Patch-based Scale Calculation for Real-time Visual Tracking

Yulong Xu, Jiabao Wang, Hang Li, Yang Li, Zhuang Miao, and Yafei Zhang

***Abstract*—Robust scale calculation is a challenging problem in visual tracking. Most existing trackers fail to handle large scale variations in complex videos. To address this issue, we propose a robust and efficient scale calculation method in tracking-by-detection framework, which divides the target into four patches and computes the scale factor by finding the maximum response position of each patch via color attributes kernelized correlation filter. In particular, we employ the weighting coefficients to remove the abnormal matching points and transform the desired training output of the conventional classifier to solve the location ambiguity problem. Experiments are performed on several challenging color sequences with scale variations in the recent benchmark evaluation. And the results show that our method outperforms state-of-the-art tracking methods while operating in real-time.**

***Index Terms*—Correlation filter, real-time visual tracking, scale calculation.**

I. Introduction

VISUAL tracking, where the objective is to estimate locations of a target in an image sequence, is one of the most important problems in the field of computer vision with applications ranging from surveillance and human-computer interactions to robotics and medical imaging [1], [3].

In recent years, tracking-by-detection methods [4]–[11] show excellent tracking performance. These approaches work by posing the task of target localization as a classification problem. The decision boundary is obtained by learning a discriminative classifier online using image patches from both the target and the background. Heriques *et al.* propose the circulant structure with kernels (CSK) by using correlation filters in a kernel space [7] which reaches a top speed in a recent benchmark [2]. The CSK method builds only on illumination intensity features and is further improved by using histogram of oriented gradients (HOG) features in the kernelized correlation filter (KCF) tracking algorithm [9]. In [8], Danelljan *et al.* extend the CSK tracker with color attributes or color names (CN) and propose an adaptive low-dimensional variant of color attributes.

Most tracking-by-detection methods [4]–[9] try to handle scale variations in complex videos under the hypothesis that the target scale is constant. This assumption, however, is not very reasonable in real life, as the distance between the moving object and the camera is often in a state of flux. On the other hand, several existing approaches [12]–[15] estimate scale variations but operate with a lower efficiency, which makes them limited in practical applications. An ideal scale calculation approach should be robust to scale variations while operating in real-time.

This letter proposes an efficient approach for robust scale calculation in visual tracking, the contributions of which are two-folds. First, in contrast to existing methods, we divide the target into four patches after localizing the object in the current frame and a new classifier is trained on every patch. The scale factor of two adjacent frames is computed by finding the maximum response position of each patch. In order to enhance the robustness in the process of scale calculation, we employ the weighting coefficients to remove the abnormal candidate points. Second, for the purpose of decreasing the location ambiguity problem in our tracker, we transform the desired training output of the conventional classifier to make more accurate positioning. We also compare our approach with 11 state-of-the-art trackers and the results show that our approach outperforms the other methods while running at more than 60 frames per second (FPS).

II. Patch-Based Scale Calculation

Recently, the correlation filter based trackers [6]–[10] are becoming increasingly popular due to their promising performance and computational efficiency. Among these trackers, the CN tracker [8] achieves very good performance with high speed (168 FPS). However, the CN tracker cannot handle the scale problem well. Once the scale of the target changes dramatically, the trackers may fail to relocate the objects. In this section, we develop our method based on the CN tracker to build a real-time patch-based tracking system which is robust to scale variations.

A. The CN Tracker

The CN tracker [8] employs a robust update scheme by considering all previous frames when computing the current model.

Manuscript received July 21, 2015; revised September 01, 2015; accepted September 07, 2015. Date of publication September 16, 2015; date of current version November 16, 2015. This work was supported by the National Natural Science Foundation of China under Grant 61402519, and by the Jiangsu Provincial Nature Science Foundation of China under Grants BK20140071 and BK2012512. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Edmund Lam.

The authors are with the College of Command Information Systems, PLA University of Science and Technology (PLAUST), Nanjing 210007, China (e-mail: whuxuyulong_88@126.com; jiabao_1108@163.com; lihang0003@hotmail.com; solarlee@126.com; emiao_beyond@163.com; miipl606@gmail.com).

(*Corresponding author: Z. Miao*.)

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LSP.2015.2479360

The key for its better performance, especially in tracking speed, is the kernel trick and the adaptive low-dimensional variant of color attributes. The CN tracker considers all extracted appearances $\mathbf{x}^j$, $j \in [1,t]$ and all their cyclic shifts $\mathbf{x}^j_{m,n}$, $(m,n) \in \{0,\cdots,M-1\} \times \{0,\cdots,N-1\}$ of the target from the first frame till the current frame t as the training examples for the classifier. They are labeled with a Gaussian function $\mathbf{y}^j(m,n)$. The cost function is constructed as the weighted average quadratic error over all previous frames and the solution is constrained to only contain one set of classifier coefficients α. Each frame j is weighted with a constant $\beta_j \geq 0$. The cost function is:

$$\varepsilon = \sum_{j=1}^{t} \beta_j \left(\sum_{m,n} \left| \left\langle \varphi(\mathbf{x}^j_{m,n}), \mathbf{w}^j \right\rangle - \mathbf{y}^j(m,n) \right|^2 + \lambda \left\| \mathbf{w}^j \right\|^2 \right), \tag{1}$$

where $\mathbf{w}^j = \sum_{m,n} \alpha(m,n) \varphi(\mathbf{x}^j_{m,n})$, φ represents the mapping to the Hilbert space induced by the kernel κ. The inner product of $\mathbf{x}^j_{m,n}$ and $\mathbf{x}^j_{k,l}$ is computed as $\langle \varphi(\mathbf{x}^j_{m,n}), \varphi(\mathbf{x}^j_{k,l}) \rangle = \kappa(\mathbf{x}^j_{m,n}, \mathbf{x}^j_{k,l})$. The constant $\lambda \geq 0$ is a regularization parameter.

The cost function is minimized by

$$A^t = \frac{\sum_{j=1}^{t} \beta_j Y^j K^j_x}{\sum_{j=1}^{t} \beta_j K^j_x \left(K^j_x + \lambda \right)}. \tag{2}$$

In (2), $Y^j = \mathsf{F}(\mathbf{y}^j)$ and $K^j_x = \mathsf{F}(\mathbf{k}^j_x)$, where F denotes the Fourier transform; $\mathbf{k}^j_x(m,n) = \kappa(\mathbf{x}^j_{m,n}, \mathbf{x}^j)$ is the output of the kernel function κ. The weights β_j are set by using a learning rate parameter η. The total model is updated using (3). The numerator A^t_N and denominator A^t_D of $A^t = A^t_N / A^t_D$ in (2) are updated separately; $\hat{\mathbf{x}}^t$ denotes the learned target appearance.

$$\begin{cases} A^t_N = (1-\eta) A^{t-1}_N + \eta Y^t K^t_x \\ A^t_D = (1-\eta) A^{t-1}_D + \eta K^t (K^t_x + \lambda) \\ \hat{\mathbf{x}}^t = (1-\eta) \hat{\mathbf{x}}^{t-1} + \eta \mathbf{x}^t \end{cases}. \tag{3}$$

In the tracking, a patch $\mathbf{z}$ with size $M \times N$ is cropped out in the new frame $t+1$. The confidence score is calculated as

$$\hat{\mathbf{y}}^{t+1} = \mathsf{F}^{-1} \left(\frac{A^t_N}{A^t_D} \odot K_z \right), \tag{4}$$

where F^{-1} denotes the inverse Fourier transform; $\odot$ is the element-wise product; $K_z = \mathsf{F}(\mathbf{k}_z)$, $\mathbf{k}_z(m,n) = \kappa(\mathbf{z}_{m,n}, \hat{\mathbf{x}}^t)$. The target position in the new frame is then estimated by finding the location with the highest score.

In order to decrease the computing time, the CN tracker applies an adaptive low-dimensional variant of color attributes. In CN tracker, let $\hat{\mathbf{x}}^t$ be the D_1-dimensional learned appearance. Then, the dimensionality reduction technique is employed to find a $D_1 \times D_2$ projection matrix $\mathbf{B}_t$ with orthonormal column vectors. This matrix $\mathbf{B}_t$ is used to compute the new D_2-dimensional feature map $\tilde{\mathbf{x}}^t$ of the appearance by the linear mapping $\tilde{\mathbf{x}}^t(m,n) = \mathbf{B}^T_t \hat{\mathbf{x}}^t(m,n)$.

B. Patch-based Scale Calculation via CN

In our approach, let the center position of the target in the frame $t-1$ be p_{t-1}, and the target scale be $w_{t-1} \times h_{t-1}$ pixels.

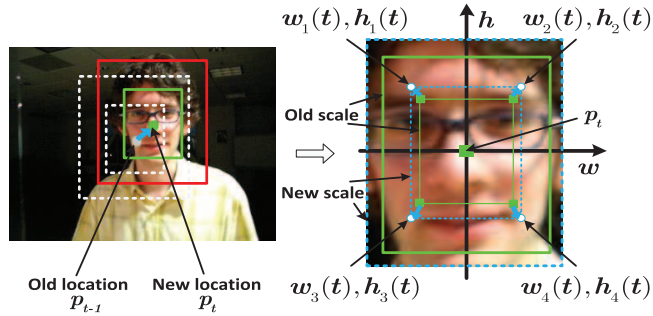

Fig. 1. Illustration of the matching points in the t-th frame.

At the $(t-1)$-th frame, we extract an image patch $\mathbf{x}^{t-1}$ with size $\rho w_{t-1} \times \rho h_{t-1}$ around the center p_{t-1}, as in [8]. Here ρ is the expansion coefficient. Unlike [8], we suggest to uniformly resize all patches with size $W \times H$, then the learned target appearance $\hat{\mathbf{x}}^{t-1}$ and the learned classifier coefficients A^{t-1} are updated using (3). Let the center position of the target p_{t-1} be $(0,0)$, then divide $\mathbf{x}^{t-1}$ into four patches and their central locations are $(w_1(t-1), h_1(t-1))$, $(w_2(t-1), h_2(t-1))$, $(w_3(t-1), h_3(t-1))$ and $(w_4(t-1), h_4(t-1))$.

A new classifier is trained on every patch using (1), so there are four different classifiers in total. In the four patch trackers, the model is updated by (3) and the dimensionality reduction technique is also adopted as described in [8].

In a new frame at time t, we calculate scale factor after localizing the object. Firstly, we extract a $\rho w_{t-1} \times \rho h_{t-1}$ candidate image patch $\mathbf{z}_{t0}$ around the center p_{t-1} and calculate the confidence score by (4). The target position p_t in the t-th frame is estimated by finding the location with the highest score. Next, we extract a $w_{t-1} \times h_{t-1}$ candidate image patch $\mathbf{z}_{t1}$ around the center p_t and set up coordinate system centered by p_t, as shown in Fig. 1. Then, the confidence score is calculated using (4) in every patch and the four matching points with the highest score are $(w_1(t), h_1(t))$, $(w_2(t), h_2(t))$, $(w_3(t), h_3(t))$ and $(w_4(t), h_4(t))$.

The scale factor γ_t is calculated as

$$\gamma_t = \sqrt{\left(\frac{\sum_{j=1}^{4} \delta_j |w_j(t)|}{\sum_{i=1}^{4} \delta_i |w_i(t-1)|} \right) \cdot \left(\frac{\sum_{j=1}^{4} \delta_j |h_j(t)|}{\sum_{i=1}^{4} \delta_i |h_i(t-1)|} \right)}, \tag{5}$$

where the weights δ_i can make the results more robust by removing the abnormal matching points, and they are set by

$$\delta_i = \begin{cases} 1, & \begin{cases} \tau_1 < |w_i(t)| < \tau_2 \\ \tau_3 < |h_i(t)| < \tau_4 \\ ||w_i(t)| - |h_i(t)|| < \tau_5 \end{cases} \\ 0, & otherwise. \end{cases}. \tag{6}$$

Here $\tau_1 \sim \tau_5$ are thresholds, which are empirical parameters.

The scale $w_t \times h_t$ in the t-th frame is calculated by

$$\begin{cases} w_t = \gamma_t w_{t-1} = w_1 \prod_{i=2}^{t} \gamma_i \\ h_t = \gamma_t h_{t-1} = h_1 \prod_{i=2}^{t} \gamma_i \end{cases}. \tag{7}$$

Finally, in the t-th frame, we extract an image patch $\mathbf{x}^t$ with size $\rho w_t \times \rho h_t$ around the center p_t to update the appearance and coefficients using (3).

C. Desired Training Output

During training, we must attach a label, also called the desired training output, to each sample. In CN tracker, it is a Gaussian function

$$\mathbf{y}(m,n) = \exp\left(-\frac{|p-p_0|^2}{2\sigma^2}\right), \tag{8}$$

where σ is spatial bandwidth proportional to target size, $p_0 = (m_0, n_0)$ is target location, $p = (m,n)$ and $|p-p_0| = \sqrt{(m-m_0)^2 + (n-n_0)^2}$.

At the point $p_0 = (m_0, n_0)$,

$$\begin{cases} \mathbf{y}\big|_{p=p_0} = \max(\mathbf{y}) = 1 \\ \frac{\partial \mathbf{y}}{\partial m}\Big|_{p=p_0} = \frac{\partial \mathbf{y}}{\partial n}\Big|_{p=p_0} = 0. \end{cases} \tag{9}$$

According to (9), the partial derivatives of Gaussian function equal to zero at the maximum point, which means that the responses are too close to be distinguished. And in CN tracker, the target position is estimated by finding the location with the highest score, so the Gaussian function in (8) will bring the location ambiguity problem, especially in scale calculation. And this problem is verified in our experiments in Section III-B. To solve this problem, we replace the desired training output of a target location using (10).

$$\tilde{\mathbf{y}}(m,n) = \exp\left(-\frac{|p-p_0|}{2\theta}\right), \tag{10}$$

where $0 < \theta < 1$ is a constant.

The partial derivatives are calculated as

$$\begin{cases} \frac{\partial \tilde{\mathbf{y}}}{\partial m} = -\frac{m-m_0}{2\theta|p-p_0|}\exp\left(-\frac{|p-p_0|}{2\theta}\right) \\ \frac{\partial \tilde{\mathbf{y}}}{\partial n} = -\frac{n-n_0}{2\theta|p-p_0|}\exp\left(-\frac{|p-p_0|}{2\theta}\right) \end{cases}. \tag{11}$$

In particular, at the point $p_0 = (m_0, n_0)$, the value and the partial derivatives of the function are

$$\begin{cases} \tilde{\mathbf{y}}\big|_{p=p_0} = \max(\tilde{\mathbf{y}}) = 1 \\ \frac{\partial \tilde{\mathbf{y}}}{\partial m}\Big|_{\substack{m=m_0^+ \\ n=n_0}} = -\frac{1}{2\theta},\ \frac{\partial \tilde{\mathbf{y}}}{\partial m}\Big|_{\substack{m=m_0^- \\ n=n_0}} = \frac{1}{2\theta} \\ \frac{\partial \tilde{\mathbf{y}}}{\partial n}\Big|_{\substack{n=n_0^+ \\ m=m_0}} = -\frac{1}{2\theta},\ \frac{\partial \tilde{\mathbf{y}}}{\partial n}\Big|_{\substack{n=n_0^- \\ m=m_0}} = \frac{1}{2\theta} \end{cases}. \tag{12}$$

Here $m = m_0^+$ and $n = n_0^+$ denote right derivative; $m = m_0^-$ and $n = n_0^-$ denote left derivative.

In (12), the right derivative and left derivative are inversely proportional to θ, which means that the responses can be distinguished easily. And this is beneficial to locating target as well as decreasing the location ambiguity in tracking.

III. Experiments

We first show that the performance is significantly improved by employing weights δ_i in (5) and using (10) in lieu of Gaussian shape of the desired output. And then we compare our scale calculation approach with state-of-the-art trackers.

A. Experimental Setup

The approach proposed in this letter is implemented in Matlab 2013a. We perform the experiments on an Intel i5-4690 CPU (3.50 GHz) with 16 GB RAM and employ 10 challenging sequences with scale variations in [2]. The expansion coefficient is set to $\rho = 2$, the expansion patch $W \times H$ is resized to 128×128, the target is 64×64 and the four patches are 32×32. We use the adaptive low-dimensional color attributes $D_2 = 3$ instead of $D_2 = 2$ in [8]. The optimal parameter values $\theta = 0.56$ for the expansion patch and $\theta_{patch} = 0.6$ for the four patches are obtained by repeated experiments at the step of 0.01 in the interval $[0.01, 1]$. The five thresholds $\tau_1 \sim \tau_5$ are empirical parameters, which are set as $\tau_1 = \lfloor 0.85 * \frac{W}{4\rho} \rfloor = 13$, $\tau_2 = \lceil 1.15 * \frac{W}{4\rho} \rceil = 19$, $\tau_3 = \lfloor 0.85 * \frac{H}{4\rho} \rfloor = 13$, $\tau_4 = \lceil 1.15 * \frac{H}{4\rho} \rceil = 19$ and $\tau_5 = \lceil 0.1 * \frac{\sqrt{WH}}{4\rho} \rceil = 2$. Here $\lfloor * \rfloor$ denotes rounding toward negative infinity and $\lceil * \rceil$ denotes rounding toward positive infinity. The parameters $\tau_1 \sim \tau_4$ set a limit to the size of the scale factor, which is controlled in the range of 0.85 and 1.15, and τ_5 ensures that the width and height of the target enlarge or reduce simultaneously. The other parameter values are the same as conventional CN tracker.

TABLE I
Comparison of Our Approach with the Baseline

Evaluation	Ours	Our#	Our*	CN
Average CLE (pixels)	**9.2**	21.5	37.9	86.3
Average DP (%)	**91.4**	74.2	75.3	74.9
Average OP (%)	**85.2**	50.7	59.8	44.8
Average speed (FPS)	64	64.4	64.6	**168**

We present the results using center location error (CLE), distance precision (DP) and overlap precision (OP). CLE is the average Euclidean distance between the estimated center location of the target and the ground-truth. DP is the percentage of frames where the CLE is smaller than a threshold. We report DP values at a threshold of 20 pixels. OP is defined as the percentage of frames where the bounding box overlap exceeds a threshold $\xi \in [0,1]$, and the threshold of 0.5 is set here, as in [17].

B. Robust Scale Calculation

Table I shows the results of our scale calculation methods proposed in Section II. We use the conventional CN tracker without scale calculation as a baseline. $*$ denotes our scale calculation method with (6) and (8); # denotes scale calculation method with (10) but without (6), which means $\delta_i = 1$ for all patches in (5). In our$*$ tracker, by using the weights δ_i to remove the abnormal matching points, our method outperforms baseline by 15% in average OP. Similarly, in our# tracker, the performance is improved with a reduction in average CLE by 64.8 pixels, which proves that replacing the Gaussian shape of the desired output with (10) can decrease the location ambiguity. Compared with the baseline CN tracker, our approach with (6) and (10) significantly improves the average OP from 44.8% to 85.2%, outperforms CN by 16.5% in average DP and enhances the performance with a reduction in the average CLE by 77.1 pixels. These clearly demonstrate the effectiveness and the robustness of our scale calculation in object tracking. However, this performance gain is achieved at the cost of a higher computational load. Comparing our approach with the baseline CN tracker, we find that the average speed of our tracker reduces from 168 FPS to 64 FPS.

TABLE II
COMPARISON OF OUR APPROACH WITH 11 STATE-OF-THE-ART TRACKERS

Evaluation	Ours	DSST	TLD	TGPR	KCF	L1APG	DSS	CN	PCOM	LSST	Struck	STC
Average CLE (pixels)	**9.2**	44.7	**31.4**	52	**30.6**	49.5	46.1	86.3	64	62.9	55.3	57.3
Average DP (%)	**91.4**	**81.1**	67.3	71.7	**85.2**	63.7	62.9	74.9	61.8	51.8	55.4	63.8
Average OP (%)	**85.2**	**71.4**	**62**	60.3	57.9	50.4	45.9	44.8	44.3	40.2	39.3	39.1
Average speed (FPS)	64	40	15.9	0.7	**256**	16	2.1	**168**	29.2	6.5	10.8	**164**

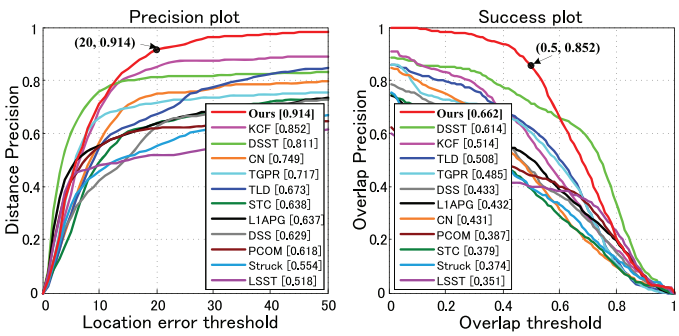

Fig. 2. Precision and success plots. The legend of the precision plot reports the average DP score at 20 pixels for each method and the legend of the success plot contains the area-under-the-curve (AUC) score for each tracker.

C. Comparison with State-of-the-art

We compare our approach with 11 state-of-the-art trackers: CN [8], DSS [15], DSST [18], KCF [9], L1APG [13], LSST [14], PCOM [16], STC [10], Struck [4], TGPR [19] and TLD [12], which have shown to provide excellent performance in the literature. The comparison on the 10 benchmark sequences is shown in Table II. We present the results using average CLE, DP and OP over all sequences. Moreover, a speed comparison in average FPS is also provided. The best three results are highlighted in **bold**.

Among the existing trackers in our evaluation, KCF provides the best results with an average CLE of 30.6 pixels. Our approach improves the performance with a reduction in average CLE by 21.4 pixels. Simultaneously, KCF also provides the best results among the existing methods with an average DP of 85.2% while our approach exceeds it by 6.2% in average DP. Finally, in overlap precision, DSST provides the best results among the existing methods with an average OP of 71.4%. Our approach transcends the tracking performance by achieving an average OP of 85.2%. In addition, our method is significantly faster than the best three performing compared trackers in descending order of OP, which is 24 FPS faster than DSST, more than 4 times faster than TLD and more than 90 times faster than TGPR in average FPS.

Fig. 2 contains the precision and success plots illustrating the average DP and OP over all the 10 sequences. In both precision and success plots, the proposed method outperforms the best existing method (KCF and DSST). In summary, the precision plot shows that our method is more robust than state-of-the-art trackers, and the success plot demonstrates that our approach computes scale more accurately on the benchmark sequences.

Our approach also has a certain robustness to handle other challenging problems. Fig. 3 shows the 10 benchmark sequences with scale variations. These sequences also pose challenging problems such as occlusion (Fig. 3(a), (e), (g) and (j)), fast motion (Fig. 3(b)), rotation (Fig. 3(c), (d), (e) and (f)), illumination variations (Fig. 3(h) and (i)) and background clutter (Fig. 3(a), (h) and (i)). Despite these challenges, our tracker accurately calculates both the scale and position of the target.

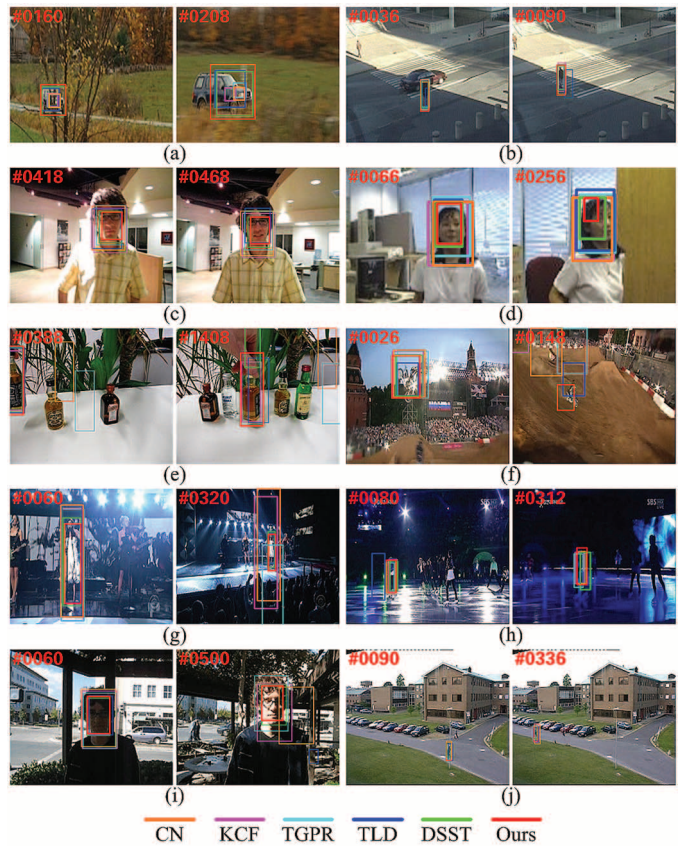

Fig. 3. A visualization of the tracking results of our approach and the state-of-the-art visual trackers CN, KCF, TGPR, TLD and DSST on 10 benchmark sequences.

IV. CONCLUSION

In this letter, we propose a scale calculation approach for visual object tracking. Our method works by dividing the target into four patches and matching the center location of each patch via color attributes kernelized correlation filter. With this method, the scale computation is transformed into locating the centers of the patches. Experiments are performed on several challenging benchmark sequences with significant scale variations. The results clearly demonstrate that our approach outperforms state-of-the-art methods, while operating in real-time.

REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, pp. 1–45, 2006.

[2] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *CVPR*, 2013, pp. 2411–2418.

[3] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1442–1468, 2014.

[4] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *ICCV*, 2011, pp. 263–270.

[5] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *ECCV*, 2012, pp. 864–877.

[6] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *CVPR*, 2010, pp. 2544–2550.

[7] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *ECCV*, 2012, pp. 702–715.

[8] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer, "Adaptive color attributes for real-time visual tracking," in *CVPR*, 2014, pp. 1090–1097.

[9] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, 2015.

[10] K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M.-H. Yang, "Fast visual tracking via dense spatio-temporal context learning," in *ECCV*, 2014, pp. 127–141.

[11] D. Wang, H. Lu, and C. Bo, "Online visual tracking via two view sparse representation," *IEEE Signal Process. Lett.*, vol. 21, no. 9, pp. 1031–1034, 2014.

[12] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, 2012.

[13] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust L1 tracker using accelerated proximal gradient approach," in *CVPR*, 2012, pp. 1830–1837.

[14] D. Wang, H. Lu, and M.-H. Yang, "Least soft-threshold squares tracking," in *CVPR*, 2013, pp. 2371–2378.

[15] B. Zhuang, H. Lu, Z. Xiao, and D. Wang, "Visual tracking via discriminative sparse similarity map," *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1872–1881, 2014.

[16] D. Wang and H. Lu, "Visual tracking via probability continuous outlier model," in *CVPR*, 2014, pp. 3478–3485.

[17] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.

[18] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *BMVC*, 2014, pp. 1–5.

[19] J. Gao, H. Ling, W. Hu, and J. Xing, "Transfer learning based visual tracking with Gaussian processes regression," in *ECCV*, 2014, pp. 188–203.

This paper previously published in IEEE Signal Processing Letters