IMPROVED KERNEL BASED TRACKING FOR FAST MOVING OBJECT

Dang Xiaoyan, Yao Anbang, Wang Wei, Zhang Ya, Wang Zhuo
Intel Labs China, 8f Raycom Infotech Park A, 2th Kexueyuan South Road, Beijing, China

Wang Zhihua
Micro Electronic Institution, Tsinghua University, Beijing, China

Keywords: Object Tracking, Kernel Based Tracking, Foreground/ Background Modelling.

Abstract: A novel approach of discriminative object representation and multiple-kernel tracking is proposed. We first employ a discriminative object representation, which introduces the foreground and background modelling ingredient to select the most discriminative features from a set of candidates via classification procedure. In the context of using kernel based tracking algorithm, a multiple-kernel strategy is employed to handle the difficulties resulted from fast motion through refining the ill-initialization position according to pre-refinement method. Extensive experiments demonstrate that the proposed tracker works better than Camshift and traditional kernel tracker.

1 INTRODUCTION

Real time object tracking is a critical task in many computer vision based applications such as surveillance, perceptual user interfaces, augmented reality, smart rooms, video compression and driver assistance in (Dorin, 2003) (Klaus, 2001), (Faith, 2005), (Hanger, 2004), (Ahmed, 2002) and (Arulampalam, 2002). Compared with other commonly used approaches like particle filter by (Arulampalam, 2002), kernel based method by (Dorin, 2003) and (Klaus, 2001) have gained more and more attention mainly due to its low computation cost, easy implementation and competitive performance.

Given an object of interest in the previous frames, the problem of object tracking is to precisely label the object locations in the remained frames. In kernel based tracking method, the target model is represented as a kernel weighted color histogram. As for object location, it is iteratively obtained through mean shift and gradient decent techniques. However, kernel based tracking faces the problem of how to design appropriate kernel for adapting complex object appearance changes, 3-D rotations and object deformations. Additionaly, it strictly depends on the assumption that object regions overlap between the consecutive frames. That means kernel based tracking will completely fail when the object moves too fast or video is sampled in very low frame rate (Faith, 2005) because these usually result in little or no overlapped regions in consecutive frames. Reference (Faith, 2005) proposed the multi-kernel tracking method to resolve this problem. Even though the tracking performance is improved, their approach imposes much more computation burden in the iterative procedure. (Hanger, 2004) tried to resolve this problem by replacing the Bhattacharyya coefficients with Matusita metric, which could better resolve the problem in math.

In order to achieve robust tracking under difficult scenarios (e.g. cluttered background, fast moving) with moderately low computation cost, we introduce a discriminative multiple kernel tracking method. First we introduce discriminative linear color feature to represent the object of interest, which works well under background clutter situation and further suppresses the object from drifting into wrong area when the background is similar to the object. Second we introduce an efficient two-step multiple-kernel tracking method instead of the baseline kernel tracking method (Klaus, 2001) to handle fast moving cases, where the first step is trying to correct the insufficient initial location for kernel tracking and

Xiaoyan D., Anbang Y., Wei W., Ya Z., Zhuo W. and Zhihua W. (2010).
IMPROVED KERNEL BASED TRACKING FOR FAST MOVING OBJECT.
In *Proceedings of the International Conference on Computer Vision Theory and Applications*, pages 261-266
DOI: 10.5220/0002766902610266

the second step is employed to estimate object location through mean shift iteration. The proposed method introduces negligible computation cost into the system but resolves the quick moving problem with favorable performance.

This paper is organized as follows: section 2 describes the discriminative object representation method; section 3 deals with the selective kernel based tracking. Section 4 gives the experimental results. Section 5 illustrates some discussion topic, and section 6 concludes our article.

2 OBJECT REPRESENTATION

Traditional kernel tracking like (Klaus, 2001) uses discrete color distribution as the target representation. For example, discrete RGB or YUV color distribution of target at location y is represented as $p(y)$, and then transformed into m-bin histograms:

Target model:

$$\hat{q} = \{\hat{q}_u\}_{u=1\ldots m}, \sum_{u\text{-}1}^{m} \hat{q}_u = 1 \quad (1)$$

However, only simple color representation of object is not discriminative enough, background information shows its importance for several reasons. First, some of the target features also present in the background, their relevance for the localization of the target is diminished. Second, in many cases, target model always contains background features. The traditional simple m-bin color histogram representation fails under these conditions.

Inspired by (Ahmed, 2002) and treating object representation as a regional detection problem, we propose a discriminative combined color feature. Based on the assumption that the background close to the target has the biggest influence to target representation, we only introduce this salient area into our representation. We search in the sub-space of RGB space to find the most discriminative combination of RGB information.

2.1 Foreground and Background Region Selection

Suppose we already select object initialized with a manual labeled rectangle region, we only need to determine the correlated background region. Since kernel based tracking algorithm has limited operational basin, we choose the background as the region of baseline kernel tracker's operational basin, which is the double area of target. An illustrative sample image is shown in Fig.1.

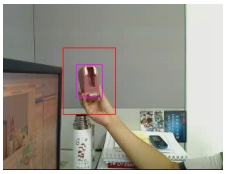


Figure 1: Object/background area sampling, inner rectangle is object area, outer rectangle is background edge.

As we can see in Fig.1, the red rectangle area minus the magenta rectangle area is the background area. We try to find a cluster of RGB sub-space to best discriminative target from background.

2.2 RGB Sub-space Selection

In tracking tasks, pure RGB pdf could not always perform well in the situations of illumination changes and background clutter. To this problem, its linear combination sub-spaces may have better performance.

We randomly generate a cluster of linear combination of RGB information, and try to find the best ones among them.

The whole procedure is as follows:

- Randomly generate feature set coefficients; Where $\alpha_i, \beta_i, \gamma_i$ are the coefficients for the ith linear combination space and M is the number of spaces, normally, we generate about 20-30 feature sets.

$$\{\alpha_i, \beta_i, \lambda_i\}_{i=1,\ldots M}, \quad \alpha_i, \beta_i, \lambda \in \mathbb{Z} \quad (2)$$

- Generate feature sets according to (3) with coefficient selected from (2): i.e. weighing RGB color components with different coefficients. Here x is the 2-D location of one pixel in image. These features are normalized to [0, 255].

$$f_i(x) = \alpha_i R(x) + \beta_i G(x) + \lambda_i B(x) \quad (3)$$

- Finally, we map histograms of object and background region in each feature space as $h(o)_i=\{h(o)_{i,j}\}$ for object region, $h(b)_i=\{h(b)_{i,j}\}$ for background region, $i=1,2\ldots M$, $j=1,2\ldots N$. Here N is the bin number of histogram.

Fig.2 shows examples of image transformed into feature space.

Based on the described process, we selected the most discriminative features, and map the whole image into the selected feature space.

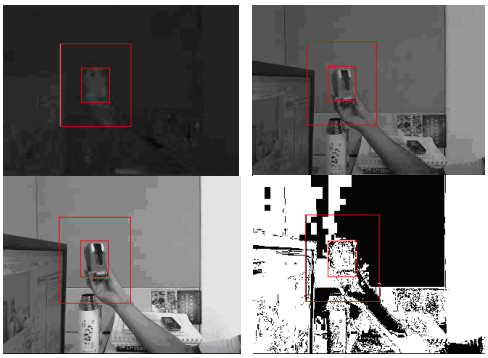


Figure 2: Several sample images in feature space.

2.3 Feature Selection Base on Fisher Measure

With the selected M feature spaces, we try to find the one who can best discriminate the object of interest from the background. Based on Fisher-criterion, we formulate the foreground-to-background log likelihood distribution as

$$l_i = \sum_{j=1}^{N} \ln\left(\frac{\max\left(h_{i,j}(o), 1e-6\right)}{\max\left(h_{i,j}(b), 1e-6\right)}\right) \quad (4)$$

Here $h_{i,j}(o)$ and $h_{i,j}(b)$ is the histogram of jth bin in ith feature space for object and background.

We then evaluate the discriminative ability of each feature with the log likelihood distribution in (4). Since larger li in (4) corresponds to higher $h(o)_i$ and lower $h(b)_i$, which means lower discriminative ability. Based on the l_i values, we can order the M space in descending discriminative ability, and select the best J features for object representation with the highest l value.

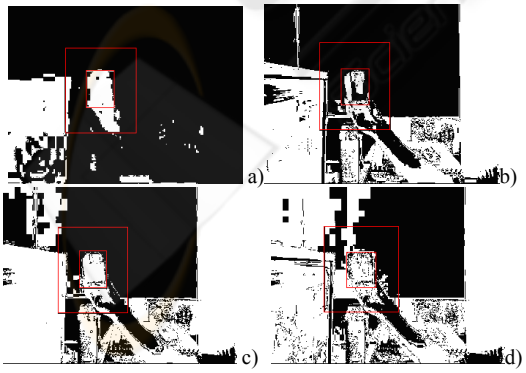


Figure 3: Four sample observation likelihood images mapped from different feature images sorted in discriminative ability descending order.

We show 4 discriminative images with the best discriminative ability in Fig.3.

From Fig.3, we can find that different linear combination of color information has different discriminative ability. For example, in image a) the object in the inner rectangle is almost totally white, and the selected background is almost all black, which means feature set in a) has the best discriminative ability.

2.4 Discriminative Combination Color Descriptor

After we made the decision of the J linear combinations, we concatenate the J histograms generated from the J selected spaces to form our discriminative linear color feature.

Define $\oplus$ as the operator of concatenate two vector, we generate final feature according to (5).

$$h = h_1 \oplus h_2 \oplus \ldots \oplus h_J \quad (5)$$

This discriminative combination of color feature introduces background influence into the selection of feature generation, and selects the best discriminative linear descriptor in RGB sub-spaces, which is totally different from the traditional way like (Klaus, 2001) and (Faith, 2005). Experiment results show it could perform better in cluttered background, illumination change and appearance variation which are normal in real applications.

3 AMENDED KERNEL OBJECT LOCALIZATION

In real applications, the object usually undergoes unpredictable movements, e.g. quick move, outburst direction change. However, traditional kernel tracking method like (Klaus, 2001) strictly depends on the assumption that object regions overlap between the consecutive frames. That is, it will fail under these unpredictable fast movement situations.

To handle this problem with little computation cost burden, we suggest an initialization position refinement + kernel tracking method, where the ill-conditioned convergence of kernel based iteration is moderately suppressed to get better result with low computation cost.

3.1 Refinement of Initialization Position

The basic idea of our approach is to reallocate the initial object position in the true target centered

region via quickly estimating an observation likelihood surface against target model. That is, the initial object position should be located in the true target region.

3.1.1 Multiple Initialization Position Selection

If we define quick movement as the objects in consecutive frames do not overlap, then quick movement is high dependent on the size of the object. The smaller the object is, the more possible it moves too fast to track.

Based on this assumption, the locations where we put our multiple initialization positions are decided using piecewise linear function *f(x)* dependent to the size *(x, y)* of the object.

$$m\_step.x = \begin{cases} x, & 0<x \le 10 \\ x/2, & 10<x \le 20 \\ x/3, & 30<x \le 40 \\ x/4, & x>40 \end{cases} \quad (6)$$

Here x is vertical size and y is horizontal size. The same strategy is used on vertical axis y as (6). We put the $K \times K$ locations according to (6). Fig.4 shows some examples of the multiple locations.

a) b)

Figure 4: Illustration of multiple initialization strategy.

In Fig.4 the dark-blue rectangle defines the object and the background, while the light-blue circles denote the candidate multiple initialization positions, while red-color circle indicates the current target center.

We can see from Fig.4 that object in a) and b) are different, and the step for a) and b) is different due to the relationship defined in (6). The step when selecting the multiple positions is highly dependent with the object size, which is due to the relationship between quick motion definition and object size.

3.1.2 Refinement of Initialization Position

Based on the selected location surface, we conduct a refinement process before kernel based tracking to select a modified kernel initialization position.

- Partition the region of observation likelihood surface into $K \times K$ sub-regions.
- For each sub-region, a Gaussian formed stochastic sampling is employed to generate the candidate position of each sub-region.
- Generate the histogram of each candidate sub-region.
- Measure the similarity between target model and each candidate histogram, and choose the candidate with largest similarity as refined initial object position. The histogram distance measure we used is Bhattacharyya coefficient.

If the largest similarity ρ_{Mf} and the second largest similarity ρ_{Ms} are very close to each other, a weighted procedure is used.

$$x = \frac{\rho_{Mf} x_{Mf} + \rho_{Ms} x_{Ms}}{\rho_{Mf} + \rho_{Ms}} \quad (7)$$

Where x_{Mf} and x_{Ms} are the candidate positions and x is the final decided initialization position.

3.2 Kernel Function

With the modified initialization position, a baseline kernel tracking is employed. In our algorithm, we use *Epanechnikov* profile:

$$k(x) = \begin{cases} \frac{1}{2} C_d^{-1} (d+2)(1-x), & if\ x \le 1 \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

Although we have other choices of kernel like *roof kernel* in (Hanger, 2004), we choose *Epanechnikov kernel* and *Bhattacharyya* distance measure for the convenience of comparison.

4 EXPERIMENTAL RESULT

To show the performance of the proposed tracking approach, we compare it with Camshift, and baseline kernel based tracker. In the experiments, the scenarios of object fast movement, rapid camera motion and object appearance change are considered. As for tracking accuracy, it is represent as the error between the estimated object positions and manual labeled ground truth.

4.1 Comparison Under Fast Object Motion

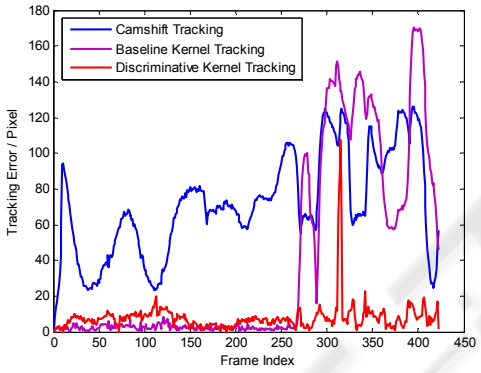
In the first experiment, fast object movement is considered. The video sequence, which is recorded with a Logitech Tessa 2.0 camera, has 450 frames. In this video, the aimed glass undergoes normal speed motion in the first 250 frames and moves with a quick unpredictable speed in the remained frames.

The tracking error curves of our approach, Camshift and kernel based tracker are shown in Fig.5 a). Some example tracking frames of our approach are given in Fig. 5 b).

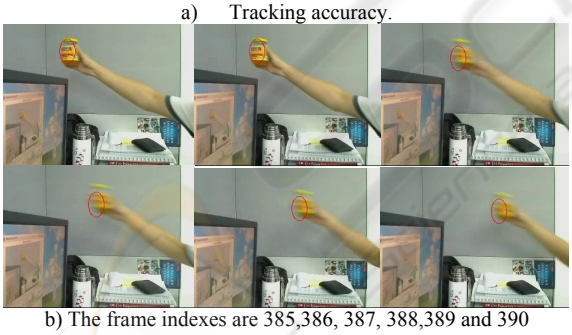
The tracking error e is defined in (9)

$$e = \sqrt{e_x^{\,2} + e_y^{\,2}} \tag{9}$$

Where e_x is the pixel error in horizontal axis x, and e_y is the pixel error in vertical axis y.



a) Tracking accuracy.



b) The frame indexes are 385,386, 387, 388,389 and 390 respectively.

Figure 5: Comparison of tracking accuracy.

We can find from Fig.5 a) that the proposed algorithm works well both at normal motion speed and quick motion speed. Camshift works worst under both situations. Baseline Kernel tracker works well under normal motion speed, but it fails to track the object in the frames of quick motion. In general, the proposed algorithm exhibits the enhanced tracking capability in handling the difficulties resulted from rapid object motion.

4.2 Tracking Under Different Situations

4.2.1 Rapid Camera Motion

The second experiment is mainly focused on rapid camera motion. In contrast with fast object motion, when the camera is not static, the background scene will change and the field of view will also vary. As a result, the tracking task becomes difficult. Fig 6 illustrates some representative tracking frames of our approach. Here, we want to point out that these frames are collected from real-time processing environment where camera moves rapidly but object is still.

Figure 6: Tracking results of successive frames under camera rapid motion.

It can be seen from Fig.6 that our proposed approach can also deal with the rapid camera movement in an effective way.

4.2.2 Object Appearance Change

Figure 7: Tracking result under different hand posture.

The third experiment is implemented on a hand video sequence in which the hand changes in its appearance and pose. In addition, background is complex. Fig.7 indicates with some tracking frames of our approach.

It can be seen from Fig.7 that the proposed approach also shows good ability to adapting to hand appearance and pose changes.

4.2.3 Tracking in Low Frame Rate

The fourth experiment is implemented on a down-sampled outdoor pedestrain video sequence, which contains typical fast object motion due to low sample rate as in (Faith, 2005). Fig 8 indicates with some tracking frames of our approach.

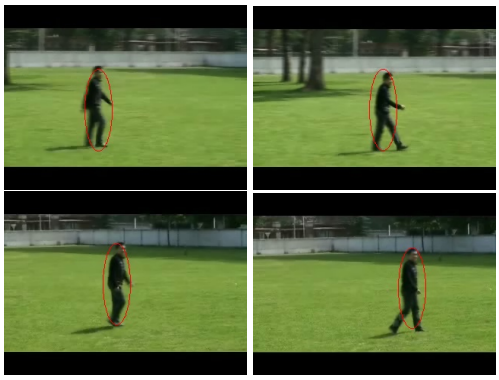


Figure 8: Tracking results of successive frames under low sample rate (frame # 94, 95, 142, and 143).

In Fig.8, the outdoor pedestrian video is down-sampled like (Faith, 2005), which contains typical rapid motion. We can see that the objects in consecutive frames 94 and 95 have no overlap, and so do frame 142 and 143. The proposed algorithm works well on this dramatically rapid motion video. Notice that the distance of consecutive object is comparatively bigger in the scenario, which is caused by the heavy down-sampling for testing.

5 DISCUSSION

Rapid movement of object using normal camera always accompanies with motion blur of image, which will result in the color drift and content degeneracy of the image due to long exposal time. That is, object appearance will be deformed or blurred under fast motion. Here, it should be pointed out that sample images in Fig.5 b) show that the proposed algorithm works well in above situations.

Based on the description of amending strategy in part 2 and 3, the extra computation cost of discriminative linear color feature and pre-refinement strategy before kernel tracking is moderately low, and hence the proposed tracking algorithm can be used in real-time tracking tasks unlike the works of (Faith, 2005) and (Arulampalam, 2002).

Proposed method uses a pre-refinement method to modify the ill-conditioned initialization position before kernel tracker, which is somewhat a fake multiple-kernel strategy. It is similar but different with (Faith, 2005) for we resolve using pre-processing with low computation cost and (Faith, 2005) use real multiple-kernel and post fusion of multiple position with high computation cost.

6 CONCLUSIONS

In this paper, we propose an efficiently discriminative combination of color feature for tracking problem, which introduces foreground / background classification idea into object representation. Also we propose a low-cost pre-refinement method to better resolve the ill-initialization problem of kernel tracker, which could enhance the performance of kernel tracker under object rapid motion. With respect to experiment results, our proposed representation and multiple kernel strategy works better than popularly used Camshift and BKT under quick motion situation. It also partly diminishes the effect of background cluster and illumination change's influence on tracking result.

REFERENCES

Dorin Comaniciu, Visvanathan Ramesh, Peter Meer, 2003. Kernel-Based Object Tracking, *IEEE transaction on Pattern Analysis and Machine Intelligence,* Vol 25, Issue 5, pp. 564-577.

Klaus Robert Muller, Sebastian Mka, Gunnar Ratsch, 2001. An Introduction to Kernel Based Learning Algorithms, *IEEE Transactions on Neural Networks*, Vol 12, No 2, pp. 181-201.

Faith Porikli, Oncel Tuzel. 2005. Multi-Kernel Object Tracking, *ICME*, pp. 1234-1237.

Hanger G D, Dewan M. 2004. Multiple Kernel Tracking with SSD, *Proc. CVPR*. Vol.1, pp. 790-797.

Ahmed Elgammal, Ramani Duraiswami, David Harwood. 2002. Background and Foreground Modeling using nonparametric kernel density estimation for visual surveillance. *Proceedings of the IEEE* , Vol 90, Issue 7, pp. 1151-1163.

Arulampalam, M. S. Maskell, S. Gordon. 2002. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transaction on Signal Processing*. Vol 50, No 2, pp. 174-188.