International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.8, No.11 (2015), pp.273-282
http://dx.doi.org/10.14257/ijsip.2015.8.11.25

Online Mean Kernel Learning for Object Tracking

Lei Li[1, 2], Ruiting Zhang[3], Jiangming Kan[1] and Wenbin Li[1]

[1]*School of Technology, Beijing Forestry University, 100083, Beijing, China*
[2]*Institute of Atmospheric Physics, Chinese Academy of Sciences, 100029, Beijing, China*
[3]*Canvard College, Beijing Technology and Business University, 101118, Beijing, China*
Corresponding Author: kanjm@bjfu.edu.cn

Abstract

Features for representing the target are the fundamental ingredient when constructing the appearance model in the tracking problem. Only one type of features is utilized to represent the target in most current algorithms. However, the limited representation of a single feature might not resist all appearance changes of the target during the tracking process. To cope with this problem, we propose a novel tracking algorithm - Mean Kernel Tracker (MKT) - to robustly locate the object. The MKT combines three complementary features - Color, HOG (Histogram of Oriented Gradient) and LBP (Local Binary Pattern) - to represent the target. And Extensive experiments on public benchmark sequences show MKT performs favorably against several state-of-the-art algorithms.

Keywords: *Sparse representation, object tracking, online mean kernel learning*

1. Introduction

Visual object tracking is an important problem in computer vision and has many applications including traffic monitoring, vehicle navigation and human computer interface (HCI), just to name a few. Although it has been investigated in the past decades [1, 2, 9-18], designing a robust tracker to cope with different objects in various scenarios is still a great challenge. A very common difficulty is to resist visual appearance changes frame by frame due to pose, sudden illumination changes and partial occlusion [3]. Such changes may make a tracker drift away from the target object.

To deal with appearance changes of the target, Collins *et. al.*, [4] proposed an online mechanism to pick out the top-ranked color features from a set of seed features based on the object/background discrimination. Hare *et. al.*, [19] applied 6 different types of Haar-like feature arranged on a grid at 2 scales and structured SVM to construct the object appearance. Kalal *et. al.*, [20] proposed a robust detector/tracker, TLD (tracking, learning, detection), which combined the random forest [8] based detector and KLT tracker [5-21]. The detector adopted a kind of feature named 2-bit Binary Patterns to learn the appearance of the target.

Many of the aforementioned approaches could not deal with all various appearance changes simultaneously due to the limited representation of the single feature used in the systems. One strategy is to design a strong feature which is robust to any appearance change of the target. Unfortunately, it is not an easy task [22-23], especially for situations where no prior knowledge of the target is available except for its initial location. Another strategy is to adaptively combine a few complementary image representations (*e.g.*, image features based on color, edge and texture information), of which each feature descriptor maps the target into one feature space and encodes one channel of information of the target. The complementary features can resist others' changes due to the

ISSN: 2005-4254 IJSIP

appearance changes, *e.g.*, illumination changes, which contribute to a robust system. Du and Piater [24] integrated multiple features, edge, and color in a probabilistic framework. The target is tracked in each feature by a particle filter and different particle filters interact each other via a message passing scheme. More recently, Kwon and Lee [25] proposed a tracking framework Visual Tracking Decomposition (VTD). VTD constructed the appearance model using the initial frame and the latest four frames by means of sparse principal component analysis (SPCA) of a set of complementary feature templates which dealt with change of pose, lighting. Nevertheless, due to the generative representation scheme, VTD might not distinguish target and background and get wrong updating of appearance model.

In this paper, to represent the target well enough, we use three complementary image representations: Color, histogram of oriented gradients (HOG) [26] and local binary pattern (LBP) [6]. Multiple kernel learning (MKL) [27, 7] has been widely used for feature combination in the object detection [28] and classification [23]. However, due to the sparsity in MKL, only one feature might be selected or have a major weight, which would not attain the goal of feature combinations. Therefore, to combine features effectively, We utilized a special edition of MKL, the Mean Kernel Learning, in which the weights of different features was set manually rather than learned. And this simplification also makes the optimization solved efficiently. Extensive experiments on many benchmark videos have shown that our Mean Kernel Tracker can outperform state-of-the-art tracking algorithms.

The rest of this paper is organized as follows. In Section 2, every part of the novel Mean Kernel Tracker (MKT) is described in detail. Then the novel tracking algorithm and implementation details is given in Section 3. Experimental results and comparison with other state-of-the-art approaches are presented in Section 4. Section 5 summarizes our work.

2. Mean Kernel Learning

In this section, we give a detail of how to combing multiple features via the Mean Kernel Learning framework. The multiple kernel learning is to learn the weighted combination of different kernels. In this work, one type of feature only corresponds to a kernel, and the kernel and feature are the same meaning if no other explanation. Given an image patch x, one can extract more than one type of feature. And different features are combined as the following formulation:

$$k(x_i, y_i) = \sum_{m=1}^{M} d_m k_m(f_m(x_i), f_m(x_j))$$
$$\sum_{m=1}^{M} d_m = 1, d_m \geq 0, \forall_m = 1 \cdots M; \tag{1}$$

where $f_m(x_i)$ is the m^{th} feature of image patch x_i, the weight d_m stands for the importance of the m^{th} kernel $k_m(f_m(x_i), f_m(x_j))$ and M is the total number of features.

Among many existing MKL learning algorithms, Simple MKL [7] enables to learn the weight d_m very efficiently. And d_m is learnt via optimizing the following optimization problem:

$$\min \frac{1}{2}\sum_{m=1}^{M}\frac{1}{d_m}\|w_m\|^2 + C\sum_{i=1}^{N}\eta_i$$
$$s.t. y_i(\sum_{m=1}^{M} w_m^T \phi(f_m(x_i)) + b) \geq 1 - \eta_i, \forall_i = 1 \cdots N; \tag{2}$$
$$\sum_{m=1}^{M} d_m = 1, d_m \geq 0, \forall_m = 1 \cdots M;$$

where y_i is the label of x_i, $\phi(f_m(x_i))$ is the implicit mapping of $f_m(x_i)$, C is the penalty parameter, η_i is the slack factor, b is the bias and N is the total number of training examples. How to solve Simple MKL efficiently can be found in detail in [7].

Due to the sparse penalty on d_i, Simple MKL often assigns a major weight to one feature. This would not attain the goal of feature combinations to resist the appearance changes of the target. Therefore, we adopted a simple edition of Simple MKL, Mean Kernel Learning to learn the appearance model. In Mean Kernel Learning, the weights of different kernels are manually set fixed $d_m = 1/M$ rather than learned automatically.

Therefore, Problem (2) can be simplified as follows:

$$\begin{aligned} &\min \frac{1}{2}\sum_{m=1}^{M}\frac{1}{M}\left\|w_m\right\|^2 + C\sum_{i=1}^{N}\eta_i \\ &s.t. y_i(\sum_{m=1}^{M} w_m^T\phi(f_m(x_i)) + b) \geq 1-\eta_i, \forall_i = 1\cdots N; \end{aligned} \tag{3}$$

By introducing Lagrange multipliers, we get the dual of Mean Kernel SVM:

$$\begin{aligned} &a^* = \min_a \frac{1}{2}a^T YKY_a - \sum_{i=1}^{N} a_i \\ &s.t.(\sum_{i=1}^{N} a_i y_i = 0, 0 \leq a_i \leq C, \forall_i = 1\cdots N; \end{aligned} \tag{4}$$

where $Y = diag(y_1, y_2 \cdots y_N)$, and $K_{ij} = \frac{1}{M}\sum_{i=1}^{M} K_m(f_m(x_i), f_m(x_j))$. This optimization problem can be solved easily using the tool LIBSVM [29]. The final decision function of Mean Kernel Learning can be expressed as:

$$F(x) = \frac{1}{M}\sum_{i=1}^{N} a_i^* y_i \sum_{i=1}^{M} K_m(f_m(x_i), f_m(x_j)) + b \tag{5}$$

In Problem (4), the optimization problem is the same as the classical SVM. However, the difference is that the affinity matrix K in Mean Kernel SVM is an average of different kernel function. This combines different features seamlessly and resists the changes of one feature channel and make the system more robust.

3. Details of the Implementation

In this section, we give the implementation details of our tracking algorithm. And then we summarize the tracking algorithm.

3.1. Preparation of Training Sets

Let l_t^* denote the center position of the target in the t^{th} frame. And x is an image patch and $l_t(x)$ is the column vector of its center position in frame t. About n examples are randomly sampled from $Z \equiv \left\{x : \gamma_l < \left\|l_t(x) - l_t^*\right\| < \beta_l\right\}$ to form the negative training set X_t^n. In our tracking system, we set the positive training $X_t^P = \left\{x : \left\|l_t(x) - l_t^*\right\| < a_l\right\}$, which means all image patches around the target in the current frame are sampled as positive ones. After constructing the training set, all the patches are normalized to the same size (32 × 32 in our experiments) for efficiency.

3.2. The Features for Tracking

Color Feature contains rich visual information, but is easily affected by large illumination changes. The image patch is down-sampled into 16 × 16, and the pixel value is concatenated and formed as the corresponding feature vector.

Edge Feature is more robust than color feature to illumination changes but may not be adaptive to background clutter scene. In our system, HOG feature [26] is extracted on the normalized patches and we use the implementation of HOG in Open CV.

Texture Feature can capture small detail of the target. LBP feature [6] is used for its efficiency and resistance to lighting effects. The normalized image patch is divided into 2 × 2 blocks. The LBP feature of each block is extracted and concatenated into a longer feature vector.

3.3. Parameter Settings

In our tracking system, all the parameter settings are fixed in our experiments. $a = 3$, $\beta_l = 16$, $\gamma_l = 40$, the penalty parameter $C = 0.5$ and about $n = 65$ negative examples are sampled. The Guassian Kernel is used for measuring the similarity of image patches in the feature space. And the bandwidth σ is 0.05, 0.1 and 0.2 for Color, HOG and LBP features respectively.

The entire procedure of the proposed tracking algorithm, Mean Kernel Tracker (MKT), is summarized in Algorithm 1.

Algorithm 1 Mean Kernel Tracker

Input: Frames 1, 2, 3, . . ., and the initial location, l_1^*, of the object in frame 1;

Out Put: object locations, l_2^*, l_3^* ..., in the subsequent frames.

1: $t = 1$;
2: Construct the training sets, X_t^p and X_t^n (Section 3-A);
3: Extract the Color, HOG and LBP features (Section 3-B);
4: Learn the appearance model $F(x)$ (Problem(4) and Eq.(5));
5: Randomly sample 500 candidates with scale change around l_t^*;
6: Extract features and find the patch x^* with the maximum score $F(x^*)$ using the classifier function $F(x)$ (Eq.(5));
7: $t = t + 1$, go to 2.

4. Experiments

We implement our Mean Kernel Tracker (MKT) in C++ language and evaluate its performance on public challenging image sequences. MKT is also quantitatively compared with some state-of-the-art tracking algorithms, FragTrack (Frag) [9], MILTracker (MIL) [15], VTD [25], VTS [17] and Struck [19]. Their codes or executable programmes are publicly available and the parameters are turned finely. All algorithms are compared in terms of the same initial positions in the first frame. And the tracking results might slightly different as stated in their papers, for the initial positions in some sequences were set as in MILTracker [15]. We also implement the Simple MKL Tracker (SimMKT). The weights of different features in SimMKT are learnt automatically, which is the only difference to MKT.

To evaluate the performance among the above algorithms and ours, the center errors between the tracking results and the ground truth are calculated and reported on image sequences. Due to the randomness in our tracking algorithm, we run our tracker five times on every sequence and average the results.

Throughout the experiments, we evaluated all approaches on the following sequences: Sylvester, David, Girl,FaceOcc, FaceOcc2, Tiger1, Tiger2, CokeCan and Shaking. The first eight sequences are from [15] and the last one from [25].

Illumination Change is the common change in the tracking problem, which might contribute to the target in the new frame dramatically dissimilar as before. This dissimilarity can effect the features based on color intensity (*e.g.*, Haar-like feature) much more than the features based on edges. Therefore, the combination of complementary features may resist the illumination change better than single features. In comparison with MILTracker and VTS and Struct Tracker, our tracker is more robust against illumination change. In Sequence David (Figure 1(a)), MILTracker introduced tracking errors since frame 46 when the target moved from the darker place to a brighter one. This phenomenon is much more obvious at frame 394 due to the sudden change of illumination. But our tracker can track the target very well. This can also be found in Sequence CokeCan (Figure 1(e)) since frame 117 and Sequence Shaking (Figure 1(f)) since frame 117. MILTracker and Struct Tracker performed unwell while our tracker still held the target tightly. VTS worked better on Sequence Shaking than our tracker but it totally lost the target on Sequence CokeCan.

Occlusion is a very challenging problem for robust tracking. Struct Tracker worked slightly better than ours on Sequence FaceOcc and FaceOcc2 (Figure 1(c)) because the target shows little appearance change except for occlusion.

However, if there existed changes of target appearances, our tracker worked much better on Sequence Tiger1 (Figure 1(d)) and Tiger2.

When occlusion happened, the tracker might failed if we only used recent a few frame to update the appearance model. This could be found in VTS on Sequence Tiger1 (Figure 1(d)) and CokeCan (Figure 1(e)). But our tracking algorithm exploited the tracked results, this could keep the most valuable information we have gotten. Even if the occlusion happened, the training set still had the target information for modeling the appearance. When the target reappeared, the tracker still could catch up the target. This could be found On Sequance Girl (Figure 1(b)) and CokeCan (Figure 1(e)).

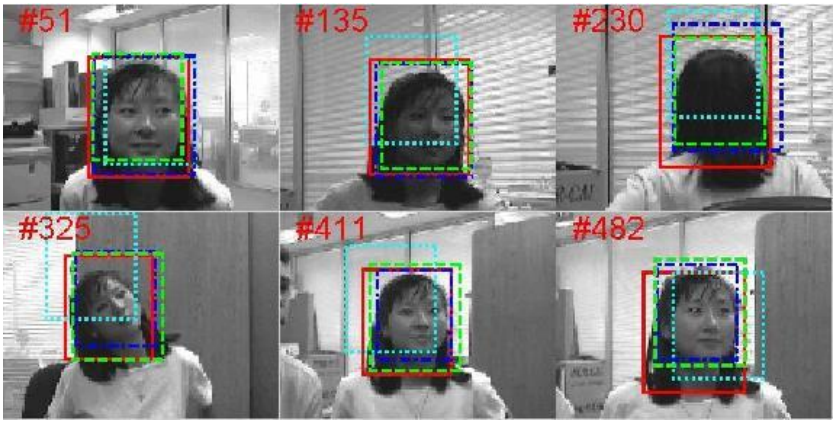
Table 1. Average Center Errors between the Tracking Results and their Ground Truth. The bold represents the best result, and the underlined the second Best

Sequence	Frag	MIL	VTD	VTS	Struck	SimMKT	MKT
Sylvester	11	11	20	18	17	<u>7</u>	**6**
David	46	23	14	13	54	<u>7</u>	**5**
Girl	27	32	13	<u>12</u>	**10**	23	<u>12</u>
FaceOcc	**6**	27	11	11	9	14	10
FaceOcc2	45	20	**7**	<u>9</u>	**7**	13	<u>9</u>
Tiger1	40	15	30	12	20	<u>5</u>	**4**
Tiger2	38	17	20	20	9	<u>9</u>	**5**
CokeCan	63	21	46	48	9	<u>8</u>	**7**
Shaking	84	38	<u>8</u>	**7**	52	55	17
fps	0.5-1	6-8	0.3-0.5	0.3-1	2-4	1-2	1-2

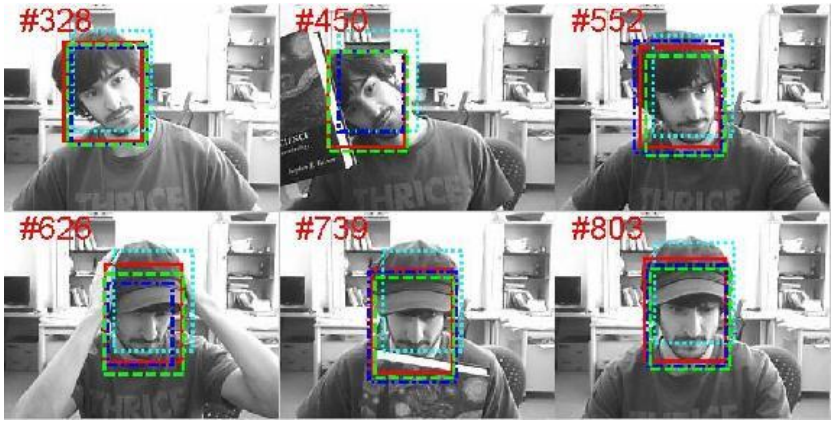
Background Clutter might hijacked the tracker, for the complex environment might contain a blob that might be similar to the target in one single feature space. This could be found in Struct Tracker in Sequence David (Figure 1(a)). The tracker slightly got wrong update at frame 206 and totally lost the target at frame 295. Struct Tracker also lost the target in Sequence Tiger1 (Figure 1(d)) at frame 245. MILTracker and VTS also performed poor performance on Tiger1 (Figure 1(d)) and CokeCan (Figure 1(e)). On these background clutter sequences, our tracker worked better than the others, for complementary features represented the target well enough.

The whole quantitative comparisons are shown in Table 1 and Figure 2. It can be concluded that our track was almost always more robust than others.

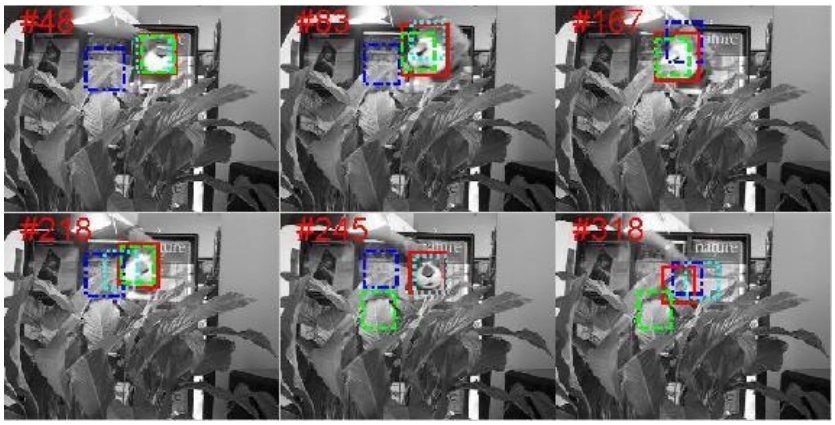
(a) David



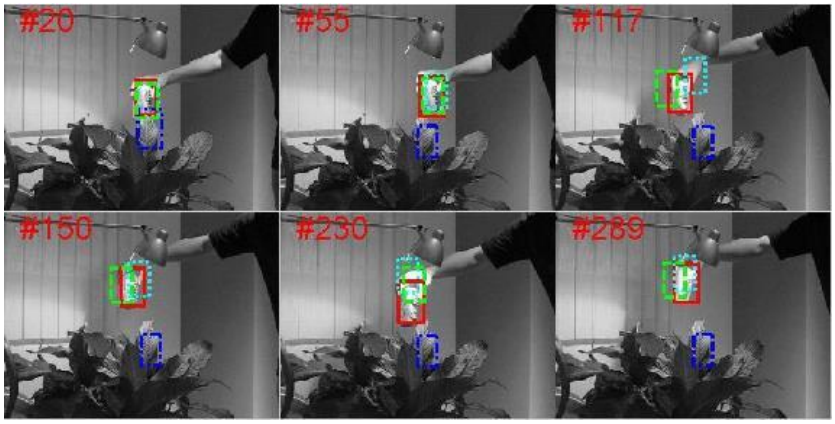
(b) Girl



(c) FaceOcc2



(d) Tiger1



(e) CokeCan

(f) Shaking

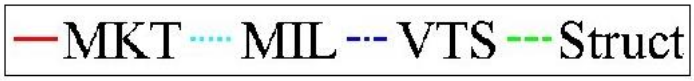
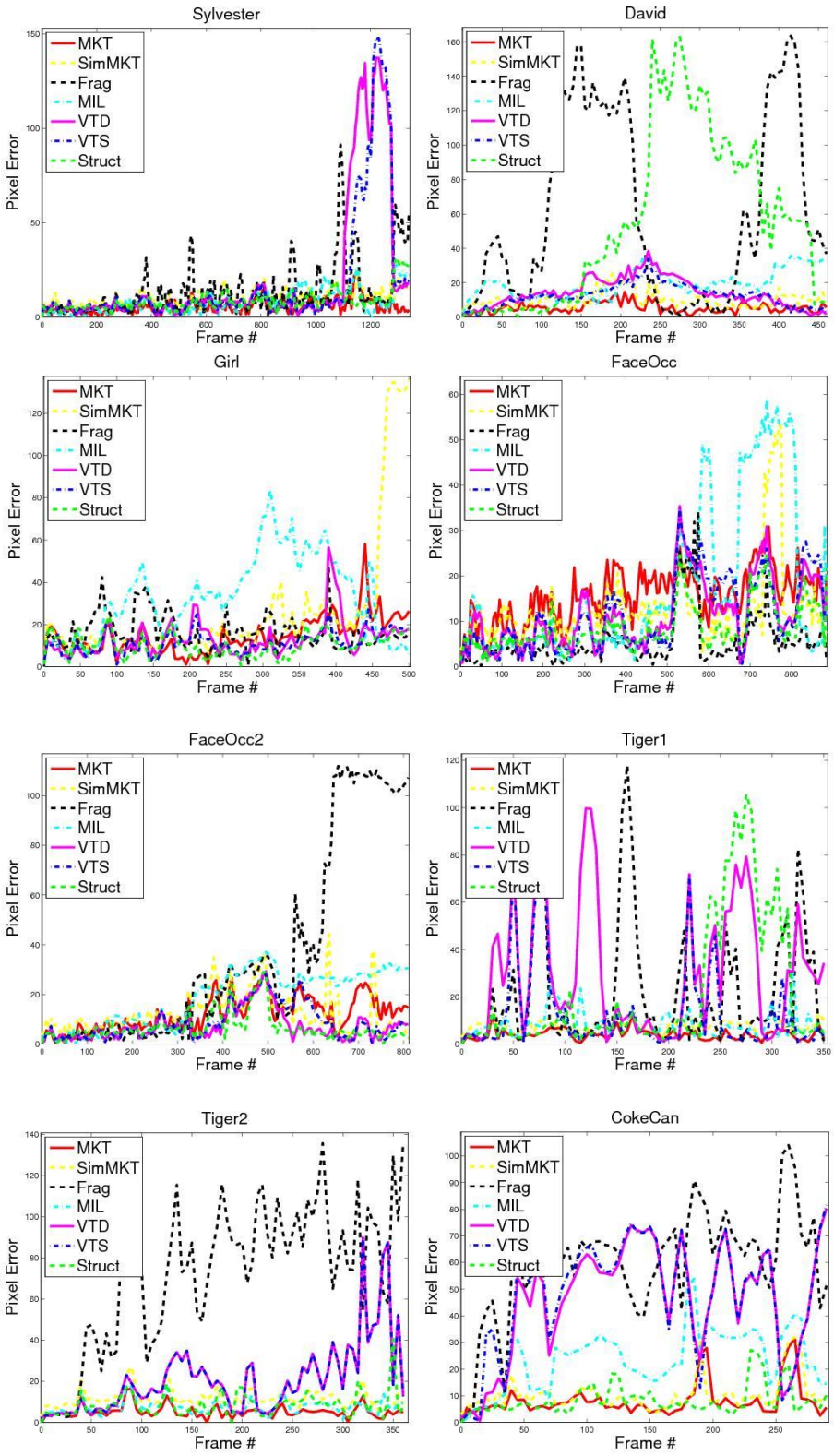


Figure 1. Representative Frames of 6 Sequences: cyan, blue, green and red object bounding boxes generated by MILTracker [15], VTS [17], Struct Tracker [19] and our MKT, respectively. This figure is best viewed in color



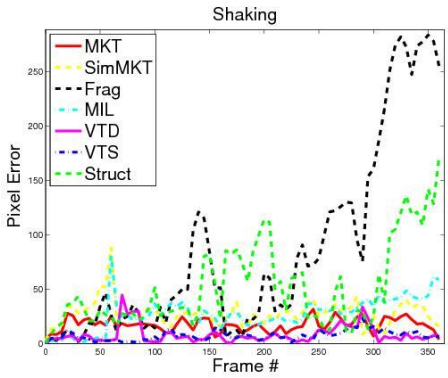


Figure 2. The Pixel Errors for Every Sequence Tested in the Experiments. This Figure is Best Viewed in Color

5. Conclusion

In this paper, a novel tracking algorithm - the Mean Kernel Tracker (MKT) - has been proposed. MKT constructs the appearance model by combining three complementary features - Color, HOG and LBP - to resist different appearance changes. Through experiments on public benchmark videos, our tracking algorithm can track objects very well under illumination change, heavy occlusion, background clutter and even backup from tracking failures, and the MKT can perform favorably against several state-of-the-art algorithms.

Acknowledgment

This work was supported by Beijing Higher Education Young Elite Teacher Project (NO. YETP1949) and National Natural Natural Science Foundation of China (Grant No. 30901164).

References

[1] M. Yang, Z. Fan, J. Fan, and Y. Wu, "Tracking nonstationary visual appearances by data-driven adaptation," IEEE TIP, vol. 18, **(2009)**, pp. 1633–1644.
[2] M. Tang and X. Peng, "Robust tracking with discriminative ranking lists", IEEE TIP, vol. 21, no. 7, **(2012)**, pp. 3273 –3281.
[3] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey", ACM Comput. Surv, vol. 38, **(2006)**.
[4] R. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features", IEEE PAMI, vol. 27, no. 10, **(2005)**, pp. 1631–1643.
[5] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," IJCV, vol. 56, **(2004)**, pp. 221–255.
[6] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns", IEEE PAMI, vol. 24, no. 7, **(2002)**, pp. 971–987.
[7] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "Simplemkl", JMLR, vol. 9, **(2008)**, pp. 2491–2521.
[8] L. Breiman, "Random forests", Machine learning, vol. 45, no. 1, **(2001)**, p. 5–32.
[9] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram", in CVPR, **(2006)**.
[10] H. Grabner and H. Bischof, "On-line boosting and vision", in CVPR, **(2006)**.
[11] Y. Wu and T. Huang, "A co-inference approach to robust visual tracking," in ICCV, **(2001)**.
[12] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking", in ECCV, **(2008)**.
[13] D. Chen, J. Zhang, and M. Tang, "Random patch based video tracking via boosting the relative spaces", in ICASSP, **(2009)**.
[14] J. Zhang, D. Chen, and M. Tang, "Combining discriminative and descriptive models for tracking", in ACCV, **(2009)**.
[15] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning", in CVPR, **(2009)**.

[16] Y. Bai and M. Tang, "Robust visual tracking via ranking svm", in ICIP, **(2011)**.
[17] J. Kwon and K. M. Lee, "Tracking by sampling trackers", in ICCV, **(2011)**.
[18] Y. Bai and M. Tang, "Robust tracking via weakly supervised ranking svm", in CVPR, **(2012)**.
[19] P. H. S. T. Sam Hare and A.Saffari, "Struck: Structured output tracking with kernels", in ICCV, **(2011)**.
[20] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-n learning: Bootstrapping binary classifiers by structural constraints", in CVPR, **(2010)**.
[21] J. Shi and C. Tomasi, "Good features to track", in CVPR, **(1994)**.
[22] M. Varma and D. Ray, "Learning the discriminative power-invariance trade-off", in ICCV, **(2007)**.
[23] P. Gehler and S. Nowozin, "On feature combination for multiclass object classification", in ICCV, **(2009)**.
[24] W. Du and J. Piater, "A probabilistic approach to integrating multiple cues in visual tracking", in ECCV, **(2008)**.
[25] J. Kwon and K. M. Lee, "Visual tracking decomposition", in CVPR, **(2010)**.
[26] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection", in CVPR, **(2005)**.
[27] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the smo algorithm", in ICML, **(2004)**.
[28] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman, "Multiple kernels for object detection", in ICCV, **(2009)**.
[29] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines", ACM IST, **(2011)**.

Authors

Lei Li was born in JiLin Province, China, in 1980. He studies in Technology Beijing Forestry University, and separately gained a bachelor's degree in 2005 and a master's degree in 2008. Now, he is a doctorate candidate in the same university. His research interests mainly include information security, image processing image compression.

Ruiting Zhang was born in HeBei Province, China, in 1981. He studies on Applied Mathematics in China Agriculture University, and gained a master's degree in 2006. Then he works in Canvard college, Beijing Technology and Business University. His research interests mainly include Machine Learning, Support vector machine.

Jiangming Kan, Corresponding author of this paper, received in PhD degree in forestry engineering from Beijing Forestry University, P.R.China in 2009. Currently, he is an associate professor in Beijing Forestry University. His research interests include computer vision and intelligent control.

Wenbin Li received M.S. and Ph.D. degrees in Shizuoka University and Ehime University, Japan, in 1987, and 1990, respectively. Starting 1992, he was a faculty with the School of Technology, Beijing Forestry University and was promoted to be a professor in 1996, Ph.D. supervisor. His current research interests include forest machinery automation and intelligent.

