Enhanced Appearance Models for Object Tracking

An Zhao, Michael J. Brooks and Anthony R. Dick
School of Computer Science
The University of Adelaide
Adelaide, SA 5005, Australia
an_zhao@hotmail.com {mjb,ard}@cs.adelaide.edu.au

Abstract

This paper is concerned with improving target appearance models to realize robust object tracking. We explore the use of feature space other than the commonly used color space for object tracking. Specifically, we employ gradient information to be used separately as well as in conjunction with color information. Our target appearance model is then represented in the form of a histogram using its gradient and color feature spaces, and frame-to-frame tracking is performed using mean shift or local exhaustive search. By combining gradients with color, we build new appearance models with combined feature spaces. Based on our extensive testing of these models, we find that they can be used to track complex objects, such as full 360-degree rotating objects, appearance-changing objects, occluding objects and zooming objects.

1. Introduction

Object tracking is a key topic of computer vision which has attracted considerable attention over the years due to its potential practical applications in areas such as surveillance, video editing and processing, human-machine interface and so on. Robust and reliable object tracking remains an unsolved problem, especially for those objects whose appearance changes significantly during the tracking process. So, finding a way to represent the target model as robustly as possible is of great importance. Generally, the target's appearance model is based on its color feature space [12, 6, 5, 8, 11]. However, we note that color-based appearance models alone are not robust enough, especially for tracking an object whose color is similar to the background. Because of this, we explore gradient feature spaces. Additionally, combined feature space (color&gradient) appearance models are constructed. We therefore focus on improving tracking results using enhanced appearance models.

The structure of the paper is as follows. In section 2, we review some related object tracking work. In section 3, we present our method for computing the gradients. A description for building gradient-based and gradient&color-based histogram models is given. In section 4, we present some testing results of our trackers based on single gradient, single color and gradient&color models. Once the target is well represented, we seek our possible candidates by a local exhaustive search of the whole image at the next frame to find maximal matching with the reference target model. The Bhattacharyya Coefficient [1, 7, 10] is employed to measure the similarity of the histogram models of the target and candidate. As a local exhaustive search of the image may be time consuming, the mean-shift algorithm [6, 5, 4] is adopted to speed up the algorithm. Section 5 gives concluding remarks.

2. Related Work

Color-based trackers [11, 8, 5, 6] have proved to be fairly robust and efficient, and they can sometimes be used to track objects whose spatial structures change markedly. However, in some situations, color-based trackers break down quickly. This is mainly because a color histogram only contains the color information distribution of our target, regardless of its shape and structure. Recently, the exploration of improved feature spaces for tracking has received greater attention. Birchfield [2] explored intensity gradients for head tracking. However, this work only focuses on the gradients of the target's boundary part, whereas we consider and use the gradient information of each pixel of our target. In [3], Black and Jepson proposed an EigenTracking approach (a parameterized matching method between the eigenspace and the reference target) to track complex objects. In [9], Isard and Blake adopt particle filters for tracking objects in dense visual clutter. Instead of tracking the full object, robust tracking is achieved through tracking outlines.

One of the most efficient and famous tracking algorithms is mean shift [6, 5, 4]. Mean shift is a

0-7695-2929-1/07 $25.00 © 2007 IEEE
DOI 10.1109/ICIG.2007.64

IEEE computer society

nonparametric estimator of density gradient which is derived from the process of Bhattacharyya Coefficient (see equation 1) maximization as follows:

---

(1) According to the definition of the Bhattacharyya Coefficient, we have:

$$\rho[p(y), q] = \sum_{u=1}^{m} \sqrt{p_u(y) q_u} (1)$$

where q is the target model, $p(y)$ is the candidate model at location y, u is the bin number of the histogram in the range of $[1 - m]$ and ρ is the Bhattacharyya Coefficient which is used to measure the similarity of target and candidate models.

(2) Using the Taylor expansion around the value $p_u(y_0)$, equation (1) can be rewritten approximately as follows:

$$\rho[p(y), q] \approx \frac{1}{2} \sum_{u=1}^{m} \sqrt{p_u(y_0) q_u} + \frac{1}{2} \sum_{u=1}^{m} p_u(y) \sqrt{\frac{q_u}{p_u(y_0)}} (2)$$

(3) To maximize (2), as the first term being independent of y, we just need to maximize the second term: $(\frac{1}{2} \sum_{u=1}^{m} p_u(y) \sqrt{\frac{q_u}{p_u(y_0)}})$.

(4) The equation (2) can be further rewritten as follows:

$$\rho[p(y), q] \approx \frac{1}{2} \sum_{u=1}^{m} \sqrt{p_u(y_0) q_u} + \frac{C_h}{2} \sum_{i=1}^{n_h} w_i k(||\frac{y - x_i}{h}||^2) (3)$$

where $w_i = \sum_{u=1}^{m} \sqrt{\frac{q_u}{p_u(y_0)}} \delta[b(x_i) - u]$, δ is a delta function, b is bin index function, x_i is the pixel's location, C_h is a normalization constant, y is the central location of tracking window, k is the kernel function and h is the bandwidth of tracking window.

(5) Similar to step (3), in order to maximize the second term of equation (3), the term $(\frac{C_h}{2} \sum_{i=1}^{n_h} w_i k(||\frac{y - x_i}{h}||^2))$ has to be maximized.

(6) $(\frac{C_h}{2} \sum_{i=1}^{n_h} w_i k(||\frac{y - x_i}{h}||^2))$ represents the density estimate computing with kernel profile $k(x)$ at y in the current frame. So, we can find the maximal value for this equation by finding the highest density location.

As the mean shift algorithm searches for the target in a neighbouring area in the next frame, it is not suitable for tracking objects having large displacement in two consecutive frames. If this is the case, a local exhaustive search is used instead of mean shift.

3. Gradients

Generally, color is the most widely used feature space [8, 11] in object tracking. However, due to the fast changing appearance or status of our target, sometimes a color histogram is not robust enough. Because of this, we explore the gradient feature space. Given a pixel, we compute absolute values of the horizontal and vertical gradients. More detailed gradient information can be gathered by also computing the absolute (left and right) diagonal gradients.

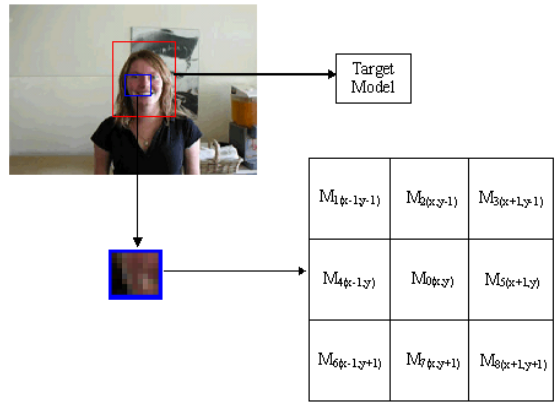


Figure 1. Illustrative diagram for calculating gradients

In Figure 1, the large rectangle (red rectangle) contains our target. Regard pixel M_0 as a randomly selected pixel from the target rectangle. A small 3×3 window (blue square in Fig. 1) is drawn using M_0 as its centre. In the small window, as the pixel M_0 (its coordinate (x, y)) is the centre, there are eight pixels $M_{1(x-1,y-1)}$, $M_{2(x,y-1)}$, $M_{3(x+1,y-1)}$, $M_{4(x-1,y)}$, $M_{5(x+1,y)}$, $M_{6(x-1,y+1)}$, $M_{7(x,y+1)}$ and $M_{8(x+1,y+1)}$ around it in the order of left to right and top to bottom (see Figure 1). Generally, we use pixel M_0's RGB value ($\Phi_r(M_0)$,$\Phi_g(M_0)$,$\Phi_b(M_0)$) to represent it[1]. In some situations, such as when the target's color is similar to its background, color is not the preferred feature space. Because of this, some other suitable feature spaces are required, and so we consider gradients.

For each pixel of the target, such as M_0, we compute its gradients as follows:

1. *Compute the absolute horizontal red gradient,G_{hr}, at*

[1] In the whole paper, Φ is a function used to access a pixel's RGB value. So, Φ_r is used to get the value of a pixel's associated red component. Φ_g and Φ_b have the similar meanings as Φ_r.

M_0 given by:

$$G_{hr}(M_0) = |\Phi_r(M_5) - \Phi_r(M_4)|; (4)$$

2. *As above, compute the absolute vertical red gradient (G_{vr}) at M_0 given by*

$$G_{vr}(M_0) = |\Phi_r(M_2) - \Phi_r(M_7)|; (5)$$

3. *For the diagonal red gradient of M_0 (Left diagonal red gradient (G_{Ldr}) : $M_3 \rightarrow M_6$, and Right diagonal red gradient (G_{Rdr}): $M_1 \rightarrow M_8$), they are achieved by:*

$$G_{Ldr}(M_0) = |\Phi_r(M_6) - \Phi_r(M_3)|; (6)$$

$$G_{Rdr}(M_0) = |\Phi_r(M_8) - \Phi_r(M_1)|; (7)$$

4. *The complete red gradients (G_{cr}) are obtained by combining horizontal, vertical, left diagonal and right diagonal red gradients, that is, $G_{cr}(M_0) = G_{hr}(M_0) + G_{vr}(M_0) + G_{Ldr}(M_0) + G_{Rdr}(M_0)$.(8)*

5. *As the above equations only compute the red gradients, we are also required to compute the green and blue gradients similarly.*

The pixel M_0's gradient value is then the triple $(G_{cr}(M_0), G_{cg}(M_0), \mathrm{G}_{cb}(M_0))$.

In the above equations, r, g, b, h, v, L, R, d, G stand for red, green, blue, horizontal, vertical, left, right, diagonal and gradient respectively.

3.1. Gradient-based target representation

Once the way for computing a feature space is determined, we represent the target in the form of a histogram. First, we choose a target, for instance: use a bounding rectangle. For different parts of the target in the rectangle, the weights are different. For those parts close to the centre of the rectangle, we put more weight, whereas those parts further from the centre, less weight is given. Then a suitable feature space is chosen to represent that target. For instance, we represent the target in the form of a gradient histogram. For each pixel of the target, we find its bin index in the histogram based on its gradient value. The normalized weight of that pixel is then added to the pixel's corresponding bin in the histogram (see Figure 2).

The procedure for building the target's gradient-based histogram model:

1. *Choose the target model at the initial frame.*
2. *For each pixel of the target, find its associated bin in the histogram.*
   (a) *For each pixel, we compute its gradient value (a, b, c) as shown above.*
   (b) *If the size of each bin is t, then each pixel's quantized value is $(\frac{a}{t}, \frac{b}{t}, \frac{c}{t})$.*
   (c) *As $(a$ or b or $c)$ is in the range of $[0 - 1020]$, the size of red gradients, green gradients or blue gradients component is approximately equal to $\frac{1024}{t}$.*
   (d) *As a result, each pixel's bin index in a $2D$ histogram can be computed through: $bin_{index} = \frac{a}{t} + \frac{1024b}{t^2} + \frac{1024^2c}{t^3}$.*
3. *Once we find each pixel's bin index, we need to find the associated weight value (Epanechnikov kernel) to be added to the corresponding bin.*
   (a) *Assume the target contains n pixels with the weights $w_1, w_2, w_3...w_n$ respectively.*
   (b) *For each pixel p_i, its weight is related to the distance between it and the centre o of the target (d_{io}).*
      i. *Suppose the distance between the boundary point b and centre o (d_{bo}) is equal to l.*
      ii. *For each pixel p_i, calculate the distance between it and centre o that is d_{io}.*
         - *if $d_{io} \leq l$, then $w_i = \frac{l^2 - d_{io}^2}{l^2} = 1 - (\frac{d_{io}}{l})^2$*
         - *if $d_{io} > l$, then $w_i = 0$.*
   (c) *Compute the sum of $w_1, w_2, w_3...w_n$ and assign the value to w_{sum}.*
   (d) *Normalize each pixel's weight by dividing its original weight w_i by w_{sum}.*
4. *Construct the target's gradient-based histogram model by adding each pixel's normalized weight (step 3) to its associated bin (step 2).*

3.2. Gradient&Color-based target representation

Previously, we only used a gradient histogram to represent our target's appearance model. Now, we add color feature space into our existing gradient histogram.

The procedure for building the target's gradient&color-based histogram model:

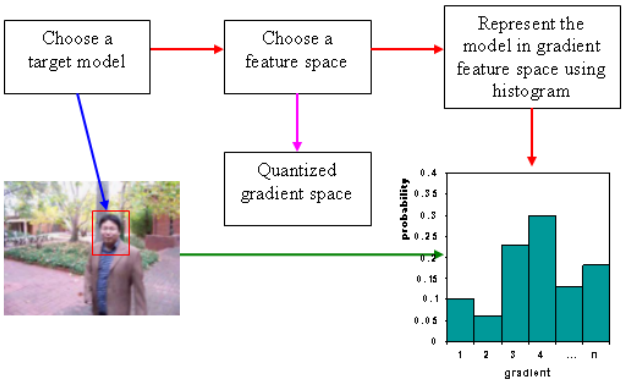


Figure 2. Flow of target's gradient-based histogram representation

1. *Choose the target model at the initial frame.*
2. *Build the gradient histogram H_A and color histogram H_B for the target.*
   (a) *Build the gradient histogram H_A using the steps shown in "The procedure for building the target's gradient-based histogram model".*
   (b) *Build the color histogram H_B following the procedure for building the target's gradient-based model. As the RGB range is in $[0-255]$, 256 is used instead of 1024 in all the equations.*
3. *Our final gradient&color-based histogram is achieved by combining the histogram H_A and H_B together as follows (adding H_B after H_A):*
   - $H_f = \alpha H_A + (1-\alpha)H_B$. *A final histogram is achieved by adding the weighted histogram H_A and H_B linearly.*
   - *For the above equation, the parameter α is set manually according to actual conditions of our target within the range of $[0-1]$. For instance, if we intend to track a black target in a dark environment continuously, α is adjusted to be a value which is larger than 0.5, or even close to 1. For the final histogram H_f, its size is dependent on the size of H_A, H_B and the value of α. In order to verify the contribution of the gradient and color histograms equally, it is recommended that histograms H_A and H_B should have the same number of bins.*

4. Experimental Results

In this section, we examine the performance of trackers based on an individual gradient model, a color model, as well as the gradient&color model. In all of these testing cases, a target is chosen at the initial frame and is not updated at any stage.

4.1. Gradient-based trackers

We built a tracker having the above gradients as the only feature space and searched globally in each frame for the best matching target. A few outputs of a tracker of this type are presented (see Figure 3).

The Bhattacharyya Coefficient surfaces for outputs shown in Figure 3 are presented in Figure 4. In this figure, for any point of the surface, such as point A with the coordinate value of (x, y, z), (x, y) represents the point's location in the horizontal plane as well as the centre of the tracking window. Similarly, z stands for the value of the Bhattacharyya Coefficient which is always in the range of [0-1]. As there are clear peaks (maximal Bhattacharyya Coefficient value) with the right location in each diagram of the Bhattacharyya Coefficient surface, it demonstrates that our gradients are a suitable feature space for object tracking.

In Figure 5, the red rectangle indicates outputs of the gradient-based tracker, whereas the blue rectangle refers to outputs of the color-based tracker. After we examine the outputs, we find that the gradient-based tracker successfully captures the full 360-degree out-of-plane rotating and zooming object, whereas the color-based tracker does not. If there is another nearby object with similar color, it is easy for the color-based tracker to capture the wrong target.

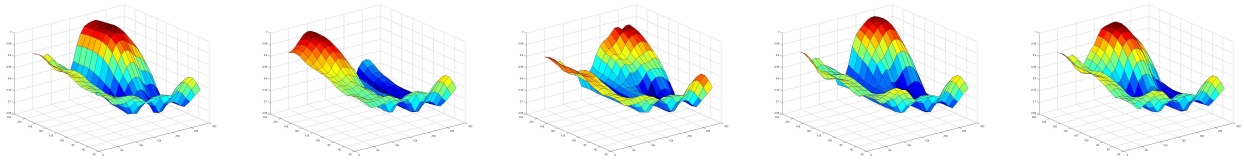
4.2. Color&Gradient-based trackers

In the above section, we chose the gradients as the only feature space used in the tracker. However, for some objects whose appearances change quickly or become occluded at some stage; a single feature space based model, such as the color model or gradient model usually does not work well. In order to rectify this, a new and robust histogram is built by combining the gradients with color into a histogram. A new tracker based on histogram of this type was built. This tracker is quite robust, as it can capture the target whose appearance changes significantly and seriously occludes in some frames (see Figure 6).

Based on our extensive testing of this tracker, we find that trackers based on combined feature spaces are robust. That is mainly because being different feature spaces, colors and gradients complement each other. In some situations, such as dark environments (see Figure 5), color can

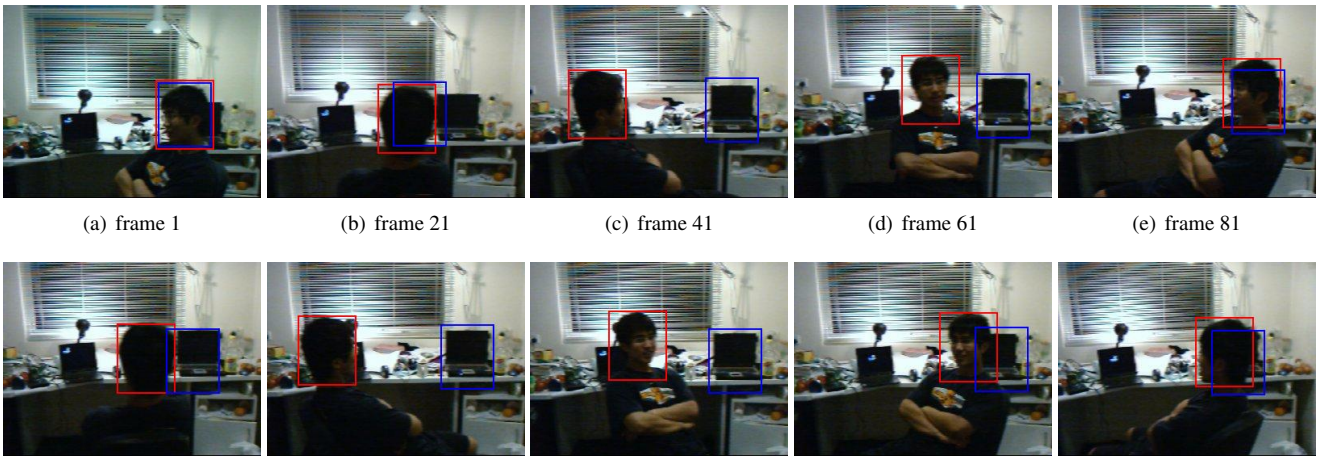
(a) frame 41 (b) frame 81 (c) frame 121 (d) frame 161 (e) frame 201

Figure 3. Tracking an appearance-changing object with a gradient tracker having a global search of the whole image for the best candidate, when compared to the reference target chosen at the initial frame. The outputs of frames 41, 81, 121, 161, and 201 are shown.



(a) frame 41 (b) frame 81 (c) frame 121 (d) frame 161 (e) frame 201

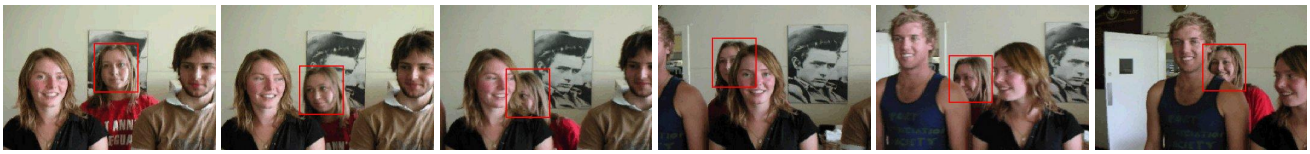
Figure 4. The Bhattacharyya Coefficient surfaces for the outputs shown in Figure 3.



(a) frame 1 (b) frame 21 (c) frame 41 (d) frame 61 (e) frame 81

(f) frame 101 (g) frame 121 (h) frame 141 (i) frame 161 (j) frame 181

Figure 5. Tracking a rotating and zooming object whose color is similar to its nearby environment.



(a) frame 1 (b) frame 15 (c) frame 22 (d) frame 45 (e) frame 68 (f) frame 100

Figure 6. Tracking an occlusive object with a tracker having both color and gradients as the feature spaces. Although the target becomes occluded at some stages, this tracker still captures the target consistently.

not be used to represent the object's characteristics accurately. However, gradients may complement the color feature space and characterize that object. On the other hand, in some situations, gradients may not work, whereas color does work. For some testing cases such as the one shown in Figure 6, single color or gradient does not work. However, if color and gradients are combined together, trackers based on the $color\&gradient$ model do capture the target firmly (see Figure 6). Because, we build a more precise and distinctive model with combined feature spaces.

5. Conclusion

In this paper, we explored feature spaces other than color, such as gradients. By combining the gradients with color, we build trackers with combined feature spaces. Compared with outputs of trackers having the color or gradients as the only feature space, the tracking results indicate that combined feature space is more promising. In some complex environments, combined feature spaces help us to represent and track the target accurately, whereas single color or single gradient does not. Based on our large number of experiments, we found that our enhanced appearance models based on combined feature spaces greatly improve the tracking results.

References

[1] F. Aherne, N. Thacker, and P. Rockett. The Bhattacharyya metric as an absolute similarity measure for frequency coded data. *Kybernetika*, 34(4):363–368, 1998.

[2] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. *International Conference on Computer Vision and Pattern Recognition*, pages 232–237, June 1998.

[3] M. J. Black and A. D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *4th European Conference on Computer Vision*, 1064:329–342, April 1996.

[4] Y. Cheng. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, August 1995.

[5] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. *in Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2:142–149, June 2000.

[6] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, May 2003.

[7] A. Djouadi, O. Snorrason, and F. Garber. The quality of training sample estimates of the bhattacharyya coefficient. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12:92–97, 1990.

[8] P. W. Fieguth and D. Terzopoulos. Color-based tracking of heads and other mobile objects at video frame rates. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 21–27, 1997.

[9] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. *4th European Conference on Computer Vision*, pages 343–356, April 1996.

[10] T. Kailath. The divergence and Bhattacharyya distance measures in signal selection. *IEEE Trans. Commun. Tech.*, 15:52–60, 1967.

[11] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. *7th European Conference on Computer Vision*, 2350:661–675, May 2002.

[12] M. Swain and D. Ballard. Color indexing. *Int, J. of Computer Vision*, 7(1):11–32, 1991.