Front. Electr. Electron. Eng. China 2008, 3(3): 274–282
DOI 10.1007/s11460-008-0045-z

RESEARCH ARTICLE

Jiangtao WANG, Jingyu YANG

Relative discriminant coefficient based multi-cue fusion for robust object tracking

Abstract In visual tracking, integrating multiple cues will increase the reliability and robustness of the tracking system in situations where no single cue is reliable. In this paper, a novel multi-cue based tracking method is presented under the particle filter framework. Considering both practical distance and Bhattacharyya distance between particles and the target, a parameter called relative discriminant coefficient (RDC) is designed to measure the tracking ability for different features. Multi-cue fusion is carried out in a reweighing manner based on this parameter. Experimental results demonstrate the high robustness and effectiveness of our method in handling appearance changes, cluttered background, illumination changes and occlusions.

Keywords visual tracking, multi-cue fusion, particle filter

1 Introduction

Object tracking in video sequences is one of the main issues of computer vision. Tracking object is useful for many vision based applications, including visual surveillance, human-computer interfaces, traffic monitoring systems and video compression, and so forth. These applications all require a robust tracking mechanism operated online or offline. To achieve robustness and to reduce uncertainty in object racking, over the past few years tremendous efforts have been devoted to enhancing visual tracking performance by designing various tracking programs and making use of different tracking features [1–6]. Great progress has been made as reported in the literature.

However, object tracking still suffers from a lack of robustness due to a dynamically changing environment. Though it is noted that the fusion of multiple cues will lead to increased reliability of the tracking system, most of the current tracking algorithms are based on single cue determined a priori and are, therefore, often limited to a particular environment. In this case, the fixed feature is either chosen at random or sometimes preliminary experiments are conducted to determine which feature to use. Adopting multiple-cues in tracking the object has attracted considerable interest from researchers to improve robustness.

Various features can be used to represent objects, such as color, depth, motion and texture. Since most of the computer vision problems are ill-posed, more features increase the robustness of solutions. Up to now, much literature has been published about the fusion of multiple-cues. For a tracking system that utilizes multiple-cues, the method by which the cues are integrated is a key issue. A straightforward approach reported in Refs. [7,8] used all cues in parallel and treating them as equivalent channels. In their work, different cues are combined directly in a likelihood manner. However, one limitation of this method is that it does not take account of the discriminant ability of the cue. Democratic integration is an architecture that allows for the tracking of objects through the fusion of multiple adaptive cues in a self-organized fashion. This method was proposed by Triesch and Malsburg in Ref. [9]. It is explored further in Refs. [10,11]. In this framework each cue creates a 2-dimensional cue report or a saliency map, the cues fusion is carried out by resulting a fused saliency map that is computed as a weighted sum of all the cue reports. Integration was extended by Spengler and Schiele in Ref. [10] by using a particle filter for tracking multiple targets. In Ref. [12] Pérez et al. presented a particle filter based visual tracker that fuses three cues: color, motion and sound. In their work, color cues serve as the main visual cue, and according to the scenario under consideration, color cues are fused with either sound localization cues or motion activity cues. A partitioned sampling technique is applied to combine different cues. The particle resampling is not carried out on the whole feature space but in each single feature space separately. This technique

Received September 18, 2007; accepted December 18, 2007

Jiangtao WANG (✉), Jingyu YANG
Department of Computer Science, Nanjing University of Science and Technology, Nanjing 210094, China
E-mail: jiangtaoking@126.com

increases the efficiency of the particle filter. However, in their case only two cues can be used simultaneously, which restricts the flexible selection of cues and the extension of the method. Differing from the above-mentioned fusion method, Loy et al. fused different cues based on relative entropy [13], or the Kullback-Leibler distances between the probability density function (PDF) of the cue and the PDF of the target. In Ref. [14] Wu formulated the problem of integrating multiple-cues for robust tracking as the probabilistic inference problem of a factorized graphical model. To analyze this complex graphical model, a modified method is taken to approximate the Bayesian inference. Interestingly, the analysis reveals a co-inference phenomenon of multiple modalities, which illustrates the interactions among different cues, i.e., one cue can be inferred iteratively by other cues. An efficient sequential Monte Carlo tracking algorithm is used to integrate multiple visual cues, in which the co-inference of different modalities is approximated.

In this paper, we concentrate on multi-cues based object tacking in a particle filter framework. Our ultimate goal is to develop a multi-cue based particle filter technique which can combine various cues online and adaptively, so as to track object robustly even in complex dynamic environment. In our work, the relative discriminant coefficients of different cues are computed to measure the tracking ability of them. Each cue is weighted based on this measure, and then all the cues are fused in a weighted sum manner.

The remainder of the paper is organized as follows. Section 2 briefly outlines theoretical background of particle filter (PF) tracking framework. Section 3 presents the target dynamics model. Section 4 gives the feature selection and observation models. Section 5 introduces our proposed data fusion tracker. This section is concluded with a summary of the tracking algorithm. Section 6 presents tracking scenarios and illustrates the performance of the tracking algorithm under a variety of conditions. The usefulness of each localization cue and their combined impact are evaluated. Finally, we conclude the paper in Sect. 7 with a summary.

2 Particle filter review

As mentioned above, our method is based on the tracking framework of the PF [15–17]. For completeness, we summarize the main idea behind the particle filter in this section.

Visual tracking is often posed as a Bayesian sequential estimation problem. In this sense, Kalman filter and the extended Kalman filter have been highly popular in designing visual trackers. However, in circumstances where heavy background clutter leads to multi-model distributions, or the system is highly non-linear and the densities are highly non-Gaussian. These techniques often fail because of its assumption of Gaussian models or linear dynamic system. Particle filter relaxes the Gaussian and linear modeling assumption of Kalman filter by employing samples to represent probability densities and invoking Monte Carlo techniques to simulate density propagation.

Particle filters, also known as sequential Monte Carlo methods (SMC), Bootstrap filters, Condensation trackers and so on, are complicated model estimation techniques based on simulation within a Bayesian framework. Denote by $\boldsymbol{X}_t = \{\boldsymbol{x}_0,\boldsymbol{x}_1,\ldots,\boldsymbol{x}_t\}$, $\boldsymbol{Y}_t = \{\boldsymbol{y}_0,\boldsymbol{y}_1,\ldots,\boldsymbol{y}_t\}$ respectively the hidden state and observation until time t. According to Bayesian estimation theory, the optimal estimate of $\boldsymbol{x}_t$ is given by the posterior mean $E[\boldsymbol{x}_t|\boldsymbol{Y}_t]$. Given the posterior PDF $p(\boldsymbol{x}_{t-1}|\boldsymbol{Y}_{t-1})$ at time $t-1$, then the current posterior PDF can be obtained by the following two steps:

Prediction step

$$p(\boldsymbol{x}_t|\boldsymbol{Y}_{t-1}) = \int_{\boldsymbol{x}_{t-1}} p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})p(\boldsymbol{x}_{t-1}|\boldsymbol{Y}_{t-1})\mathrm{d}\boldsymbol{x}_{t-1}. \quad (1)$$

Update step

$$p(\boldsymbol{x}_t|\boldsymbol{Y}_t) = \frac{p(\boldsymbol{y}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{Y}_{t-1})}{p(\boldsymbol{y}_t|\boldsymbol{Y}_{t-1})}. \quad (2)$$

In the prediction step, the prior $p(\boldsymbol{x}_{t-1}|\boldsymbol{Y}_{t-1})$ is propagated to $p(\boldsymbol{x}_t|\boldsymbol{Y}_{t-1})$ through a dynamical system model $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$. The predicted state is then corrected by an observation likelihood function $L(\boldsymbol{Y}_t|\boldsymbol{x}_t)$ in the update step. In order to avoid the integral in Eq. (1), the key idea of particle filtering is to approximate the posterior PDF by a weighted sample set $S = \{(s^{(n)},b^{(n)})|n = 1,2,\ldots,N\}$. Each sample s represents one hypothetical state of the object, with a corresponding discrete sampling probability b, where $\sum_{n=1}^{N} b^{(n)} = 1$. The mean state of an object is estimated at each time step by

$$\mathrm{E}(S) = \sum_{n=1}^{N} b^{(n)} S^{(n)}. \quad (3)$$

The sample weight corresponding to each sample is proportional to the observation likelihood function, and we would present the observation likelihood function in Sect. 4.

3 Target dynamics model

To track an object sequentially in video, we initially choose a region which defines the object. The shape of this region is fixed a priori through the definition of an ellipse or a rectangular box. In our case we choose a rectangular box characterized by the state vector

$$\boldsymbol{x} = \left\{x,y,x',y',s_x,s_y,s'_x,s'_y\right\}, \quad (4)$$

where x and y denote the centroid of the ellipse or rectangle, x' and y' are the velocities of the centroid, s_x and s_y denote the length of the half axes or sides and s'_x and s'_y denote the velocities of s_x and s_y. A first order auto-regression (AR) dynamics is chosen on these parameters. This has the form

$$\boldsymbol{x}_t = \boldsymbol{A}\boldsymbol{x}_{t-1} + \boldsymbol{B}v_t, \tag{5}$$

where v_t is a multivariate normal distribution; the matrix $\boldsymbol{A}$ defines the deterministic component; $\boldsymbol{B}$ denotes the stochastic component. It is straightforward to extend this model to second order if it is required to capture more complex dynamical motions. For the time being we use an Ad-hoc model composed of three independent constant velocity dynamics on $\boldsymbol{x}_t$, $\boldsymbol{y}_t$ and s_t with respective standard deviations 1 pixel/frame, 1 pixel/frame, and 0.1 frame^{-1}.

4 Feature selection and observation model

Choosing appropriate features plays important roles for robust tracking in the particle filter framework. Feature selection relies on the availability of the object feature. For simplicity, in this work, we select two popular features: color and edge information as the representing feature. Our choice of features is based on their feasibility for robust visual tracking and their adaptability to our probabilistic framework. All the feature models are based on histograms for it has the properties that they allow for some changes in the object appearance without changing the histogram.

4.1 Color model

Color models are widely used in object tracking as they achieve robustness against non-rigidity, rotation and partial occlusion. We take a histogram-based non-parametric color model which is originally introduced by Comaniciu et al. for the mean-shift based object tracking [6]. We determine the color distribution inside a rectangular region R(x) centered at the location x, which is denoted as $p(x) = \{p_u(x)\}$, $u = 1,2,\ldots,m$, where m is the number of bins. In our experiments, the histograms are typically calculated in RGB space using $8 \times 8 \times 8$ bins. To increase the reliability of the color distribution and lower the weight of the pixels, smaller weights are assigned to pixels that are closer to the border of the target region by employing a kernel weighting function. More specific $p(\boldsymbol{x})$ can be obtained as:

$$p_u(\boldsymbol{x}) = E\sum_{i=1}^{n_h} k\left(\left\|\frac{\boldsymbol{y}-\boldsymbol{x}_i}{h}\right\|^2\right)\delta\left[b(\boldsymbol{x}_i)-u\right], \tag{6}$$

where δ is the Kronecker delta function, k is a kernel function, such as the Epanechnikov kernel. Normalization factor E ensures that the sum of the bins is one. Given a reference histogram q that defines the target model, and histogram $p(\boldsymbol{x})$ extracted from a candidate state $\boldsymbol{x}$, the similarity between them is evaluated based on the Bhattacharyya coefficient

$$d_b[p(\boldsymbol{y}),q] = \sqrt{1-\sum_{u=1}^{m}\sqrt{p_u(\boldsymbol{y})q_u}}. \tag{7}$$

Upon this, the corresponding weight can be defined as follows:

$$b_c \propto \text{likelihood}_c = \frac{1}{\sqrt{2\pi}\sigma_c}\exp\left(-\frac{d_b^2}{2\sigma_c^2}\right), \tag{8}$$

where σ_c is the standard deviation which specifies the Gaussian noise in the measurement.

If the image is gray, we construct the intensity histogram in the same way as one channel of the color space.

4.2 Edge model

Edge feature has been proven useful for modeling the shape of the object to be tracked. Here we describe the edge using a histogram based on the estimated gradient orientation. Let the target be represented by a rectangular centered at (x, y), the width and height of it are h and w. Then we can construct the gradient orientation histogram as:

$$H_g(k) = \sum_{x=1}^{h}\sum_{y=1}^{w} k(x,y)\delta\left[\theta(x,y)-z\right], \quad z = 1,2,\ldots,Z, \tag{9}$$

where z is the histogram bin index, $\theta(x,y)$ is the gradient orientation of pixel at (x, y), δ is the Kronecker delta function, and $k(i, j)$ is the kernel function emphasizing the spatial arrangement of the features in the image.

In order to remove the burden of noise, the gradient orientation is filtered to include only gradients with magnitude m above a predefined threshold T. In our case, T is defined as follows:

$$T = w\max\ (im) + (1-w)\ \text{mean}\ (im). \tag{10}$$

So we have

$$\hat{\theta}(i,j) = \begin{cases} \theta(i,j), & m(i,j) > T, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

For each cue, given two histograms q and p that describe the target model of the current frame and the reference frame in this feature space respectively, by using Bhattacharyya coefficient, the distance between them can be computed as:

$$\rho(p,q) = \sum_{k=1}^{M}\sqrt{p_k q_k}, \tag{12}$$

$$d_b(p,q) = \sqrt{1-\rho(p,q)}. \quad (13)$$

Based on the Bhattacharyya distance, the weight for particles in this feature space can be defined as:

$$b_e \propto \text{likelihood}_e = \frac{1}{\sqrt{2\pi}\sigma_e}\exp\left(-\frac{d_b^2}{2\sigma_e^2}\right), \quad (14)$$

where σ_e is the standard deviation which specifies the Gaussian noise in the measurement.

After selecting the tracking feature and constructing the target model, we can use the particle filter tracking method described in Ref. [16] to estimate the target state in single cue cases.

5 Relative discriminant coefficient based multi-cues fusion

When we use multi-cues for object tracking, we should decide how to fuse these cues efficiently. Generally, there are two choices: fusing different features with constant weights and fusing features with time-varied weights according to the change of tracking states. In practice, different features have different discriminant capacities despite for the same target, and the discriminant capacities may change in time. Thus constant weights may lead to poor tracking performance, so the time-varied weighting idea is applied here. In the tracking process, we suppose that features with better quality have bigger weight and vice versa.

The Bhattacharyya distance defined in Ref. [18] shows the similarity of the feature distribution between the particle sample and the reference model. And it is a popular technique [19] for measuring the feature's performance in multi-cues based object tracking. In the single cue case, particles with smaller Bhattacharyya distances approach the target more closely. However, this is not tenable when multiple-cues are applied due to that different features have different sensitivity to appearance changes of the target, and the Bhattacharyya distance cannot describe the similarity between particles and target veritably. To evaluate the particle's quality straightforwardly regardless of the feature space it belongs to, another distance called practical distance is designed. Given two rectangles R_1 and R_2 centered at (x_1, y_1) and (x_2, y_2), as shown in Fig. 1(a), they represent the reference target and the sample particle (target candidate) respectively. We define the practical distance as the superposition degree between them. Generally, three factors should be considered, i.e., the center distance between two rectangles (Fig. 1(a)), the pose angle difference between them (Fig. 1(b)) and the size (area) difference between them(Fig. 1(c)). To have a more explicit understanding, here we suppose that the object do not perform rotary motion. Then when computing the practical distance we only need take account of two factors: the position distance and the size difference between them. As shown in Fig. 1, this can be accomplished through Eq. (15) as:

$$d_p(R_1,R_2) = \sqrt{d_x^2 + d_y^2 + w_1|h_2-h_1| + h_2|w_1-w_2|}, \quad (15)$$

where $d_x = x_1 - x_2$ and $d_y = y_1 - y_2$ are the center biases between them, w_1, h_1, w_2, h_2 are the width and height of these two rectangles respectively. In the computing of practical distance, the feature spaces which the particle belongs to are not considered, so this distance gives the particle quality in a straightforward manner for different feature spaces.

In order to estimate the tracking ability of various cues, we first compute the Bhattacharyya distances in all feature spaces and the practical distance for each particle. Then in each feature space, we rank all the particles in a ascend sequence according to the Bhattacharyya distance for this cue, and the first L particles are selected as the training samples to evaluate the tracking ability of this cue. For each particle in the training samples the performance ratio

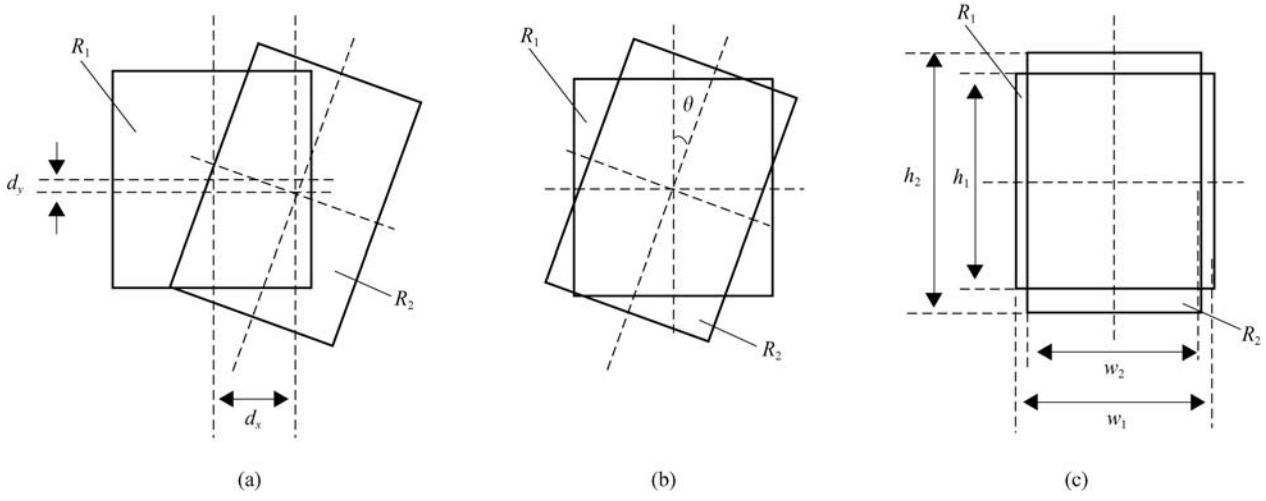

Fig. 1 Physical distance for two particles

t of it in this feature space is computed as:

$$t_i = \frac{d_b^i(p,R)}{d_p^i(p,R)}, \quad i = 1,2,\ldots,L. \tag{16}$$

The mean performance ratio for this feature space is $\hat{t} = \frac{1}{K}\sum_{i=1}^{K} t_i$, $\hat{t}$ indicates the cue's discriminant ability. To measure the difference among these cues, we define the relative discriminant coefficient (RDC) of cue f as:

$$\mathrm{RDC}_f = \frac{\hat{t}_f}{\sum_{i=1}^{F} \hat{t}_i}, \quad f = 1,2,\ldots,F, \tag{17}$$

where F is the number of the cues we adopt. Our objective is to fuse multiple-cues adaptively according to the current cue weight, which can be achieved by generating new particle weight as:

$$B^{(n)} = \prod_{a=1}^{F} (b_a^{(n)})^{\mathrm{RDC}_a}. \tag{18}$$

Then the new weight is normalized to have a sum of 1:

$$\hat{B}^{(n)} = \frac{B^{(n)}}{\mathrm{sum}(B)}. \tag{19}$$

Based on Eq. (18) all cues are combined into one through fusing the particle weight in different feature spaces. This new weight is embedded in the particle filter tracking framework described in Ref. [4] to locate the pedestrian. At the start frame, the reference models are initialized manually, and each cue weight is also computed by evaluating the performance of training samples. In the next coming frame, the initialized cue weights are used to estimate the target state in the particle filter framework, and then the reference models are updated by the currently estimated target models, and the cue weights are also updated based on the new reference models. Thus we update reference models and cue weights at $n-1$ frame, then we use it for the pedestrian tracking at n $(n>1)$ frames, through which different cues can be weighted adaptively. The general algorithm is given as follows:

Initialization

For $l = 1, 2,..., N$, generate samples $\{x_0^{(l)}\}$ from the prior distribution $p(x_0)$. And initialize different features' RDC $\{\mathrm{RDC}_0^{(k)}\}$ for $f = 1,2,\ldots,F$. Set initial weights as $b_0^{l,f} = 1/N$.

For $t = 1{:}T$

Resample

Select N states (they can repeat) starting from $\{x_{t-1}^{(l)}\}$, and carry out the following procedure:

1) For each particle, calculate the cumulative probability as $c_t^0 = 0$, $c_t^l = c_t^{l-1} + b_t^l$.

2) We generate a random number $r \in [0,1]$ that is uniformly distributed, and find the smallest j for which $c_t^j > r$.

3) The selected state is $\tilde{x}_{t-1}^l = x_{t-1}^j$.

Prediction

1) Spread the states $\{\tilde{x}_{t-1}^i | i = 1,2,\ldots,M\}$ to $\{x_t^i | i = 1,2,\ldots,M\}$ using the state dynamic model.

2) For each new state x_t^i, find their corresponding weights in each feature space based on the likelihood models given in Sect. 4;

3) Normalize the weights as

$$\hat{b}_t^{l,k} = \frac{1}{N}\sum_{h=1}^{N} b_t^{h,k}, \quad k = 1,2,\ldots,M.$$

Multi-cues fusion

Combine multiple-cues through $b_t^l = \sum_{i=1}^{M} F_{t-1}^i \hat{b}_t^{l,i}$, and normalize it as $\hat{b}_t^l = \frac{1}{N}\sum_{i=1}^{N} b_t^l$.

State estimation

$$\hat{\boldsymbol{X}}_t = \sum_{i=1}^{N} \hat{b}_t^i x_t^i.$$

Update the RDC

Update $\{F_t^{(k)}\}$ for $k = 1,2,\ldots,M$ based on $\hat{\boldsymbol{X}}_t$.

End

6 Experiment and analysis

This section evaluates the performance of the proposed approach in real-word video sequences. In the test, four distinctly different tracking scenarios:

1) Object tracking with changing illumination.
2) Object tracking with different poses.
3) Object tracking under occlusions.
4) Object tracking in infrared videos is considered.

In order to highlight the difference between tracking results with different cues, both trackers based on single cues and combined cues are used. All tracking results are obtained using 200 particles. When multi-cues are used for tracking, 20 particles are selected as the training sample to weight different cues. Implemented in the Matlab 6.5 environment, our algorithm runs at 3 to 7 frame/s on an AMD 2.0 GHz CPU. The tracking result is shown in the figures with a white bounding box. We list the most difficult and representative four video sequences in the following subsections.

6.1 Tracking a walking person under illumination changes

The first video sequence shown in Fig. 2 is a person walking in a street (this sequence is available online at http://www.cs.technion.ac.il/Labs/Isl/rudzsky/site/db.htm). The challenge of this video sequence includes highly changing illumination caused by the shadows of trees, and rapid shape changes of no-rigid human body. For comparison, we use three particle filter algorithms, two of which use the single cue of color and edge respectively and the other uses combined-cues proposed in this work. The tracking results show

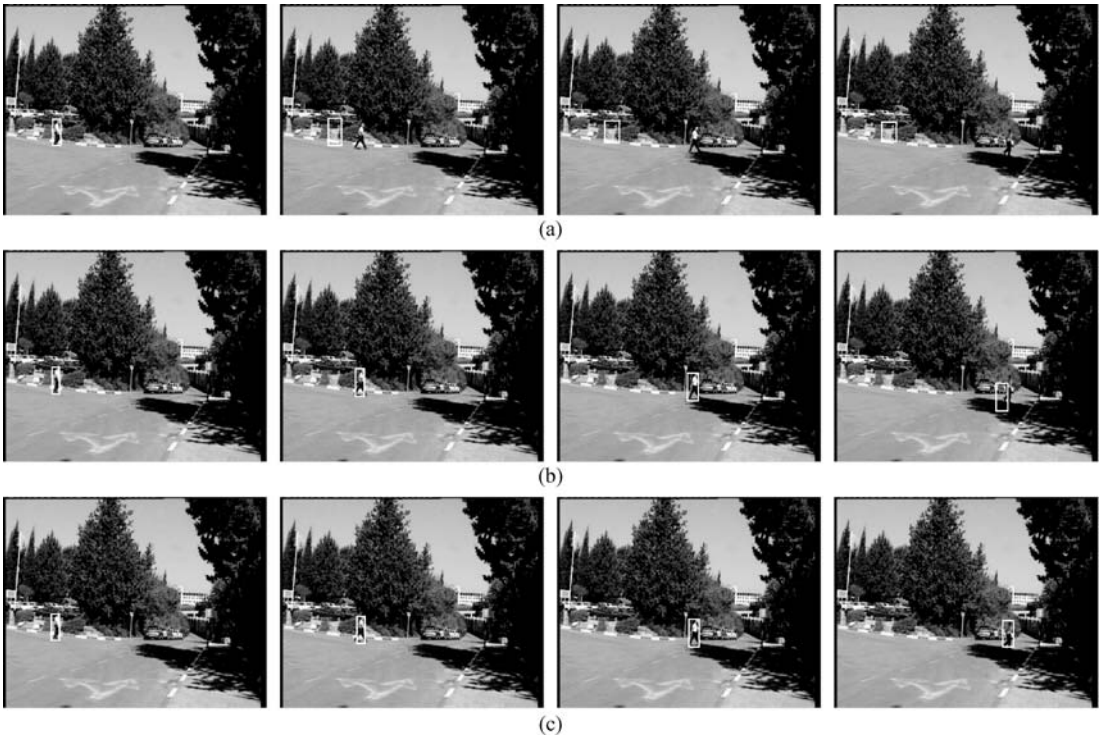

Fig. 2 Tracking a person against an illuminate changing background. (a) Color cue; (b) edge cue; (c) our method (From left to right they are frames 3, 59, 117 and 161)

that the tracker with the color cue is distracted quickly by background regions with similar color (Fig. 2(a)); though the tracker with edge cue can locate the person, large errors appear when the person moves into region of the tree shadows (Fig. 2(b)). Compared with tracker with single cue, combined-cues based tracker can improve the tracker's performance. Even under illumination changes it can track the person efficiently (Fig. 2(c)).

Fig. 3 Tracking a face with different poses. (a) Intensity cue; (b) edge cue; (c) our method (From left to right they are frames 343, 387, 402 and 435)

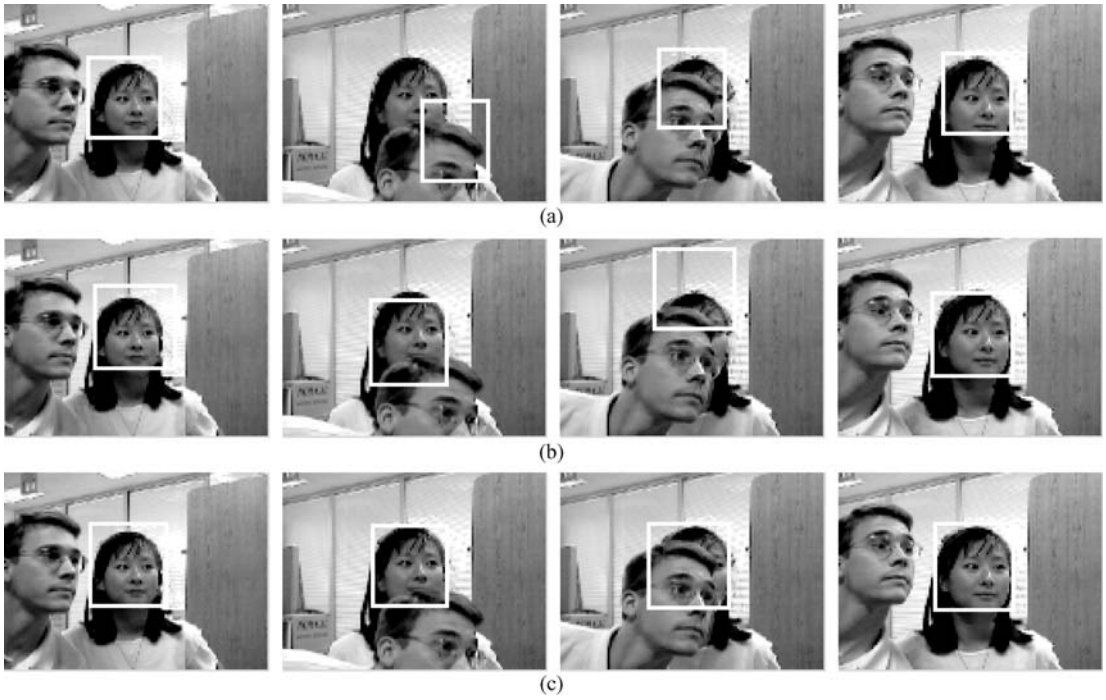

Fig. 4 Tracking the face under occlusions. (a) Color cue; (b) edge cue; (c) our method (From left to right they are frames 419, 442, 459 and 472)

6.2 Tracking a man's face under large pose variations

In Fig. 3, another difficult tracking situation is tested. In this video a man walks in an office while changing his head pose dramatically (this sequence is available online at http://www.cs.toronto.edu/~dross/ivt). The difficulty in this sequence is that the facial appearance changes significantly due to pose variations. In this test, color feature is replaced by intensity feature. If the feature model is single, the tracker will lead to large errors after a short period of time; especially when there is a drastic pose variation. But the tracker with adaptively combined cues can stably follows the face throughout the sequence.

6.3 Tracking a woman's face under occlusions

Object tracking under occlusions is one of the main bottleneck problems in computer vision. In this video sequence the head of the woman is occluded by a man who passes in front of her (this sequence is available online at http://www.ces.clemson.edu/~stb/research/headtracker/seq/). Figure 4 gives some tracking results of this test. We can see that trackers with single cue fail when the occlusion takes place (Fig. 4(a) and (b)). By contrast, the tracker with combined cues obtains more perfect results. As shown in Fig. 4(c), our algorithm is robust to these difficult conditions.

To further illustrate the importance of fusing multiple cue adaptively, we computed the tracking error in x-y coordinate Error_{xy} of the three algorithms. Here, the error is computed as:

$$\text{Error}_{xy} = \sqrt{(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2}, \tag{20}$$

where $(\hat{x}_i, \hat{y}_i)$ is the estimated target position and (x_i, y_i) is the real target position obtained manually. Figure 5 shows the error curves. They indicate that, smaller errors occur by using the combined feature model.

6.4 Tracking a moving person in infrared video

Generally it was found that infrared images enable better image segmentation, but their tracking performance with current algorithms is poorer [20], for infrared image cannot provide as much information as visible ones about objects. In this test video a man walks from afar toward the camera (this sequence is available online at http://

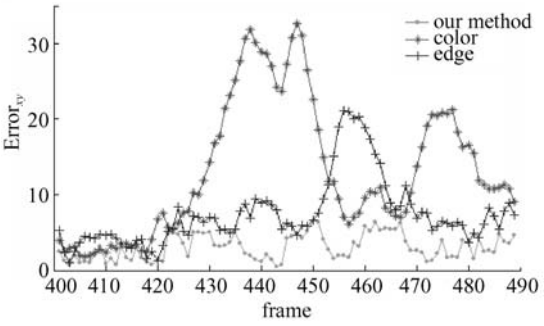

Fig. 5 Tracking errors of three trackers

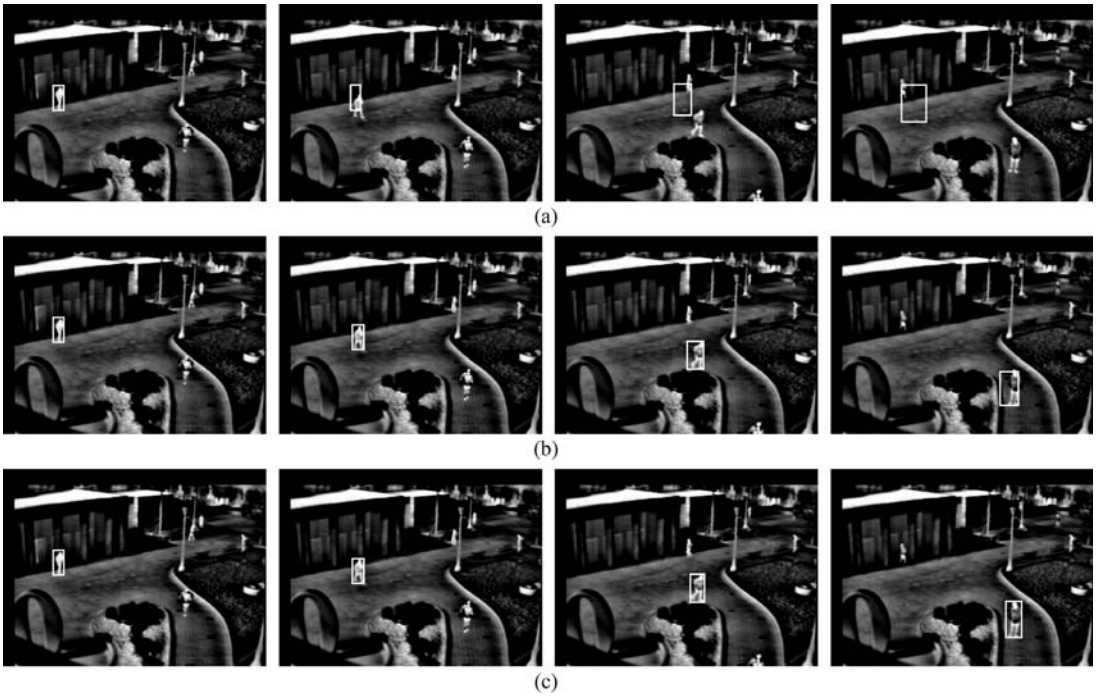

Fig. 6 Tracking a person in an infrared sequence. (a) Intensity cue; (b) edge cue; (c) our method (From left to right they are frames 2041, 2147, 2405 and 2715)

www.cse.ohio-state.edu/otcbvs-bench/). Along the walking process, the intensity and shape for the pedestrian region all change greatly. Figure 6 shows the tracking performance of three trackers with the intensity cue, the edge cue and combined cues respectively. The particle filter with the intensity cue (Fig. 6(a)) is distracted by background regions with similar intensity, and loses the target quickly. Though the particle filter with the edge cue (Fig. 6(b)) can track the pedestrian successfully; it loses some scale information of the pedestrian. Compared with the tracking results with single cues, the particle filter with multiple-cues fused by our proposed method (Fig. 6(c)) performs much better and tracks the pedestrian both in position and scale nicely.

Figure 7 shows RDC curves for intensity and gradient cues combined with the presented approach. It can be seen that the RDC of both cues vary along the tracking time, and in general the RDC of the gradient is bigger than the weight of intensity, which implies that the gradient feature is more reliable than the intensity feature. This is supported by the practice that pedestrian tracking with gradient cues has more accurate results than with intensity cues.

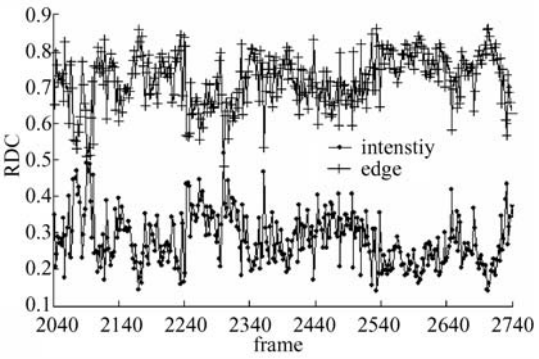

Fig. 7 RDC curves for intensity and edge cues

7 Conclusions

We have presented a multi-cues fusion algorithm to improve the performance of particle filter based object tracking. In this fusion algorithm both the feature distance and the physical distance for particles are considered and the relative discriminant coefficients are defined to evaluate the tracking ability for different cues. Compared with previous multi-cues fusing methods, the strong points of our method are as follows: First, different cues' tracking ability can be measured distinctly; second, the tracking frame in which new cues can be added or removed flexibly. As demonstrated by four challenging video sequences, our algorithm can track objects robustly in the presence of illumination changes, large pose variations, expression changes, and partial or full occlusions.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant Nos. 60472060 and 60632050).

References

1. Collins R T, Liu Y, Leordeanu M. Online selection of discriminative tracking features. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2005, 27(10): 1631–1643
2. Han B, Davis L. Robust observations for object tracking. In: Proceedings of the 12th International Conference on Image Processing. Washington: IEEE Press, 2005, 2: 442–445
3. Yang C, Duraiswami R, Davis L. Efficient mean-shift tracking via a new similarity measure. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Washington: IEEE Press, 2005, 1: 176–183
4. Comaniciu D, Ramesh V, Meer P. Real-time tracking of non-rigid objects using mean shift. In: Proceedings of the 1st Conference on Computer Vision and Pattern Recognition. Washington: IEEE Press, 2000, 2: 142–149
5. Jepson A D, Fleet D J, EI-Maraghi T F. Robust online appearance models for visual tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2003, 25(10): 1296–1311
6. Comaniciu D, Ramesh V, Meer P. Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2003, 25(5): 564–577
7. Li P, Francois C. Image cues fusion for object tracking based on particle filter. In: Francisco J, Bruce A, eds.Articulated Motion and Deformable Objects. Lecture Notes in Computer Science. Berlin: Springer-Verlag, 2004, 3179: 99–107
8. Wang H, Suter D. Efficient visual tracking by probabilistic fusion of multiple cues. In: Proceedings of 18th International Conference on Pattern Recognition. Washington: IEEE Press, 2006, 4: 892–895
9. Triesch J, Von der Malsburg C. Democratic integration: self-organized integration of adaptive cues. Neural Computation, 2001, 13(9): 2049–2074
10. Spengler M, Schiele B. Towards robust multi-cue integration for visual tracking. Machine Vision and Applications, 2003, 14(1): 50–58
11. Shen C, Van den Hengel A, Dick A. Probabilistic multiple cue integration for particle filter based tracking. In: Proceedings of the 7th Digital Image Computing: Techniques and Applications. Sydney: The University of Queensland, 2003, 399–408
12. Pérez P, Vermaak J, Blake A. Data fusion for tracking with particles. In: Proceedings of the IEEE, 2004, 92 (3): 495–513
13. Loy G, Fletcher L, Apostoloff N, et al. An adaptive fusion architecture for target tracking. In: Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition. Washington: IEEE Press, 2002, 261–266
14. Wu Y, Huang T S. Robust visual tracking by integrating multiple cues based on co-inference learning. International Journal of Computer Vision, 2004, 58(1): 55–71
15. Gordon N, Salmond D, Smith A. A novel approach to non-linear/non-Gaussian Bayesian state estimation. In: IEEE Proceedings-F, 1993, 140(2): 107–113
16. Nummiaro K, Koller-Meier E, Gool L V. An adaptive color-based particle filter. Image and Vision Computing, 2003, 21(1): 99–110
17. Isard M, Blake A. Condensation–conditional density propagation for visual tracking. International Journal of Computer Vision, 1998, 29(1): 5–28
18. Bhattacharyya A. On a measure of divergence between two statistical populations defined by their probability distributions. Bulletin of the Calcutta Mathematical Society, 1943, 35: 99–110
19. Brasnett P, Mihaylova L, Bull D, et al. Sequential Monte Carlo tracking by fusing multiple cues in video sequences. Image and Vision Computing, 2007, 25(8): 1217–1227
20. Goubet E, Katz J, Porikli F. Pedestrian tracking using thermal infrared imaging. Mitsubishi Electric Research Laboratories Technical Report TR126, 2005