Probabilistic Tracking with Adaptive Feature Selection

Hwann-Tzong Chen[1,2] Tyng-Luh Liu[1]
[1]Institute of Information Science
Academia Sinica
Taipei 115, Taiwan

Chiou-Shann Fuh[2]
[2]Department of CSIE
National Taiwan University
Taipei 106, Taiwan

Abstract

We propose a color-based tracking framework that infers alternately an object's configuration and good color features via particle filtering. The tracker adaptively selects discriminative color features that well distinguish foregrounds from backgrounds. The effectiveness of a feature is weighted by the Kullback-Leibler observation model, which measures dissimilarities between the color histograms of foregrounds and backgrounds. Experimental results show that the probabilistic tracker with adaptive feature selection is resilient to lighting changes and background distractions.

1 Introduction

Methods to track moving objects in a dynamic background mainly depend on two mechanisms: the appearance model and the search algorithm. The appearance model induces the similarity measurement of the target and candidates; the search algorithm finds the most possible state of the object according to the similarity measurement. Isard and Blake propose to track an object's contour by the CONDENSATION algorithm [6]. The contour is modeled by control points, which should be located at edges or corners during tracking. In their probabilistic formulation, the similarity is estimated by the likelihood function—a conditional distribution that yields the probability of observing the object in an image frame under the predicted contour configuration. Toyama and Blake [10] introduce an exemplar-based probabilistic tracker that does not require parametric models but compares shapes in a metric space.

Most region-based tracking algorithms use texture or color information as the appearance models. For example, Jepson et al. apply the wavelet-based texture features to represent objects [7]. As to the color information, it has been demonstrated that color histograms are effective observation models for real-time visual tracking [1], [2], [4], [8]. Generally, a color-based tracking algorithm relies on some similarity functions, e.g. the Bhattacharyy coefficient, to compare the color histograms of the target and the candidates, and incorporates a deterministic [4] or a stochastic [8] algorithm to search the state space efficiently. In this kind of tracking framework, the color histograms are usually computed under a fixed color space such as RGB or HSV during the whole tracking process. Therefore, the features used to construct the appearance model are fixed regardless of the changes in circumstances for tracking.

While previous color-based tracking algorithms are quite efficient, the lack of adaptation in color models leads to performance deficiency in handling situations such as illumination changes or background distractions. Shi and Tomasi [9] have pointed out that good features are as important as good tracking algorithms. Similar arguments on the importance of features are also set out for object-detection problems, e.g. [11]. While testing a tracking algorithm, one can easily observe how severely the performance may degrade with inadequate features. Typically, bad features result in an objective function with so many local minima that even a versatile algorithm loses its track in the maze.

We address the problem of adaptive feature selection for real-time tracking. Our goal is to select adaptively a combination of appropriate color features while the tracking process proceeds. A related work is proposed by Collins and Liu [3]. They embed online feature selection into a mean-shift tracker that estimates the translation of an object using the most discriminative color features. The features are selected from a set of parameter settings that resemble different color spaces. Similar ideas on discriminating foreground and background were suggested earlier by Wu and Huang in [12], where transductive learning is used to classify color distributions in the form of Gaussian mixture.

Our contribution to adaptive feature tracking is that we integrate feature selection and visual tracking in a unified probabilistic framework consisting of particle filters. The states of the features and the object's configuration are inferred iteratively in a Bayesian formulation. The experimental results indicate that our method can adaptively select good features to prevent background distraction and to accommodate illumination changes and rapid motion.

Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04)
1051-4651/04 $ 20.00 IEEE

2 Particle Filtering for Tracking

The concept of applying particle filters to real-time object tracking can be best captured by investigating the CONDENSATION algorithm [6]. Owing to its simplicity, we shall use the formulation of CONDENSATION to describe hereafter the steps of particle filtering. Further details on the theoretical issues and applications of particle filters can be found in [5].

Essentially, the idea of filtering is to infer the marginal posterior distribution of the state $\mathcal{X}_t$, given previous observations $\mathcal{Z}_{0:t}$. From the Bayes' formula and the Markov prior, we can derive a recursive form of the posterior

$$p(\mathcal{X}_t|\mathcal{Z}_{0:t}) \propto p(\mathcal{Z}_t|\mathcal{X}_t) \int_{\mathcal{X}_{t-1}} p(\mathcal{X}_t|\mathcal{X}_{t-1}) p(\mathcal{X}_{t-1}|\mathcal{Z}_{0:t-1})\,. \tag{1}$$

The recursive form allows us to use the posterior at time step $t-1$ as the prior for time step t. A particle filter approximates the posterior $p(\mathcal{X}_{t-1}|\mathcal{Z}_{0:t-1})$ by a finite set $\{X_{t-1}^k\}_{k=1}^K$ of K particles; each particle is associated with a weight π_{t-1}^k to form $\{X_{t-1}^k, \pi_{t-1}^k\}_{k=1}^K$. To carry out the recursion of particle filtering, we still need two probabilities $p(\mathcal{Z}_t|\mathcal{X}_t)$ and $p(\mathcal{X}_t|\mathcal{X}_{t-1})$, which correspond to an observation model and a dynamic model respectively. We will explain how we choose these two probabilities in the following sections. For now we just outline the steps in one iteration of particle filtering:

1. Resample $\{X_{t-1}^k, \pi_{t-1}^k\}$ into $\{\tilde{X}_{t-1}^k, 1/K\}$;
2. Predict by sampling from $X_t^k \sim p(\mathcal{X}_t|\mathcal{X}_{t-1} = \tilde{X}_{t-1}^k)$ to generate $\{X_t^k, 1/K\}$;
3. Measure and weight by $\pi_t^k \propto p(\mathcal{Z}_t|\mathcal{X}_t = X_t^k)$ to generate $\{X_t^k, \pi_t^k\}$, normalized so that $\sum_{k=1}^K \pi_t^k = 1$.

Note that we will create two particle sets $\{X_t^k, \pi_t^k\}_{k=1}^K$ and $\{F_t^m, \omega_t^m\}_{m=1}^M$ to estimate the states of object configurations and color features based on the recursion.

3 Adaptive Feature Selection

In color-based tracking various features can be obtained by manipulating the parameters of color spaces. We adopt the assumption proposed in [3] to choose better color features for tracking. Briefly, the most significant feature is the one that best distinguishes the foreground object from the background scene. Based on different features, we compute the color histogram of foreground (background) region in different color spaces. The effectiveness of a feature is thus relevant to the *dissimilarity* between each pair of foreground and background color histograms. We propose to measure the dissimilarity of two histograms by the Kullback-Leibler distance (the relative entropy).

To begin with, we have to define the foreground and background regions for a target. We apply the center-surround concept proposed in [3]. However, our approach includes the spatial information to make the representation model more appropriate, by applying different spatial weighting schemes. The foreground is the region inside a bounding box that just encompasses the object; the background is the region, excluding the foreground, inside a larger bounding box. The widths (heights) of foreground and background bounding boxes are kept at the ratio of $1:2.2$ for all the experiments presented in this paper. For the foreground region, a 2D Gaussian is incorporated to give the pixels closer to the center higher weights. As for the background region, we use a reverse 2D Gaussian to give the pixels that are farther from the center higher weights.

3.1 Feature selection via particle filtering

Assume now we know the state $\mathcal{X}_t = (x_t, y_t, w_t, h_t)$ of the object's configuration. Let the state of a feature be denoted as $\mathcal{F}_t = (\alpha_t, \beta_t, \gamma_t)$. Then, given a pixel $I_t(\mathbf{u})$, we can compute the inner product between its RGB values and $\mathcal{F}_t$, i.e., $\langle I_t(\mathbf{u}), \mathcal{F}_t \rangle = \alpha_t R + \beta_t G + \gamma_t B$. By uniformly dividing the possible range of the inner products into $N = 128$ bins, we can calculate the weighted histograms q_t^1 and q_t^2 inside the foreground and background regions $\mathcal{R}_t^1(\mathcal{X}_t)$ and $\mathcal{R}_t^2(\mathcal{X}_t)$ by

$$q_t^i(n; \mathcal{X}_t, \mathcal{F}_t) = C_t^i \sum_{\mathbf{u} \in \mathcal{R}_t^i(\mathcal{X}_t)} g^i(|\mathbf{u} - \mathbf{x}_t|) \delta_{n, b(\mathbf{u}; \mathcal{F}_t)}\,,\ i = 1, 2\,, \tag{2}$$

where n is the bin index and $\mathbf{x}_t = (x_t, y_t)$ denotes the center of bounding box. Note that q_t^1 and q_t^2 are weighted spatially by the Gaussian $g^1(\cdot)$ and the reverse Gaussian $g^2(\cdot)$. The mapping $b(\cdot)$ returns the histogram bin index based on the quantization of $\langle I_t(\mathbf{u}), \mathcal{F}_t \rangle$. The Kronecker delta helps to add the Gaussian weights of pixels belonging to bin n. The normalization term C_t^i ensures $\sum_{n=1}^N q_t^i(n) = 1$.

We estimate the best feature by the recursion of particle filtering. At time step $t-1$ we have the particle set $\{F_{t-1}^m, \omega_{t-1}^m\}_{m=1}^M$ that represents $\mathcal{F}_{t-1}$. Each particle F_{t-1}^m consists of a 3-tuple $(\alpha_{t-1}^m, \beta_{t-1}^m, \gamma_{t-1}^m)$. Recall that we need to define the dynamic model $p(\mathcal{F}_t|\mathcal{F}_{t-1})$ for particle filtering. Here we use simple, unconstrained Brownian motion: $F_t^m = \tilde{F}_{t-1}^m + \mathbf{v}_t^m$, where $\mathbf{v}_t^m \sim \mathcal{N}(\mathbf{0}, \Sigma)$. Each new feature particle F_t^m determines a pair of histograms $q_t^1(n; \mathcal{X}_t, \mathcal{F}_t = F_t^m)$ and $q_t^2(n; \mathcal{X}_t, \mathcal{F}_t = F_t^m)$, and the observation model of feature state is then defined by

$$p(\mathcal{Z}_t|\mathcal{F}_t = F_t^m) \propto 1 - \exp\{-\lambda\, KL(q_t^1 \| q_t^2)\}\,, \tag{3}$$

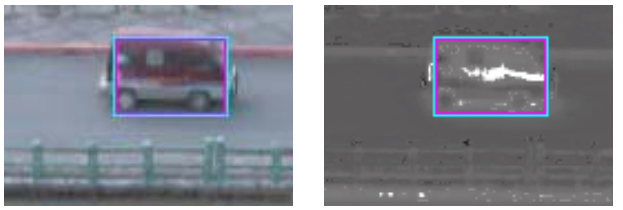
where $KL(q_t^1 \| q_t^2) = \sum_{n=1}^{N} q_t^1 \log(q_t^1/q_t^2)$ denotes the Kullback-Leibler distance between q_t^1 and q_t^2. Plugging in all we need for the recursion, we obtain an update particle set $\{(F_t^m, \omega_t^m)\}_{m=1}^M$ for the next time step.

3.2 The likelihood images

Each feature particle F_t^m is associated with a likelihood image of the same size as current image frame I_t. We denote the likelihood image as $L_t^m \equiv L_t^m(I_t; F_t^m)$. Each pixel $L_t^m(\mathbf{u})$ in the likelihood image is computed by first projecting $\langle I_t(\mathbf{u}), F_t^m \rangle$ to get the bin index n^*, and then deriving the pixel value from the log-ratio $\log(q_t^1(n^*; \mathcal{X}_t, F_t^m)/q_t^2(n^*; \mathcal{X}_t, F_t^m))$. Combining all feature particles, we get a compound likelihood image $\hat{L}_t$ as

$$\hat{L}_t(I_t; \mathcal{F}_t) = \sum_{m=1}^{M} \omega_t^m L_t^m(I_t; F_t^m)\,. \tag{4}$$

Fig. 1 illustrates a cropped image frame and the corresponding compound likelihood image.



(a) (b)

Figure 1. (a) A cropped image frame. (b) The compound likelihood image of (a).

4 Color-Based Probabilistic Tracking

The color-based probabilistic tracking is performed on compound likelihood images. Another particle filter $\{X_t^k, \pi_t^k\}_{k=1}^K$ is used to infer the current configuration state of object. The observation model of a configuration $X_t^k = (x_t^k, y_t^k, w_t^k, h_t^k)$ is straightforward: According to the predicted configuration, we add together the pixel values inside the foreground region $\mathcal{R}_t^1(X_t^k)$ in the compound likelihood image, then we map the sum by a logistic function to make the observation model constrained in $[0, 1]$. In short, the observation $p(\mathcal{Z}_t|\mathcal{X}_t)$ is approximated by

$$p(\mathcal{Z}_t|\mathcal{X}_t = X_t^k) \propto \left(1+\exp\left(-A \sum_{\mathbf{u}\in\mathcal{R}_t^1(X_t^k)} \hat{L}_t(\mathbf{u})\right)\right)^{-1}. \tag{5}$$

As to dynamic models, we use a first-order autoregressive process to model the state transition: $X_t^k = \tilde{X}_{t-1}^k + B\mathbf{w}_t^k$, where $\mathbf{w}_t^k$ is a vector drawn from a standard normal distribution and B is a scaling matrix. Algorithm 1 summarizes the iterative steps for feature selection and tracking.

Algorithm 1: Color Feature Tracking

Input : Image frames $\{I_t\}_{t=0}^T$, object's initial configuration, the number of tracking particles K, and the number of feature particles M.

Initialization: Generate the tracking particles $\{X_0^k, \pi_0^k\}_{k=1}^K$ and the feature particles $\{F_1^m, \omega_1^m\}_{m=1}^M$; Set $t \leftarrow 1$.

while $t \leq T$ **do**

1. Acquire a new image frame I_t;
2. Construct the compound likelihood $\hat{L}_t = \sum_{m=1}^M \omega_t^m L_t^m(I_t; F_t^m)$;
3. Apply particle filtering on $\hat{L}_t$ to get $\{X_t^k, \pi_t^k\}_{k=1}^K$;
4. Estimate $\hat{X}_t$ from $\{X_t^k, \pi_t^k\}_{k=1}^K$ for displaying, and get the new center-surround area;
5. Apply particle filtering for feature selection and get $\{F_{t+1}^m, \omega_{t+1}^m\}_{m=1}^M$;
6. $t \leftarrow t + 1$;

5 Experimental Results

The experiments of color-based tracking and feature selection are carried out using Algorithm 1. We use $K = 50$ particles for tracking and $M = 20$ particles for feature selection. The object's configuration is given for the initial image frame. At the initial stage of Algorithm 1, a feature particle set $\{F_1^m, \omega_1^m\}_{m=1}^M$ must be constructed by trying all the combinations of 3-tuple $\{(\alpha_1, \beta_1, \gamma_1) | \alpha_1, \beta_1, \gamma_1 \in \{-2, -1, 0, 1, 2\}\}$, to find the top 20 features. The values of weights $\{\omega_1^m\}_{m=1}^M$ are estimated by the Kullback-Leibler observation model defined in (3).

In the first experimental result we compare the performances of a normal tracker and the tracker with feature selection. The normal tracker selects the best feature in the initial frame, and then keeps on using this feature to track the moving object subsequently. When the object moves towards a background region of wrongly estimated, high likelihood values, the tracker will be distracted, like in Fig. 2. On the other hand, the tracker with adaptive feature selection can find good discriminative features to prevent distraction, see Fig. 3. The second experimental result highlights the tracker's capability of handling motion blur and illumination changes. Some sample frames are shown in Fig. 4.

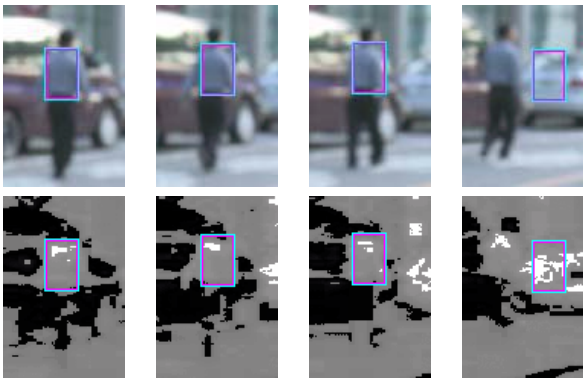


Figure 2. Top: The tracking results of a normal tracker with a fixed feature. Bottom: The corresponding likelihood images.

Figure 3. Top: The results of tracking with adaptive feature selection. Bottom: The compound likelihood images.

6 Conclusion

The feature selection scheme of our approach maintains a flexible set of weighted color features rather than using a fixed one. We have presented that particle filtering, when integrated with the Kullback-Leibler observation model, affords us a probabilistic manner to choose good features. Furthermore, it provides a good way to construct a compound likelihood image according to the feature weights. Tracking an object in a compound likelihood image is more robust than in the original image, since possible locations of the target are emphasized and background distractions are suppressed, by the discriminative features. Although we address the problem of selecting color features, the formulation can be extended to selecting other types of features. For example, including gradients or textures into the feature set may be useful when no significant color features are found.

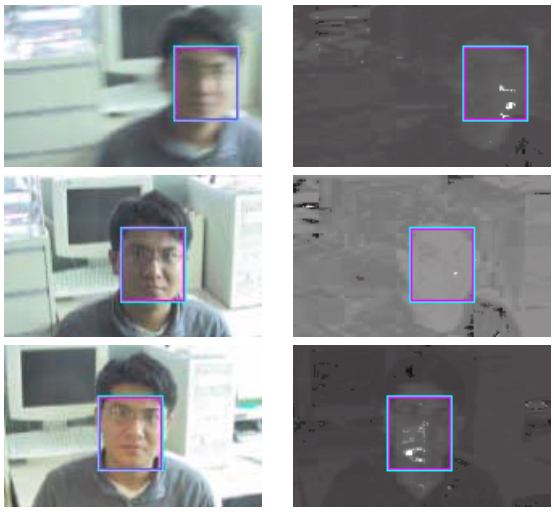


Figure 4. Real-time tracking under motion blur and illumination changes.

References

[1] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *Proc. Conf. CVPR*, pages 232–237, Santa Barbara, CA, 1998.

[2] H.-T. Chen and T.-L. Liu. Trust-region methods for real-time tracking. In *Proc. Eighth ICCV*, volume 2, pages 717–722, Vancouver, Canada, 2001.

[3] R. Collins and Y. Liu. On-line selection of discriminative tracking features. In *Proc. Ninth ICCV*, pages 346–352, Nice, France, 2003.

[4] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. Conf. CVPR*, volume 2, pages 142–149, Hilton Head Island, SC, 2000.

[5] A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, Berlin, 2001.

[6] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *Proc. Fourth ECCV*, volume 1, pages 343–356, Cambridge, England, 1996.

[7] A. Jepson, D. Fleet, and T. El-Maraghi. Robust online appearance models for visual tracking. *IEEE Trans. PAMI*, 25(10):1296–1311, October 2003.

[8] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *Proc. Seventh ECCV*, volume 1, pages 661–675, Copenhagen, Denmark, 2002.

[9] J. Shi and C. Tomasi. Good features to track. In *Proc. Conf. CVPR*, pages 593–600, Seattle, WA, 1994.

[10] K. Toyama and A. Blake. Probabilistic tracking in a metric space. In *Proc. Eighth ICCV*, volume 2, pages 50–57, Vancouver, Canada, 2001.

[11] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. Conf. CVPR*, volume 1, pages 511–518, Kauai, HI, 2001.

[12] Y. Wu and T. Huang. Color tracking by transductive learning. In *Proc. Conf. CVPR*, volume 1, pages 133–138, Hilton Head Island, SC, 2000.