revised form: 04.04.2015
Accepted: 10.05.2015
Corresponding author: Dr. Yasser Baleghi,
Department of Electrical and Computer Engineering, Babol University of Technology, Babol, Iran.

[1] An early version of this work was presented in [22]
[2] http://www.a-asvadi.ir/idct/
[3] https://sites.google.com/site/trackerbenchmark/benchmarks/v10
[4] http://votchallenge.net/
[5] http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/
[6] http://gpu4vision.icg.tugraz.at/index.php?content=subsites/prost/prost.php
[7] http://www4.comp.polyu.edu.hk/~cslzhang/CBWH.htm
[8] http://vision.cse.psu.edu/data/vividEval/software.html
[9] http://www.cs.technion.ac.il/~amita/fragtrack/fragtrack.htm
[10] http://www.shengfenghe.com/visual-tracking-via-locality-sensitive-histograms.html
[11] http://www.cvg.rdg.ac.uk/slides/pets.html