Published in final edited form as:

Optik (Stuttg). 2015 October ; 126(19): 1830–1837. doi:10.1016/j.ijleo.2015.05.018.

Learning based particle filtering object tracking for visible-light systems

Wei Sun*

School of Aerospace Science and Technology, Xidian University, No. 2 Tabai Rd., Xi'an 710071, China

Abstract

We propose a novel object tracking framework based on online learning scheme that can work robustly in challenging scenarios. Firstly, a learning-based particle filter is proposed with color and edge-based features. We train a support vector machine (SVM) classifier with object and background information and map the outputs into probabilities, then the weight of particles in a particle filter can be calculated by the probabilistic outputs to estimate the state of the object. Secondly, the tracking loop starts with Lucas–Kanade (LK) affine template matching and follows by learning-based particle filter tracking. Lucas–Kanade method estimates errors and updates object template in the positive samples dataset, and learning-based particle filter tracker will start if the LK tracker loses the object. Finally, SVM classifier evaluates every tracked appearance to update the training set or restart the tracking loop if necessary. Experimental results show that our method is robust to challenging light, scale and pose changing, and test on eButton image sequence also achieves satisfactory tracking performance.

Keywords

Object tracking; Lucas–Kanade algorithm; Support-vector machine; Particle filtering; Online learning scheme; eButton

1. Introduction

Visible light systems are widely employed in vision-based applications because of the low cost. In general, visual object tracking is the process of location estimation for one or multiple objects in videos. It is required by many applications such as visual navigation, human-computer interfaces, video communication, compression and surveillance. Key requirements imposed on a robust tracker can be given as following: (1) being able to track arbitrary targets and tracking accurately through challenging conditions, (2) tracking in real-time and capturing of the object when it reappears in the camera's field of view. So, it is a great challenge to design robust features and tracking methods which can cope with

* Tel.: +86 13991805685. wsun@xidian.edu.cn.

Uncited references
[16,23–25].

variations in natural scenes, such as changes in illumination, shape, viewpoint or partial occlusion of the target.

In the past decades, various features like points, articulated models, contours, or optical flow [1,2] are used, and many tracking methods based on image patch or color histograms are proposed [1]. In [3], in which color and local binary pattern (LBP) histograms are used to build the objects model, these features result in a system less sensitive to illumination changes and partial occlusion. But these generative model trackers often fail in cluttered background. Recently, particle filter [4] has been successfully applied to solve non-linear and non-Gaussian tracking problems, and some novel particle filter have been proposed in [7,20]. Besides these achievements, it is still a challenge on how to build an adaptive model of the target's appearance which can be generalized to possible future appearances. Several research groups have carried out investigation on this open question and some methods have been proposed. Han et al. [6] introduced a mean-shift based sequential kernel density approximation technique to update a target appearance model online. However, the addition and deletion of a Gaussian component highly depend on the pre-defined threshold values.

So, for handling appearance changes and coping with background clutter [15], a learning scheme with a classifier that represents the decision boundary between the object and its background may be the best choice. Avidan et al. [5] presented an algorithm to adapt the constituent parts and combined an ensemble of classifiers itself to new appearances. Lei et al. [7] tried to adapt an off-line learning ensemble classifier of a particular tracked object for the changing appearance. Despite their success, these methods are limited by the fact that they cannot accommodate very large changes in appearance.

In recent years, many discriminative trackers [9–12] that are capable to track, learn and detect have been proposed. Grabner et al. [8] demonstrated a semi-supervised on-line learning scheme to tackle the problem of uncertain class assignments of training examples. Babenko et al. [13] suggested a Multiple Instance Learning (MIL) approach for object detection. However, if the tracked location is not precise, the appearance model ends up getting updated with a sub-optimal positive example. Over time this can degrade the model, and can cause drift. Kalal et al. [15] formulated tracking as an online learning problem. Combining with a prior classifier, this method takes all incoming samples as unlabeled and uses them to update the tracker. But it throws away global features of the object. It also suffers from drift and fails if the object leaves the scene for longer than expected.

To deal with the problems discussed above, we present an approach in which template tracking and learning-based particle filtering detection are independent processes that exchange information by learning mechanisms. Firstly, we build the joint color-texture histogram to represent an object reference model, and propose a learning-based particle filter, using the probabilistic output of the SVM classifier to approximate the weight of the particles. Secondly, the classifier of initial target appearance model is quickly learned from the first frame and it will be used to detect the probability distribution of the object by learning-based particle filter in the following frames. Thirdly, to update the object model online and overcome the position drifting problem, we combine template matching and

learning-based particle filtering tracking techniques, they work together and supplement each other, and the switching scheme will be discussed in detail in the following sections.

The rest of the paper is organized as follows. Section 2 gives an overview of particle filter based visual tracking system. Section 3 introduces our learning-based object model and describes how it is adapted. Section 4 proposes learning-based tracking framework. Section 5 presents a number of comparative experiments and analysis, which is followed concluding remarks and future work in Section 6.

2. The scheme of classic particle filtering algorithm

Over past few years, condensation algorithm, also known as particle filter, have proved to be a powerful tool for image tracking [3–5]. In [4], an important advantage of particle filtering frameworks has been proposed and the strength of these methods lies in their simplicity, flexibility, and systematic treatment of nonlinearity and non-Gaussianity.

The main idea of the particle filter algorithm is that post probability distribution of the tracked object over state $X = (x, y, w, h)^T$ can be approximated by a set of weighted particles

$S_t = \{s_t^j\}$, $j \in \{1 \cdots J\}$, where (x_t^j) specifies the center of the tracked object's position in the image, and (w, h) are the dimensions of the target rectangle. Each particle $s_t^j = (x_t^j, \pi_t^j)$ consists of its state vector x_t^j and an importance weight π_t^j . The set of particles is updated from one frame to the next by the following recursive procedure:

Firstly, a new sample set S_t is drawn with replacement from the previous set S_{t-1} , where a sample s_{t-1}^i from the old set is chosen with probability proportional to its weight π_{t-1}^i .

Secondly, for each sample x_t^j , a new state is determined by updating the importance weights π_t^j with the likelihood of the observation

$$\pi_t^j = p(Z_t | X_t = x_t^j, Z_0, Z_1 \cdots Z_{t-1}) \quad (1)$$

where the likelihood depends on all frames $Z_0, Z_1 \cdots Z_{t-1}$, because the observation model is adapted over time. The sample set is propagated through a dynamic model given by

$$s_t = A \cdot s_{t-1} + w_{t-1} \quad (2)$$

where A defines the deterministic component of the model and w_{t-1} is a multivariate Gaussian variable. In our application, we use a first order model for A describing a region moving with constant velocities v_x, v_y .

The state of a target is estimated by the weighted average over states of the particles,

$$\left(\bar{x}, \bar{y}, \bar{w}, \bar{h}\right)^T = \sum_{j=1}^J \pi_t^j \cdot \left(x_t^j, y_t^j, w_t^j, h_t^j\right)^T \quad (3)$$

So, we need to compute the likelihood $p(Z_t|X_t=x_t^j, Z_0, Z_1 \cdots Z_{t-1})$ to get the weight of particles. Using color features, the target is tracked by comparing its histogram with the histograms of the sample set using Bhattacharyya distance, for discrete densities of color histograms $p = \{p^{(u)}\} \ u = 1 \dots m$ and $q = \{q^{(u)}\} \ u = 1 \dots m$, the Bhattacharyya distance is defined as

$$\rho[p, q] = \sum_{u=1}^m \sqrt{p^{(u)} q^{(u)}} \quad (4)$$

The larger ρ is, the more similar the distributions are. According to [2], particle filters may perform poorly when the posterior is multimodal as the result of ambiguities or multiple targets. To keep up with the changes in real-world scenarios, it is best not to rely on a fixed target model, but to adapt the model over time. In this way, spatial target properties can be updated.

3. Learning-based particle filter for object tracking

3.1. Analysis of target appearance model

As discussed above, illumination conditions, visual angle, as well as the camera parameters can influence the tracking result of particle filter. We do some statistical research on the TLD dataset of 10 video sequences [15], among these videos, the 'David' video has been used in several recent published papers, since it could present challenging lighting, scale and pose changes. Fig. 1 shows snapshots from 'David' and Fig. 2 lists RGB and LBP histograms of object. Color histogram is calculated in RGB space for each color channel (R, G and B) [4]. LBP is a texture operator with a low computational complexity that describes an object's local structure [6,7]. The object in this sequence is manually annotated with bounding boxes. In Fig. 2(a) and (c), color and texture histograms of the object in the bounding boxes are various over time. In Fig. 2(b) and (d), Bhattacharyya distance between the initial object histogram and other image frames decreases sharply.

Hence, we get the following conclusions: to cope with appearance variations of the target object during tracking, the model of the object should be adaptive over time. However, perfect target models cannot be build off line [13,14], since the appearance of object is not known in advance. So, we can build a binary classifier that represents the decision boundary between the object and its background to substitute Bhattacharyya distance, and we update the target model by retraining the classifier which is able to incrementally adjust to the changes in the specific tracking environment.

3.2. Learning-based particle filter

In our case, we will propose a leaning-based particle filter. Instead of Bhattacharyya distance, we determine weight value for each particle by a binary classifier. Support vector machines (SVMs, also support vector networks) are supervised classifier with associated learning algorithms. An SVM model is a representation as points in space, and examples of different categories are divided by a clear gap that is as wide as possible. New examples are mapped and predicted to belong to a category based on which side of the gap they fall on. Some evident advantages to SVM include: (1) computationally efficient nonlinear classifiers; (2) can deal with high-dimensional data; (3) have shown very good performance in countless domains.

However, SVM classifier never really output an actual probability. The output of an SVM classifier is the distance of the test instance to the separating hyperplane in feature space. Platt [21] scaling basically fits a sigmoid on top of SVM decision values to scale it to the range of [0,1], which can then be interpreted as a probability.

Given training examples $x_i \in \mathcal{X}$, $i = 1, \dots, l$, labeled by $y \in \{+1, -1\}$, the binary support vector machine computes a decision function $f(x)$ such that $\text{sign}(f(x))$ can be used to predict the label of any test example x . Platt [21] proposes approximating the posterior $P(y = 1|x)$ by a sigmoid function.

$$Pr(y=1|x) \approx P_{A,B}(f) \equiv \frac{1}{1+\exp(Af+B)}, f=f(x) \quad (5)$$

let f_i be an estimate of $f(x_i)$, the best parameter setting $z^* = (A^*, B^*)$ is determined by solving the following regularized maximum likelihood problem with N_+ of the y_i 's positive and N_- negative.

$$\min_{z=(A,B)} F(Z) = - \sum_{i=1}^l (t_i \log(p_i) + (1 - t_i) \log(1 - p_i)) \quad (6)$$

For $p_i = P_{A,B}(f_i)$, and

$$t_i = \begin{cases} \frac{1}{N_-+2} & \text{if } y_i = +1 \\ \frac{N_++1}{N_++2} & \text{if } y_i = -1 \end{cases}, i=1, \dots, l \quad (7)$$

So, according to Eq. (1), for each sample x_t^j , new state is determined by updating the importance weights π_t^j with the likelihood of the observation

$$\pi_t^j = Pr(y=1|x_t^j) \quad (8)$$

And current state of the target is estimated as

$$\left(\bar{x}, \bar{y}, \bar{w}, \bar{h}\right)^T = \sum_{j=1}^J Pr\left(y=1|x_t^j\right) \cdot \left(x_t^j, y_t^j, w_t^j, h_t^j\right)^T \quad (9)$$

According to Eq. (9), we integrate the classifier within a particle filter and the target is modeled by a SVM classifier, we define it as learning-based particle filter in this paper. By accomplishing the resampling procedure according to the probabilistic output of the classifier, we avoid the well known problem of sample depletion [20], which is largely responsible for loss of track. In addition, we propose an idea to update the object model online for the robustness of tracking.

4. Proposed learning-based tracking algorithm

4.1. Initialization of target model

To make the algorithm less sensitive to lighting conditions and compatible with the gray level images, in our experiments, the color histograms are typically calculated in the RGB space using 3_3_3 bins. We defined the vectorized RGB histogram as $p = \{p^{(u)}\} \ u = 1. \dots m$. LBP histogram in (8,1) neighborhood using uniform rotation invariant LBP. We defined the vectorized LBP histogram as a vector $q = \{q^{(u)}\} \ u = 1. \dots n$, in this paper, $m = 27, n = 10$. Combining the color and texture histogram together, we can get the new feature of each sample $v = \{p, q\}$, which is stacked as a single 37-dimensional vector. Then we evaluate a histogram intersection kernel, and after linearization, a single 1369-dimensional vector can be calculated for the following classifier training.

We initialize object model by using a SVM which learns off-line to distinguish between the object and the background. In the first frame, the initial classifier is trained using labeled examples as shown in Fig. 3. We select N_{object} bounding boxes that are closest to the initial bounding box as positive patches. At the same time $N_{background}$ negative examples are extracted by taking regions of the same size as the target window from the surrounding background as shown in Fig. 3(c). All the vectors of the positive and negative samples are stacked as training set. Note that these iterations are only necessary for initialization of the tracker. We use the tracked patches as positive samples to update the training set during the tracking procedure, and the negative sample will be updated too.

According to Eq. (9) and the initial object model, we want to make sense the learning-based particle filter for the object defined as the face in the inner bounding box in Fig. 4(a). Using the video ‘David’ at 647th frame and a target model trained by the method discussed above, we evaluate the SVM classifier for image patches (also called particles of a particle filter) located in the outer bounding box, and obtain a confidence value map. It is similar to the surface plot of the Bhattacharyya coefficient in a small area around the face of the soccer player [4]. From the confidence map given in Fig. 4(b), we can get the position of the bounding box by the particle filter algorithm in the next frame as shown in Fig. 4(c).

4.2. Proposed learning and tracking loop

In this paper, we start the proposed algorithm by training an initial classifier from a labeled training set. Generally, adapting the observation model is the most costly part of the algorithm. However, according to our experience, if the object is distinctive, a simple tracker may already fulfill the requirements [17–19]. Therefore we combine template matching and learning-based particle filtering tracking techniques which can supplement each other.

The proposed tracking loop starts with Lucas–Kanade template matching algorithm [17]: $I_n(x)$ is the n th image in a sequence, $x = (x, y)^T$ are the position, a sub region in $I_0(x)$ is defined as $T(x)$ (also called template). Let $w(x; p)$ denote the deformations parameters of the template. The error summation is over all pixels of the template:

$$\sum_{x \in T} [I_n(w(x; p)) - T(x)]^2 \quad (10)$$

The minimization of Eq. (10) is performed with respect to the warping parameters p [18] and it is a least squares problem. The closed form solution is obtained by taking the partial derivative of Eq. (10) and then setting it to equal zero. The result is the tracked object in $I_n(w(x; p_n))$ and the parameter p_n .

Template will be updated every frame (or every n frames) by a new template extracted from the current image. As different blocks have different amount of texture, the error function must be normalized by the variance of the pixels in the block. However, the problem with this native strategy is the drifting, small errors can be accumulated and the template steadily drifts away from the object. So, we evaluate the tracking results by the classifier for every frame, and the tracked patches with high probabilistic output in Eq. (5) or low cumulative error function in Eq. (10) are used to update the training set. In order to guarantee that the algorithm always properly distinguishes between object and background, the classifier is retrained via fast re-learning after some updating.

When the error of template tracking is higher than the given threshold, learning-based particle filter algorithm will be used for reliable tracking, and the tracking results will also be evaluated with the trained classifier. Similarly, the high confidence tracking results will be added to the training set to retrain the classifier, and the template matching algorithm will start again. In order to handle temporary occlusions of the target, we stop updating the training set if the confidence below a threshold.

As discussed above, the flow chat of the tracking and learning scheme is given in Fig. 5.

The proposed algorithm in Fig. 5 can be given as the following steps:

- 1, Get the input image, the initial position, and the bounding box of the tracking target;
- 2, Extract the feature of objects and background by RGB and LBP histograms;
- 3, Train the object model by SVM classifier for the feature extracted from Step 2;

- 4, Track the object by using Lucas–Kanade template matching algorithm;
- 5, Calculate the error of SSD and confidence of the tracking patch. If under the given threshold h_1 , go to Step 4; otherwise, continue to Step 6;
- 6, Learning-based particle filter tracking at the position given in Step 5. If the confidence of the tracked patch is higher than the given threshold h_2 , go back to Step 4; otherwise, repeat Step 6;
- 7, The tracked patch will be used to update the train set if the threshold is less than error h_1 and higher than confidence h_2 . The classifier will be retrained.

According to the procedures mentioned above, the following snapshots in Fig. 6 show the details of how to track the face in the ‘David’ sequence, which presents challenging lighting, scale and pose changes. The blue bounding box and the green one represent the template matching and particle filter tracking respectively, and the blue cloud is composed by the positions of the sample set of particle filter. In Fig. 7(a), we reveal some typical tracked patches for updating the training set during the tracking procedure. It can be seen that the patches which are added to the training dataset are accuracy and reliable. In Fig. 7(b), we give the confidence of the tracked patch in each frame and the confidence is increasing over time. From the results given in Figs. 6 and 7, we can conclude that the proposed algorithm given in Fig. 5 can work well during challenging changing video.

5. Experiments

In this paper, we provide robust target tracking solution for scenarios, in which the appearance of targets changes over time, and an SVM classifier which can learn online is used to validate the trajectory outputted by the LK track [17]. If the trajectory is not validated, a detector by learning-based particle filtering is performed to capture the target. In contrast to other methods, we do not rely on an offline trained detector and we do change the model of the object during run-time.

We tested our system on several challenging video sequences with challenging lighting, scale and pose changes. For all sequences we have labeled ground truth the object for every frame [15]. Note that the RGB and LBP features we use are fairly robust to scale and orientation changes presented in these clips, so scale changes are considered in our implementation to update the training set. The scale changes can be seen in Fig. 7(a) – the subjects’ head size ranges from 62×65 pixels to 52×55 pixels during the procedure of template tracking.

In the experiment, we utilized RGB and LBP histogram as the object and background features. The Lucas–Kanade template represents the brightness and texture status of the object [17], it gives relatively consistent information about the object. With the features, we get the positive sample set by 50 image patches obtained around the initial position of the object and 150 negative image patches for background as shown in Fig. 3(b) and (c). When 15 positive samples are updated, we will retrain the SVM classifier. The threshold parameters of our method are fixed for all of the experiments after normalizing the error function by variance. The subjective results with empiric threshold $h_1 = 0.003$ and $h_2 = 0.8$

are summarized in Figs. 8–11. Video sequences from TLD dataset named ‘Panda’, ‘Pedestrian 1’, ‘Motocross’, ‘Car Chase’ were tested on the proposed algorithm. In video ‘Panda’, the most challenging problem is scale and pose changes. In video ‘Pedestrian 1’, the background is similar to the object. The next two videos also suffer from the problem of scale changes and occlusion. Figs. 8–11 show screen captures for thousands of the frames. The blue bounding box and the green one represent the template matching and particle filter tracking respectively. From the test results, we can see that the tracking results show good performance in the whole image sequence, and the proposed algorithm is robust to scale, illumination, view changes and even to occlusion.

One of the goals of this work is to demonstrate that the proposed algorithm is robust and stable. The ground truth bounding box and some other outputs of trackers: OnlineBoost [9], Semi-Boost [10], BeyondSemiBoost [11], MIL [12], CoGD [13], TLD1.0 [15], Sun (the proposed algorithm, named by the author's family name) are given in Fig. 12. We give the corresponding relationship between the indicator letters and the color: { ‘gt’, ‘OnlineBoost’, ‘SemiBoost’, ‘Beyond SemiBoost’, ‘MIL’, ‘coGD’, ‘TLD1.0’, ‘Sun’ } ; and { ‘red’, ‘blue’, ‘red’, ‘yellow’, ‘black’, magenta, ‘green’, ‘cyan’ } ; Detection is considered to be correct if its overlap with the ground truth bounding box is larger than 25%. The overlap values for ‘David’, ‘Pedestrian’, ‘Car chase’ and ‘Panda’ sequences are calculated for every frame, in most video frames, with the given ground truth position, the overlap value of our algorithm is larger than 25%. We pick out some typical frames from the video sequence. In Fig. 12(c)–(e), the object is lost for most of other trackers, and the proposed algorithm achieves the satisfactory performance in these changing videos because of the proposed online learning scheme and the global feature by RGB and LBP.

Some objective and quantitative results are summarized in Table 1. The performance of the algorithms is evaluated using precision P and recall R . P is the number of true positives divided by number of all responses and R is the number true positives divided by the number of object occurrences that should have been detected. When comparing a trajectory with ground truth, the trajectory is normalized (shift, aspect and scale correction) so that the first bounding box matches the ground truth. All remaining bounding boxes are normalized with the same parameters.

Table 1 shows the achieved performance evaluated by P/R . In all cases the proposed algorithm outperforms the Online Adaboost and SemiBoost Trackers, and in most cases it outperforms or ties the MIL tracking algorithm. The reason for the superior performance lies in that the proposed algorithm is able to handle ambiguously labeled training examples, which are provided by the global features of RGB and LBP histograms. Rather than extracting only local image patches and taking the risk that the image patch is suboptimal.

Finally, we give additional test results based on image sequence named ‘kitty’ which were captured by a wearable camera eButton [22], and the results are shown in Fig. 13 with the same parameter as those in the previous Figures. The blue bounding box and the green one represent the template matching and learning-based particle filter tracking results respectively. Obviously, the proposed algorithm can also produce robust tracking results for video suffering challenging light changing and motion.

6. Conclusions

Our approach combines tracking, learning and detection in the senses where tracking and detection are independent processes that exchange information by learning. The combination of these ideas has yielded non-rigid active appearance models. The resulting approach leads to a precise and flexible tracker that is quickly applicable to track arbitrary objects in unknown environments. Furthermore, the proposed algorithm is very robust against the drifting problem with the online classifier. Note that the computation of all feature types can be done very efficiently using integral images and integral histograms, which makes classification with tracking still real-time.

There are many interesting ways to improve this work in the future. First of all, the motion model could be replaced with something more sophisticated model for other complicated motion. Furthermore, a more reliable mathematic model for switch modes between template matching and particle filter tracking should be paid more attention to, which could further improve the performance with the presence of severe occlusions, and could also potentially reduce the amount of computation by selecting appropriate tracking templates and updating the training dataset.

Acknowledgements

This work was supported by Fundamental Research Funds for the Central Universities under Grant K50511040008, National Nature Science Foundation of China (NSFC) under Grants 61201290, 61003196, 61105066, 61305041, 61305040, the China Scholarship Council (CSC) and the National Institutes of Health Grants No. R01CA165255 of the United States.

References

1. Yilmaz A, Javed O, Shah M. Object tracking: a survey. *ACM Comp. Surv. (CSUR)*. 2006; 38(4):13.
2. Isard M, Blake A. Condensation – conditional density propagation for visual tracking. *Int. J. Comp. Vis.* 1998; 29(1):5–28.
3. Breitenstein MD, Reichlin F, Leibe B, Koller-Meier E, Van Gool L. Robust tracking-by-detection using a detector confidence particle filter. *12th International Conference on Computer Vision*. 2009:1515–1522.
4. Nummiaro K, Esther K-M, Luc Van G. An adaptive color-based particle filter. *Image Vis. Comp.* 2004; 21(1):99–110.
5. Avidan S. Ensemble tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* 2007; 29(2):261–271. [PubMed: 17170479]
6. Han B, Comaniciu D, Zhu Y, Davis LS. Sequential kernel density approximation and its application to real-time visual tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* 2008; 30(7):1186–1197. [PubMed: 18550902]
7. Lei Y, Ding X, Wang S. Adaboost tracker embedded in adaptive particle filtering. *18th International Conference on, Pattern Recognition*. 2006; 4:939–943.
8. Grabner H, Leistner C, Bischof H. Semi-supervised on-line boosting for robust tracking. *ECCV*. 2008:234–247.
9. Grabner H, Bischof H. On-line boosting and vision. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2006; 1:260–267.
10. Grabner, H., Leistner, C., Bischof, H. *Computer Vision – ECCV*. Springer; Berlin, Heidelberg: 2008. *Semi-Supervised on-line Boosting for Robust Tracking*; p. 234–247.

11. Stalder, S., Grabner, H., Gool, LV. Beyond semi-supervised tracking: tracking should be as simple as detection, but not simpler than recognition. IEEE 12th International Conference on Computer Vision Workshops; ICCV Workshops. 2009;
12. Babenko B, Yang MH, Belongie S. Visual tracking with online multiple instance learning. IEEE Conference on Computer Vision and Pattern Recognition CVPR. 2009:983–990.
13. Yu, Q., Ba Dinh, T., Medioni, G. Computer Vision – ECCV. Springer; Berlin, Heidelberg: 2008. Online Tracking and Reacquisition using Co-trained Generative and Discriminative Trackers; p. 678–691.
14. Kalal Z, Matas J, Mikolajczyk K. P-N learning: bootstrapping binary classifiers by structural constraints. IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2010:49–56.
15. Kalal Z, Mikolajczyk K, Matas J. Tracking-learning-detection. IEEE Trans. Pattern Anal. Mach. Intell. 2012; 34(7):1409–1422. [PubMed: 22156098]
16. Avidan S. Support vector tracking. IEEE Trans. Pattern Anal. Mach. Intell. 2004; 26(8):1064–1072. [PubMed: 15641735]
17. Schreiber D. Robust template tracking with drift correction. Pattern Recog. Lett. 2007; 28(12): 1483–1491.
18. Bouguet, JY. Technical Report. Intel Microprocessor Research Labs; 1999. Pyramidal implementation of the Lucas Kanade feature tracker description of the algorithm.
19. Ishikawa MT, Baker S. The template update problem. IEEE Trans. Pattern Anal. Mach. Intell. 2004; 26:810–815. [PubMed: 18579941]
20. Klein DA, Schulz D, Frintrop S, Cremers AB. Adaptive real-time video-tracking for arbitrary objects. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2010:772–777.
21. Platt J. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. Adv. Large Margin Classif. 1999; 10.3:61–74.
22. Bai, Y., Li, C., Jia, W., Li, J., Mao, Z-H., Sun, M. Designing a wearable computer for lifestyle evaluation. 38th Annual Northeast Bioengineering Conference; Philadelphia, PA. March 16–18; 2012. p. 243–244.
23. Sun W, Guo BL, Li DJ, Jia W. Fast single-image dehazing method for visible-light systems. Opt. Eng. 2013; 52(9):093103.
24. Sun W. A new single image fog removal algorithm based on physical model. Int. J. Light Electron Opt. 2013; 124(21):4770–4775.
25. Sun, W., Han, L., Guo, B., Jia, W., Sun, M. A fast color image enhancement algorithm based on max intensity channel. J. Mod. Opt. 2014. <http://dx.doi.org/10.1080/09500340.2014.897387> (ID: 897387)

Fig. 1.
Ground truth object bounding box in 'David' video sequence.

Fig. 2. Histogram of David video sequence and Bhattacharyya distance with the initial object model. (a and c) RGB and LBP histogram of the ‘David’ video sequence respectively. (b and d) Bhattacharyya distance of RGB and LBP histogram with the first frame respectively.

Fig. 3. Initial classifier using labeled examples. (a) Initial bounding box, (b) positive patches, (c) negative patches, (d) bounding box in LBP image.

Fig. 4. Surface plot of the confidence value of a small area around the face. (a) The given area, (b) the confidence value around the area, (c) the position calculated by particle filter for tracking the following frames.

Fig. 5.
Flow chart of the proposed algorithm.

Fig. 6.

Snapshots with details of how to track the face in the ‘David’ sequence. From left to right, the frame number of the top row is 000/449/450/472/473/498, and the bottom row is 513/522/605/683/701/756.

Fig. 7. Results on ‘David’ sequence (a) Snapshots of learning templates. (b) Confidence of the tracking patches.

Fig. 8.

Results given by the proposed algorithm on 'Panda' sequence. From left to right, the frame number of the top row is 0001/0098/0099/0138/0190/0214/0291/0358, and the bottom row is 0469/0557/0611/0709/0841/0907/0947/0972.

Fig. 9.

Snapshots from the 'Pedestrian 1' sequences with the objects marked by the bounding box.

From left to right, the frame number of the top row is 009/019/029/039/049/059/069, and the bottom row is 079/089/099/109/119/128/139.

Fig. 10.

Snapshots from the 'Motocross' sequences with the objects marked by the bounding box.

From left to right, the frame number of the top row is 009/049/079/089/099/109/119, the medium row is 129/179/209/229/249/259/289, and the bottom row is 299/319/329/339/359/369/399.

Fig. 11.

Snapshots from the 'Car Chase' sequences with the objects marked by the bounding box. From left to right, the frame number of the top row is 009/029/039/049/069/079/109, the medium row is 119/149/159/199/209/219/239, and the bottom row is 269/349/359/379/389/399/409.

Fig. 12.

Ground truth bounding box and outputs of trackers. From left to right, the frame number is 446/107/043/305/364 in ‘David’, ‘Pedestrian 1’, ‘Motocross’, ‘Panda’, and ‘Car Chase’ sequence.

Fig. 13.

Tracking results on eButton images with challenging lighting and motion. From left to right, the frame number of the top row is 001/006/030/137/154, and the bottom row is 177/220/221/259/260.

Table 1

Performance of tracking algorithms evaluated by precision P and recall R .

Sequence	Frame	Gt	OnlineBoost	SemiBoost	BeyondSemi	MIL	coGD	TLD1.0	Sun
David	761	1.000/1.000	0.412/0.288	0.346/0.346	0.323/0.244	0.150/0.150	1.000/0.999	1.000/1.000	0.954/0.954
Pedes. 1	140	1.000/1.000	0.606/0.143	0.479/0.329	0.292/0.100	0.693/0.693	1.000/1.000	1.000/1.000	0.929/0.929
Car Chase	410	1.000/1.000	0.791/0.668	0.864/0.476	0.702/0.636	0.761/0.834	0.976/0.992	1.000/0.922	0.902/0.989
Panda	1000	1.000/1.000	0.947/0.947	1.000/0.308	0.990/0.292	0.926/0.926	0.336/0.334	0.895/0.859	0.973/0.973