An Improved CAMSHIFT Tracking Algorithm Applying on Surveillance Videos

Shwu-Huey Yen[a], Jui-Chen Chien[b] and Hsiao-Tsung Lin[c]

Department of Computer Science and Information Engineering, Tamkang University
New Taipei City, Taiwan, R.O.C

[a]105390@mail.tku.edu.tw, [b]699410212@s99.tku.edu.tw, [c]601420283@s01.tku.edu.tw

Keywords: tracking, mean shift, camshift, flooding, surveillance

Abstract. In this paper, we present an improved version of CAMSHIFT algorithm applying on surveillance videos. A 2D, hue and brightness, histogram is used to describe the color feature of the target. In this way, videos with poor quality or achromatic points can be characterized better. The flooding process and contribution evaluation are executed to obtain a precise target histogram which reflects true color information and enhances discrimination ability. The proposed method is compared with existing methods and shows steady and satisfactory results.

Introduction

Visual tracking is an important task in the computer vision applications such as video surveillance, navigation, traffic management, human-computer interaction etc. However, it is a very challenging problem due to abrupt object motion, changing appearance patterns of the object and/or the scene, occlusions, and camera motion.

There are many tracking methods emerged [1- 3]. One of the famous approaches is the mean shift algorithm [4]. The mean shift algorithm is a non-parametric technique that climbs the gradient of a probability distribution to find the nearest dominant mode. The algorithm has achieved considerable success in object tracking because of its simplicity and robustness.The algorithm is extended to video sequences called CAMSHIFT (Continuously Adaptive Mean Shift) [5]. CAMSHIFT uses the newly found mode as the initial mode for the next frame, i.e., center of the search window, and applies Mean shift algorithm again on the next frame. It also adaptively changes the search window sizes according to the previous tracking results. CAMSHIFT is first proposed for head and face tracking [5], therefore, it may fail when the target has multiple colors and/or a similar color to background's. Comaniciu et al. [6-7] proposed a kernel-based improvement which uses a weighted histogram computed from a circular region to represent the target. Authors also used the Bhattacharyya coefficient as a similarity measure. The kernel-based tracker can track general objects and have a good result. However, it evaluates Bhattacharyya coefficient on every iteration, which makes it computation expensive.

When the target is tracked continuously, the target size often changes over time. There are several methods proposed to solve this problem. In [5], it uses moments to estimate the size, but the method is originated for face tracking with a fixed ratio of 1: 1.2. In [9], it runs the tracking procedure three times with the size of the previous one, and increased/decreased 10% of that size, respectively. Finally, the size is decided by the one with the best Bhattacharyya coefficient. Unfortunately, small size usually turns out to be the best result and this causes the window smaller than it should be.

The Improvements

The proposed method is based on CAMSHIFT tracking algorithm with the improvements: (1) a 2D HY-histogram is used for features; (2) a flooding procedure is applied when the initial ROI (region of interest) is labeled; (3) an adaptive window size method is proposed. The details are described below.

2D HY-Histogram. The basic part in mean shift related tracking algorithms is to build a target histogram consisting of features of the ROI. Features may be simple (e.g., color only) or complicated (e.g., color, illumination, shape, texture and motion). Too many features result less tolerance in identifying the target and a more expensive computation. Since color is a simple feature and robust to rotation, scaling, several research, such as tracking faces [5], the specified region [10], deformable objects [11], adopt hue as the tracking feature. But in a visual surveillance system, especially when a camera is mounted in a high position, videos usually are degraded and contain many achromatic points due to low saturation, or too bright/dark illumination. Thus, besides the hue, the brightness Y is included as the target feature in our method. Similar to hue, it is robust to rotation and scaling, and it can describe achromatic points. The brightness value Y is evaluated as in Eq. 1 where R, G, B are the red, green, blue components.

$$Y = B \cdot 0.114 + G \cdot 0.587 + R \cdot 0.299\,. \tag{1}$$

In the proposed algorithm, a 2D HY-histogram of size $m \times n$ is designated to describe the target. For a point P, we use notations P_{hue}, P_Y and P_S to denote the hue, brightness, and saturation values of P. And, the term "color" in here is to stand for both hue and brightness information. Index function b associates a pixel P such that $b(P) = (i, j)$ where (i, j) is the index of the HY-histogram feature space according to Eq. 2 with hue_bin $i = 1, \cdots, m$ and brightness_bin $j = 1, \cdots, n$.

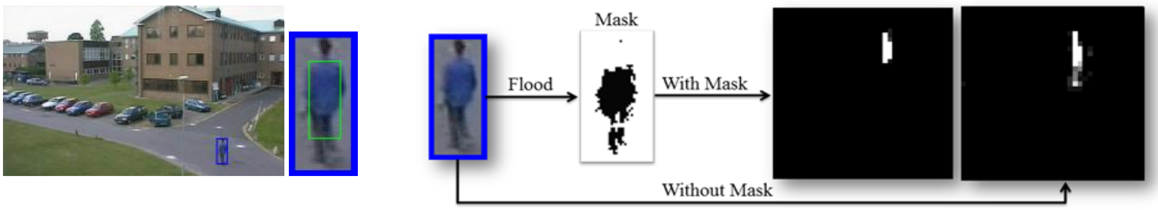
$$i = \begin{cases} m & \text{if } P \text{ is achromatic,} \\ \left\lfloor \dfrac{P_{hue}}{hue\_width} \right\rfloor + 1 & \text{otherwise,} \end{cases} \quad \text{and} \quad j = \left\lfloor \frac{P_Y}{Y\_width} \right\rfloor + 1\,. \tag{2}$$

P is achromatic if $P_S \le 0.1$ (low saturation), or $P_{max} \le 0.1$ (too dark), or $P_{min} \ge 0.9$ (too bright). P_{max} and P_{min} are maximum and minimum of normalized R, G and B values of P. In Eq. 2, the last hue_bin ($i = m$) is for achromatic points and $hue\_width = 360/(m\text{-}1)$ & $Y\_width = 256/n$. In our experiment, we use $m = 41$ and $n = 40$. To simplify notations, q_u and p_u are for the target and the candidate 2D HY-histograms respectively.

Flooding Procedure. When an initial rectangular ROI is labeled, the target feature (i.e., q_u) is calculated and the proposed algorithm proceeds to track the target on the next frame. To remove the irrelevant background color information included in the ROI, we adopt a flooding preprocess. In Fig. 1(a), a user chooses a rectangular ROI (blue one). We assume user's labeling is reasonable, i.e., the target is inscribed in the ROI which may contain some background and/or miss the target partially, but it covers most of the target. Taking half size of the ROI, we get a ROI$__{small}$ (green one) and the corresponding histogram $hist\__{small}$. Observing Fig. 1(a), the ROI contains quite a few background points but ROI$__{small}$ contains mostly true foreground points. Consequently, $hist\__{small}$ can be used to identify true foreground points of an ROI. A point P (with $b(P) = u$) in the ROI is classified to be foreground if $hist\__{small}(u) > Th(P)$. The threshold $Th(P)$ is given in Eq. 3 and it is a function of the distance d, the Euclidean distance between P to the center of ROI.

$$Th\,(P) = \min\,(d/M * r,\ 0.06), \tag{3}$$

where M is the maximum distance so that d/M is at most one, and r is a constant set to be 0.08. The rationale of Eq. 3 is when P is near the center then it is more likely to be a foreground point. Thus, $Th(P)$ is proportion to the distance d. A mask is formed by those classified foreground points as shown in Fig. 1(b). Comparing two histograms in Fig. 1 (b), there are many irrelevant color information kept in the q_u if no flooding preprocess. Thus, the initial target histogram q_u , for later tracking used, is built with flooding preprocess.



(a) (b)

Figure 1. Flood operation: (a) an ROI (blue rectangle) and the ROI_$_{small}$ (green rectangle); (b) the corresponding target histograms q_u with and without Flood operation.

(a) horizontal projection (σ_h) (b) The back projection of q_u (c) vertical projection (σ_v)

Figure 2. The back projection of q_u and its horizontal projection and vertical projection

Adaptive Window Size. When tracking, the search window (whose center is the mode x_c) of the previous frame is passed to the current frame and proceeds to track and localize the target again. Once the tracking is completed, the new mode y_1 is located and the size of the search window needs to be updated. By observing the histogram back projection in Fig. 2, the width and height of the updated window should be proportional to the standard deviations of vertical projection (σ_v) and horizontal projection (σ_h). With y_1 being the new mode center, we update the ROI width and height to be $\lambda\sigma_v$ and $\lambda\sigma_h$ for next frame. In CAMSHIFT [5], the target area is approximated by $M_{00}/256$, thus we have the following relation:

$$\text{Area of the ROI} \approx M_{00}/256 \approx \lambda\sigma_v * \lambda\sigma_h\,, \tag{4}$$

where $M_{00} = \sum_x \sum_y I(x,y)$, the zero$^{\text{th}}$ order moment. From Eq. 4, $\lambda \approx \sqrt{M_{00}/\sigma_v\sigma_h}$. In the experiment, we update the width and the height to be

$$\text{width\_}_{next\_frame} = \tfrac{1}{4}\sigma_v\sqrt{M_{00}/\sigma_v\sigma_h} \quad \text{and} \quad \text{height\_}_{next\_frame} = \tfrac{1}{4}\sigma_h\sqrt{M_{00}/\sigma_v\sigma_h}\,. \tag{5}$$

The Proposed Method

The proposed algorithm is depicted in Fig. 3. It consists of 4 main stages: initialization, contribution evaluation, object tracking and updating. In this study, a user chooses a rectangular ROI and we assume the labeling is reasonable as mentioned.

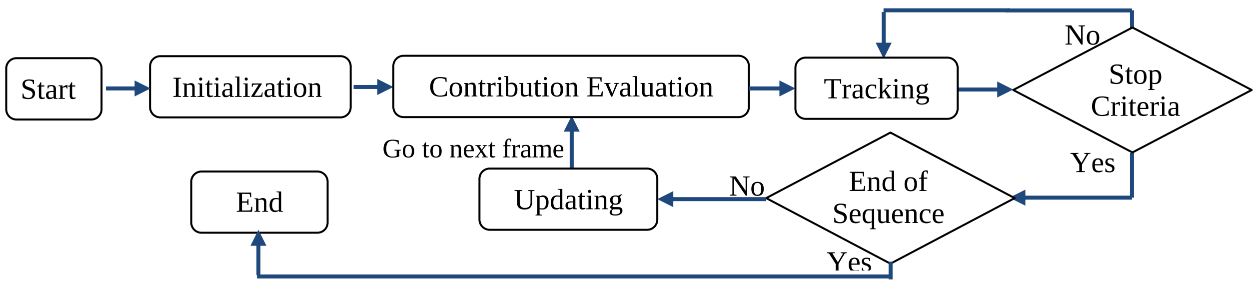


Figure 3. Flow chart of the proposed method

Initialization. It includes ROI labeling and target histogram q_u generating. The aforementioned flooding preprocess is used and let $\{x_i\}_{,\ i=1,\ \dots,\ K,}$ be the set of pixels within the mask. Similar to the kernel-based tracker [6,7], the target distribution density function q_u is denoted by Eq. 6:

$$q_u = C\sum_{\forall x_i \in \text{Mask}} k(\|x_i\|^2)\delta[b(x_i)-u], \tag{6}$$

where $k(l)$ is Epanechnikov kernel, $k(l)$=(1-l) if l < 1 and 0 otherwise. $\delta[t]$=1 if t = 0 and 0 otherwise, and C is a normalized factor to make total sum of q_u to be one.

Contribution Evaluation. The relation between background and foreground will be changed over time for moving objects. Therefore, the discriminative ability on each color feature u should be evaluated as in [3]. When an ROI is given, by extending outwards half of the width and half of the height, the reference background is defined as the area between these two rectangles. The color histogram in the reference background, $hist_b$, would be used in evaluating the contribution of colors. The formula is shown below:

$$C_u = \begin{cases} \log\left(\dfrac{\max(hist_o(u),\delta)}{\max(hist_b(u),\delta)}\right) & hist_o(u) > hist_b(u), \\ 0 & otherwise. \end{cases} \tag{7}$$

where C_u is the contribution for each color u, and δ is a small value to avoid overflow. Contribution zero means such color feature is more likely to appear in the background aand it has little contribution in identifying the target.

Object Tracking. Similar to the **Initialization**, the color of the target distribution density function q_u is obtained by Eq. 6 except now the computation is referring to every point x_i within the ROI.

$$q_u = C\sum_{\forall x_i \in ROI} k(\|x_i\|^2)\delta[b(x_i)-u], \tag{8}$$

Updating. After tracking of the current frame is completed, the new mode y_1 is located. Both the target histogram q_u and the window size will be updated. The target histogram centered at y_1, $p_u(y_1)$, of the current frame is evaluated by Eq. 8. Update q_u by

$$q_u = \alpha \cdot q_u(x_c) + (1-\alpha)\cdot p_u(y_1), \tag{9}$$

where $q_u(x_c)$ is the histogram from previous frame. In our experiment, since the initial q_u contains true target color information so α is set to be 0.98. Window size is updated by Eq. 5.

The complete algorithm is given below.

(a) Label an ROI with x_c as the center. Using flooding preprocess and Epanechnikov kernel, a HY-histogram $q_u(x_c)$ is obtained by Eq. 6.

(b) Calculate the contribution C_u of each color u by Eq. 7.

(c) Go to next frame and $y_0 \leftarrow x_c$.

(d) Evaluate the weight w_i for the point x_i by

$$w_i = q_u \cdot C_u, \tag{10}$$

where $b(x_i) = u$.

(e) Find the new mode y_1 by

$$y_1 = \sum_{i=1}^{N}(w_i \cdot x_i) \Big/ \sum_{i=1}^{N}(w_i). \tag{11}$$

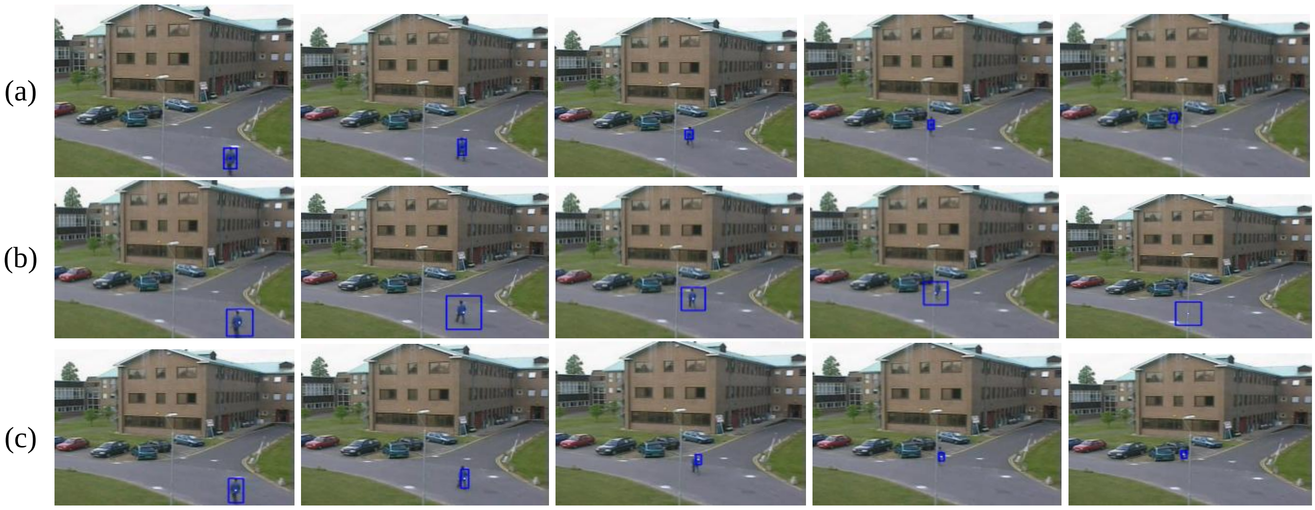
(f) If $\| y_1 - y_0 \| \le \varepsilon$ or number of iterations $\ge M$, then goto (g);
else $y_0 \leftarrow 1/2(y_0+y_1)$ and goto (d).

(g) If end of video then goto (j).

(h) Update q_u and window size by Eq. 9 & Eq. 5.

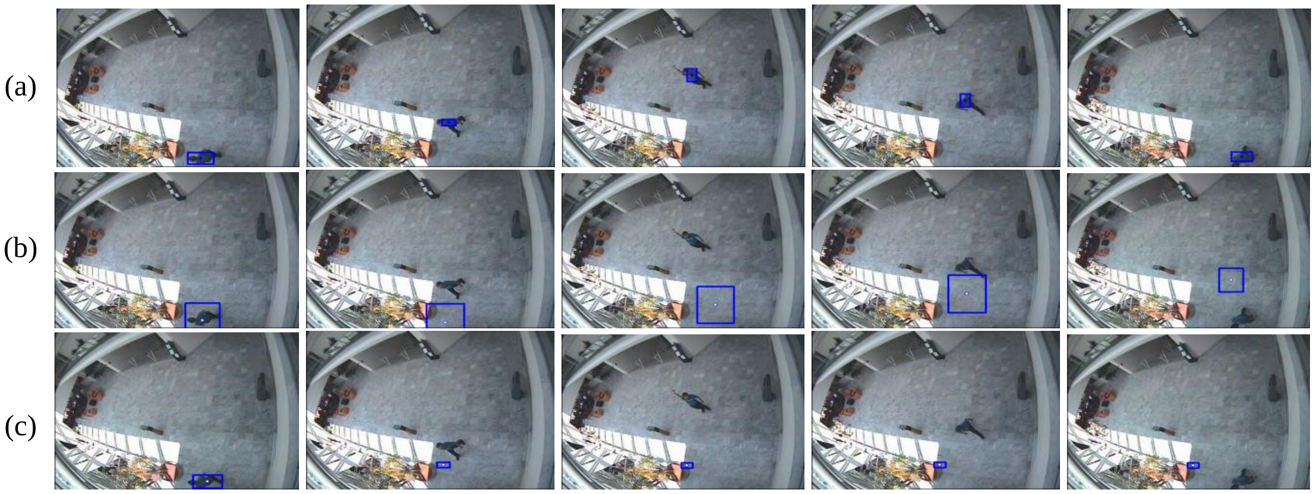
(i) Let $x_c = y_1$ and goto (b).

(j) End.



frame 1 frame 39 frame 91 frame 130 frame 169

Figure 4. V_1, the results from different methods: (a) Ours; (b) CAMSHIFT [5]; (c) Kernel-based [7].



frame 1 frame 23 frame 81 frame 111 frame 136

Figure 5. V_2, the results from different methods: (a) Ours; (b) CAMSHIFT [5]; (c) Kernel-based [7].

Experimental Results

Two videos are tested. V_1 (Fig. 4, from PETS 2001, 384 x 288, 20 fps) shows a pedestrian who went pass a light pole and walked towards a parking lot. V_2 (Fig. 5, from web, 384 x 288, 20 fps) shows a person went up and back in a fast speed. The proposed method is compared with CAMSHIFT [5] and the kernel-based tracker [7]. The related parameters used are the same as suggested in [5] and [7]. The same initial ROIs are used for different methods while testing.

The results are exhibited in Fig. 4, 5, where the row (a) is the proposed method, the row (b) is CAMSHIFT, and the row (c) is the kernel-based method. As shown on these figures, the proposed method has demonstrated stable and satisfactory results. Overall, CAMSHIFT took colors of the background inside the ROI as feature, and this made the ROI bigger than it should be. Kernel-based tracker worked better since the use of kernel; however, it tends to have a smaller ROI as mentioned.

In V_1, the ROI got stuck with the light pole and failed for CAMSHIFT; the kernel-based tracker successfully tracked the target but the localization of ROI is not stable. It is prone to be jumping around the tracked target. In V_2, the colors of the target and the background are mostly achromatic. Using hue alone, as in CAMSHIFT, the feature of the target is not sufficient. As for the kernel-based method, it uses normalized R, G, B (a histogram of 4196 bins), the results are not always the same depending the initial ROI.

Conclusions

This paper presents a hue-and-brightness histogram as the target feature. The 2D histogram not only retains achromatic information, but also describes a target in a more precise way. In addition, flooding process is used to remove background noise so that the histogram faithfully reflects the color information in a target. While tracking by CAMSHIFT, the contribution on each color is considered and the target histogram and the ROI size are adjusted adaptively. Thus, the proposed method is very robust and can solve poor quality problem which is common in surveillance videos. From experimental results on several videos, the proposed method outperformed the existing methods. Nevertheless, the tracking may fail if two persons with the same color information come across each other for the proposed method. How to properly include the motion information without expensive computation would be our future work.

Acknowledgement

Authors would like to thank for the grant support from National Science Council (NSC 101-2221-E-032-054) of Taiwan, R.O.C.

References

[1] H. T. Nguyen and A. Smeulders: Robust Tracking Using Foreground Background Texture Discrimination, *International Journal of Computer Vision*, Vol. 69, no. 3 (2006), pp. 277–293.

[2] H. Stern and B. Efros: Adaptive Color Space Switching For Face Tracking in Multi-Color Lighting Environment, *Proceedings of IEEE Int. Conf. on Automatice Face and Gesture Recognition* (2002), pp.249–254

[3] R. Collins, Y. Liu and M. Leordeanu: Online Selection of Discriminative Tracking Features, *IEEE Journal of Pattern Analysis and Machine Intelligence*, Vol. 27, no. 10 (2005), pp.1631-1643

[4] K. Fukunaga and L.D. Hostetler: The Estimation of The Gradient of A Density Function, *With Applications in Pattern Recognition, IEEE Transactions on Information Theory*, Vol. 21, no. 1 (1975), pp. 32-40.

[5] G. R. Bradski: Computer Vision Face Tracking for Use in A Perceptual User Interface, *Intel Technology Journal*, 2nd Quarter (1998), pp. 13-27.

[6] D. Comaniciu and P. Meer: Mean shift: A robust approach toward feature space analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, no. 5 (2002), pp. 603–619.

[7] D. Comaniciu, V. Ramesh and P. Meer: Kernel-Based Object Tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, no. 5 (2003), pp. 564-577.

[8] T. Lindeberg: Feature Detection With Automatic Scale Selection, *International Journal of Computer Vision*, Vol. 30 (1998), pp. 79-116.

[9] D. Comaniciu, V. Ramesh, and P. Meer: Real-Time Tracking of Non-Rigid Objects Using Mean Shift, *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (2000), pp. 142-149.

[10] J.G. Allen, R.Y.D. Xu and J.S. Jin: Object Tracking Using Camshift Algorithm and Multiple Quantized Feature Spaces, *Proceedings of The Pan-Sydney Area Workshop on Visual Information Processing*, Australian Computer Society, Inc. (2004), pp. 3-7.

[11] C.- W. Lin: Application of Mean Shift to Real-Time Visual Tracking for A Deformable Object. Master Thesis of Mechanical And Electro-Mechanical Engineering Department, National Sun Yat-Sen University (Taiwan, 2009).