Robust Visual Tracking Using Dynamic Feature Weighting Based on Multiple Dictionary Learning

Renfei Liu[1, 2], Xiangyuan Lan[1], Pong C Yuen[1, *], G C Feng[2]
[1]Department of Computer Science, Hong Kong Baptist University, Hong Kong
[2]School of Mathematics and Computational Science, Sun Yat-sen University, P.R. China
{renfeiliu, xylan, pcyuen}@comp.hkbu.edu.hk, mcsfgc@mail.sysu.edu.cn

***Abstract*-Using multiple features in appearance modeling has shown to be effective for visual tracking. In this paper, we dynamically measured the importance of different features and proposed a robust tracker with the weighted features. By doing this, the dictionaries are improved in both reconstructive and discriminative way. We extracted multiple features of the target, and obtained multiple sparse representations, which plays an essential role in the classification issue. After learning independent dictionaries for each feature, we then implement weights to each feature dynamically, with which we select the best candidate by a weighted joint decision measure. Experiments have shown that our method outperforms several recently proposed trackers.**

Keywords-visual tracking, feature weighting, sparse coding, dictionary learning.

I. INTRODUCTION

Challenges in visual tracking, such as occlusions, cluttered backgrounds, and pose variance have always limited the performance of trackers. Plenty of methods have been proposed to deal with various situations aiming at appearance modeling. And recently, more and more methods are using sparse representation due to its capability of capturing the most essential information from training sets and its robustness to noise [1, 2, 3, 7, 8, 9, 10]. Due to these advantages, methods using multiple features or multiple dictionaries to sparsely represent the target have been proposed [4, 5]. And in [4], using multiple dictionaries enhances the method's robustness against noise since it can more promisingly maintain the essential information. However, existing approaches using multiple features have drawbacks. First, previous methods treated each feature fairly and ignored the fact that different features have different importance [5]. Feature weighting is a crucial problem since if we treat features with different importance as the same, the target would not be represented well and thus the performance of the approach would be limited. Second, existing methods treated relationships between features statically. Situations in different frames can be significantly various, thus the weights of features should be dynamically updated to avoid noise pollution.

In this work, we focus on measuring the importance of different features dynamically in multi-feature dictionary learning. Inspired by the online discriminative dictionary learning [2], we develop a novel feature weighting scheme in multiple dictionary learning (FWMDL) for visual tracking which exploits both the reconstructive and discriminative capability of dictionaries. Inspired by [16], to strengthen the capability of the representation of the learned dictionaries, a specific class label is associated with each dictionary item instead of using only the reconstruction error as in [1, 7, 9, 10], which means that the algorithm learns sparse dictionaries and linear classifiers at the same time. To select the target from the candidates, we linearly combine the weighted joint decision measure, which is based on a combination of a quadratic appearance distance and classification error. Motivated by [2], to deal with appearance changes, we update the dictionaries and classifiers dynamically with the new samples.

Our contributions are as follows:

First, we propose a novel dictionary learning algorithm in which multiple dictionaries are optimally learned and updated for the sparse representation of the tracked targets with multiple features. By simultaneously learning multiple dictionaries with multiple features, the proposed tracker achieves more accurate representation and lower tracking error.

Second, we introduce a novel feature weighting scheme based on multiple dictionary learning for feature fusion. Dynamically weighting different features enables different features to play different role in appearance modeling based on their capabilities of describing the tracked target.

Third, the proposed method achieves superior performance and outperforms several state-of-the-art algorithms.

II. PROPOSED METHOD

A. Tracking Algorithm

Our goal is to model the tracked target accurately and robustly with a set of training samples given $X = \{X_1, X_2, \dots, X_N\} \in \mathbb{R}^{d \times N \times K}$. To construct template sets with multiple features, K features are extracted from the samples and there are in total N background and target templates in each feature. We assign each template with a class label from $Y = \{1, -1\}$. Each X_n is a matrix of $d \times K$ dimension, extracted from an image region corresponding to a positive sample(target) or negative sample(background). In our case, for the K columns of each X_n, we learn K dictionaries separately, and each dictionary is corresponding to one feature. In the situation of sparse coding, inspired by [1], each x_n^k is

*Corresponding author

978-0-9928-6265-7/16/$31.00 ©2016 IEEE

sparsely represented by its relative dictionary, $x_n^k \approx D_k c_n^k$, where c_n^k is the sparse code of x_n^k and is defined by:

$$c_n^k = arg\ \min_c \|x_n^k - D_k c\|^2 + \lambda_1 \|c\|_1 \quad (1)$$

where the parameter λ_1 balances the sparsity and the reconstruction error. In this case, the dictionaries should be not only generative, but discriminative as well. Therefore, motivated by [2], the sparse code, c_n^k, is used as the feature descriptor thus could be used to learn the dictionary and the classifier:

$$\min_{D_k, W_k} \sum_n l(y_n^k, f(c_n^k, W_k)) + \lambda_2 \|W_k\|_F^2 \quad (2)$$

where l is the loss function, λ_2 serves as a trade-off parameter, and y_n^k is the label vector for x_n^k, of which the non-zero position suggests the class label of x_n^k. f here is a linear classifier with classification parameters W_k represented by:

$$f(c_n^k, W_k) = W_k c_n^k \quad (3)$$

We aim to make the learned dictionaries good for classification, and samples from the m^{th} class will be represented by the dictionary items from the m^{th} class. Moreover, we take into account the sparse coding error and a linear regression loss for the equation so that the l above can be formulated by:

$$l(D_k, W_k; x_n^k, y_n^k, l_n^k) = (1-\mu)\|y_n^k - W_k c_n^k\|_2^2 + \mu\|l_n^k - c_n^k\|_2^2 \quad (4)$$

where $l_n^k = [1, \dots, 0, 1, 0, \dots, 0]$ is an ideal sparse code for x_n^k.

Thus, the learning framework is as follow:

$$\min_{D,W} \sum_{k=1}^{K} \sum_{n=1}^{N} (1-\mu)\|y_n^k - W_k c_n^k\|_2^2 + \mu\|l_n^k - c_n^k\|_2^2 + \lambda_2 \|W_k\|_F^2$$
$$s.t.\ c_n^k = arg\ \min_c \|x_n^k - D_k c\|^2 + \lambda_1 \|c\|_1 \quad (5)$$

where the variable μ controls the contribution of the sparse code error and the linear regression error.

The learned dictionaries are both reconstructive and discriminative due to the label information and the classification error during the optimization.

After the dictionaries are learned, we are able to classify the test samples. However, the way to use the dictionaries should be connected to the different characteristics of each feature. That is to say, the dictionaries should be combined by an approach that would take into account their respective importance. In our method, we use the capability of describing the target of each dictionary to measure the importance, and use a weighted joint decision measure to score each sample. The formulation is as follow:

$$\varphi_k(x) = \left\|x_{tr}^k - D_k c_n^k\right\|^2 + \omega_k \|y_n^k - W_k c_n^k\|^2 \quad (6)$$

where x_{tr}^k is the weighted average of the k^{th} feature of the tracking results. We add the weight in the classification error because the importance of the classification error should not be static, and should be updated dynamically. If the capability of describing the target of a feature is good enough, then the weight of the classification error should be big and vice versa.

In order to obtain the reconstruction error $\left\|x_{tr}^k - D_k c_n^k\right\|^2$, we accumulate each feature extracted from the bounding box separately at the optimal location into a set T_k. We add the optimal locations computed by the tracking algorithm to T_k and delete those from older frames from T_k, and T_k has fixed number of elements, denoted as U. We compute the x_{tr}^k above by weighted average of the elements in T_k, where the weight of each element in T_k can be computed by $e^{\mathcal{E}}$, where $\mathcal{E}$ is the joint decision error. While initialization, the weight of T_k is set to 1, and it only has one element-the ground truth.

Combining all the dictionaries, we add the weighted decision scores together:

$$\phi(s) = \sum_{k=1}^{K} \varphi_k(s) \quad (7)$$

The weight of each joint decision error in the K features is measured by the capability of describing the target. Thus, in our method, we compute the weight using the quadratic appearance distance of each dictionary. Thus the formulation is defined as:

$$\omega_k = \rho\left(1 - \frac{\left\|x_{tr}^k - D_k c_n^k\right\|^2}{\sum_{k=1}^{K} \left\|x_{tr}^k - D_k c_n^k\right\|^2}\right) \times (K-1)^{-1} \quad (8)$$

where ρ is a constant and controls the contribution of the weight.

Taking the weight of each dictionary into account, the tracker is more adaptive to different situations and challenges, and thus the robustness is enhanced.

B. Optimization

Although the dictionaries are learned separately, the learning procedures for each dictionary are the same. Thus we only consider one global optimization procedure. The objective function is as follow:

$$\Gamma_k = \sum_{n=1}^{N} (1-\mu)\|y_n^k - W_k c_n^k\|_2^2 + \mu\|l_n^k - c_n^k\|_2^2 + \lambda_2 \|W_k\|_F^2$$
$$s.t.\ c_n^k = arg\ \min_c \|x_n^k - D_k c\|^2 + \lambda_1 \|c\|_1 \quad (9)$$

However, the function above is nonlinear and nonconvex, thus we adopt the stochastic gradient descent in [3]. We compute the gradient with respect to W_k by:

$$\frac{\partial \Gamma_k}{\partial W_k} = (1-\mu)(W_k c_n^k - y_n^k)(c_n^k)^T + \lambda_2 W_k \quad (10)$$

However, the dictionary D_k is implicitly defined on the sparse code c_n^k instead of explicitly defined in Γ_k. In order to obtain the gradient with respect to D_k, we use the implicit differentiation algorithm on the fixed point equations as in [3, 15]. Therefore, we obtain the chain rule:

$$\frac{\partial \Gamma_k}{\partial D_k} = \frac{\partial \Gamma_k}{\partial c_n^k} \frac{\partial c_n^k}{\partial D_k} \quad (11)$$

where $\frac{\partial \Gamma_k}{\partial c_n^k} = (1-\mu)(W_k)^T(W_k c_n^k - y_n^k) + \mu(c_n^k - l_n^k)$. In order to obtain $\frac{\partial c_n^k}{\partial D_k}$, we define the fixed point equation $(D_k)^T(D_k c_n^k - x_k) = -\lambda sign(c_n^k)$ where we apply the sign function to the elements of c_n^k individually. Afterwards, we

calculate $\frac{\partial c_{n_\Delta}^k}{\partial W_{k_\Delta}} = \left(D_{k_\Delta}{}^T D_{k_\Delta}\right)^{-1}\left(\frac{\partial D_{k_\Delta}{}^T x_k}{\partial D_{k_\Delta}} - \frac{\partial D_{k_\Delta}{}^T D_{k_\Delta}}{\partial D_{k_\Delta}} c_n^k\right)$, where Δ and $\bar{\Delta}$ are the indices of all non-zero and zeros values. Then we have:

$$\frac{\partial \Gamma_k}{\partial D_k} = -D_k \xi_k (c_n^k)^T + (x_n^k - D_k c_n^k)(\xi_k)^T \qquad (12)$$

where $\xi_k \in \mathbb{R}^L$ and $\xi_{k_{\bar{\Delta}}} = 0$, $\xi_{k_\Delta} = \frac{\partial \Gamma_k}{\partial c_n^k}\left(D_{k_\Delta}{}^T D_{k_\Delta}\right)^{-1}$.

After doing this, we obtained the gradients with respect to W_k and D_k. We use the learning rate $min(\eta, \eta i_0/i)$ adopted in [3], where η is a constant, and $i_0 = M/10$, where M is the iteration number. More details are in Algorithm 1.

Algorithm1

Input: Training set $X = \{X_1, X_2, \dots, X_N\} \in \mathbb{R}^{d \times N \times K}$ with labels Y, λ_1, λ_2, μ, η, W^1, D^1

for m = 1 **to** M

for k = 1 to K

Obtain training sample X;

for n = 1 **to** N

Derive y_n^k from x_n^k;

Calculate sparse code c_n^k according to Equation (1);

Calculate the active set Δ_n and compute ξ_k through Equation (12);

Get the learning rate $min(\eta, \eta i_0/i)$;

Compute the gradients with respect to W_k^m and D_k^m through Equation (10) and (12);

Update W_k^m and D_k^m using

$W_k^m = W_k^m - \eta_m \frac{\partial \Gamma_k^n}{\partial D_k^m}$ and

$D_k^m = D_k^m - \eta_m \frac{\partial \Gamma_k^n}{\partial W_k^m}$

end for

$W_k^{m+1} = W_k^m$, and $D_k^{m+1} = D_k^m$;

end for

$W^{m+1} = \{W_k^{m+1}\}_{k=1}^K$, and $D^{m+1} = \{D_k^{m+1}\}_{k=1}^K$;

end for

Output: New W and D;

C. Initialization

In the first frame, we use the K-SVD [6] on samples we obtained, both negative and positive, and they share the same size. And we combine the sampled items to form the initialized dictionaries $D^0 = \{D_1^0, D_2^0, \dots, D_K^0\}$. Then we calculate the sparse code for x_n^k to form the matrix C_k, which contains all the sparse codes of samples of the k^{th} feature. And we apply the ridge regression model to initialize $W^0 = \{W_1^0, W_2^0, \dots, W_K^0\}$: $W_k^0 = arg\ \min_{W_k^0} \|Y^k - W_k^0 C_k\|^2 + \lambda_3 \|W_k^0\|^2$, where Y^k is the label matrix for the samples of k^{th} feature. And this equation can be computed by: $W_k^0 = Y^k (C_k)^T (C_k (C_k)^T + \lambda_3 I)^{-1}$, where I represents the identity matrix.

III. Implementation Details

A. Tracking Procedure

We are given a ground-truth bounding box $b^1 = (x^1, y^1, s^1)$ in the first frame, where the (x^1, y^1) is the central point and s^1 is the scale. To obtain items in the initial dictionaries, we randomly shift the bounding box near and far away from the central point to get the positive and the negative samples, without overlap. Afterwards, we initialize the D^0 and W^0 as discussed above. The tracking result is obtained by referring the location of target in current frame via previous information. We sample around the previous frame b^{t-1} through a Gaussian distribution $p(b^t | b^{t-1})$, and among all the candidates, we use the weighted joint decision measure mentioned above to select the target.

B. Update

We update the dictionary and the classifier periodically. In each frame, we sample the positive items near the bounding box, and negative ones far away from the bounding box. We control the distance from the decided next bounding box to make sure that most negative samples are pure background information so that they differentiate the target to the most extent. We add these samples to a set, S, and if the size of S reaches a threshold V, we update the dictionaries and empty S.

While the elements in set T_k *and* S are accumulated, the result of the tracker could contain significant noise so that it is not reliable if the reconstruction error or the classification error is bigger than a pre-defined threshold. In this case, we will skip this frame to avoid updating with potentially noisy sample. See Algorithm 2 for details.

Algorithm2

Input: Frame sequences $I_1, I_2, \dots, I_t$.

Initialization I_1

Given the ground-truth $b^1 = (x^1, y^1, s^1)$, N^+ positive samples and N^- negative samples are obtained;

Extract multiple features from samples with label information Y;

Initialize D^0 and W^0;

Add all the K features into set S, and add the initial state b^1 into set $\{T_k\}_{k=1}^K$;

Tracking Procedure:

Sample Z candidates according to Gaussian distribution around the target in the last frame;

Extract K features from each candidate and calculate the sparse codes;

For each candidate, use the proposed weighting

method to compute joint decision measure error;

Select the candidate with the smallest joint decision measure error ϕ as the target result b^t;

Sample N^+ positive samples and N^- negative samples near and far away from the result, and obtain new samples X_{new}.

Add K features of tracking result b^t into $\{T_k\}_{k=1}^K$ and add X_{new} into set S;

Remove the oldest items from T_k if the size of T_k is bigger than U;

If the size of set S is equal to V, update D and W, and then empty S.

Output: Tracking results for each frame sequence $b^1, b^2, \ldots, b^t$.

IV. EXPERIMENTS

A. Experiment Setting

We use two features, LBP [12] feature and HOG [11] feature, to generate the dictionaries, thus the K in the experiments is set to 2. The proposed tracker is evaluated in ten videos which cover different kinds of challenging factors, such as illumination, fast motion, scale changes, cluttered background, and occlusion. To further demonstrate its effectiveness, other five state-of-the-art tracking algorithms are also run for comparison including the L1 method [1], the MTT method [7], the DFT method [13], the IVT method [14], and the ODDL method [2]. The source codes of these compared methods are provided by the authors.

The widely accepted metric, center location error is adopted for evaluation. The center location error is defined as the Euclidean distance between centers of the bounding box and the ground truth. For fair comparison, all trackers are set to be with the same initialization parameters.

B. Results

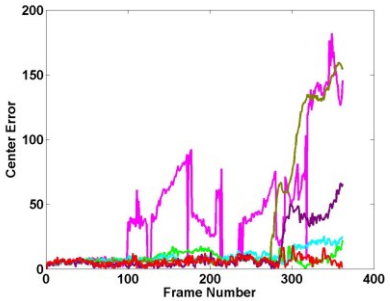
Without considering the time for dictionary learning, the time used for object tracking for each frame is about *0.39* second. Table I records the video-by-video comparison results in term of center location error. The best three results are shown in red, green and blue. We can see that the proposed tracker outperforms other sparse representation-based trackers and achieves the least error in overall comparison, which shows that the feature fusion scheme facilitate the tracking performance. Moreover, by using the proposed feature weighting scheme, the proposed tracker achieve a better performance than ODDL.

Figure 1 demonstrates frame-by-frame comparison in terms of center location error. We can see that even under large illumination (e.g. Car4, Shaking), fast motion (e.g. Jumping), occlusion (e.g. Faceocc, Faceocc2), or cluttered background (e.g. Football), the proposed tracker achieves a much more stable performance and maintain lower tracking error, which illustrates the effectiveness of the proposed feature weighting scheme to appearance variations and background change.

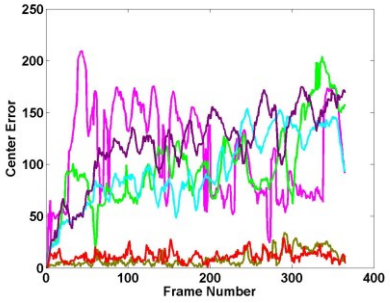
Figure 2 shows the tracking result we get from ten challenging sequences comparing with other five state-of-the-art algorithms. We can see that our method performs stable and robust in various situations since it never lost the target in all sequences.

TABLE I. VIDEO-BY-VIDEO COMPARISON IN TERMS OF CENTER LOCATION ERROR (MEASURED IN PIXELS)

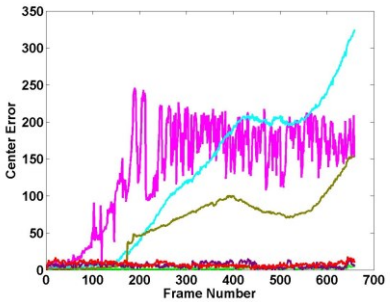
Sequence	L1	IVT	MTT	DFT	ODDL	OURS
Car4	137.8	1.9	126.6	61.9	6.2	7.7
Coke11	51.7	57.4	13.9	32.2	11.5	5.9
Faceocc2	14.4	7.2	13.2	10	11.8	15.2
Football	45	8.3	9.5	30.8	12.6	5.9
Girl	113.8	62	26.2	35.2	15.7	17.1
Jumping	4.2	4.3	85.8	60.5	5.6	5.5
Shaking	110.9	97.7	94.4	8.9	118.9	11.2
Singer1	155.4	12.2	12.1	26.7	39.6	26.9
Walking	8.7	3.7	5.9	282.8	9	7.9
Faceocc	18.8	19.7	22.6	13.6	24.6	24.8
Average	66.07	**27.44**	41.02	56.26	**25.55**	**12.81**



(a) Football



(b)Shaking



(c)Car4

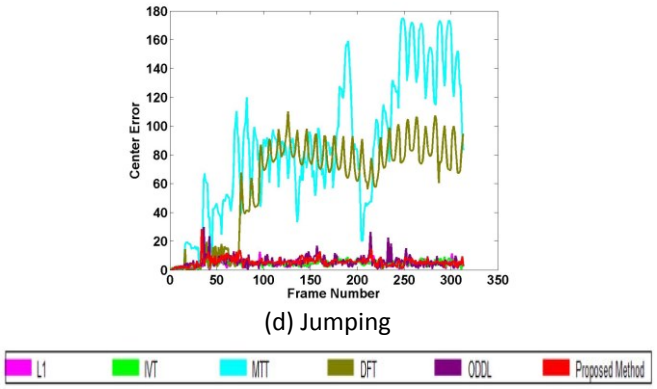


Fig.1 Frame-by-frame comparison in terms of Center Location Error (Measured in pixels).

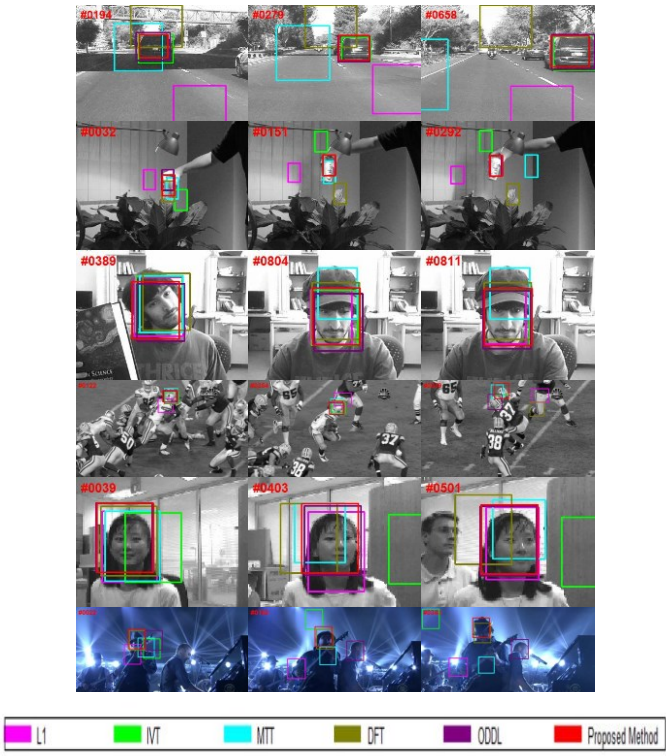


Fig.2 Some of the tracking results of the proposed method and the compared methods on ten challenging sequences (from up to down are *Car4*, *Coke11*, *Faceocc2*, *Football*, *Girl*, and *Shaking*).

V. CONCLUSION

In this paper, we developed a novel feature weighting scheme in multiple dictionary learning and proposed a robust visual tracker. We measured the contribution of each feature in each frame to adaptively and progressively update the weight of the dictionaries. We extract multiple features from each sample and then calculate the sparse codes for all the extracted features. By using the sparse codes we obtained, we can learn dictionaries and classifiers more effectively. And after the dictionaries are learned, we weight the classification error of each feature basing on the capability of describing the target of each feature. By using the weighted classification error, we combine it with the quadratic appearance distance to serve as a joint decision measure error, and select the target from the candidates according to the error. The weighting procedure takes into account the characteristics of each feature at different time, thus enhanced the adaptiveness of the dictionaries, and has improved the tracking performance in both reconstructive and discriminative way. Experiment results have shown that the proposed method achieves superior performance comparing to state-of-the-art methods.

VI. ACKNOWLEDGEMENT

This project is partially supported by Hong Kong RGC General Research Fund HKBU 212313 and HKBU 12202514.

REFERENCES

[1] X. Mei and H. Ling, "Robust visual tracking using L1 minimization," in *Proc. ICCV*, pages 1436-1443, 2009.

[2] F. Yang, Z. Jiang and L. S. Davis, "Online discriminative dictionary learning for visual tracking," in *Proc. WACV*, pages 854-861, 2014.

[3] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE Trans Pattern Anal. Mach. Intell.*, 34(4): 791–804, 2012.

[4] J. Xing, J. Gao, B. Li, W. Hu, and S. Yan, "Robust object tracking with online multi-lifespan dictionary learning," in *Proc. ICCV*, pages 665-672, 2013.

[5] H. Fan and J. Xiang, "Robust visual tracking with multitask joint dictionary learning," *IEEE Trans. Circuits Syst. Video Technol.*, DOI: 10.1109/TCSVT.2016.2515738.

[6] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Sig. Proc.*, 54(11): 4311–4322, 2006.

[7] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via multi-task sparse learning," in *Proc. CVPR*, pages 2042-2049, 2012.

[8] S. Zhang, H. Yao, X. Sun and S. Liu, "Sparse coding based visual tracking: Review and experimental comparison," *Pattern Recogn.*, 46(7): 1772-1788, 2013.

[9] X. Lan, A. J. Ma, and P. C. Yuen, "Multi-Cue Visual Tracking Using Robust Feature-Level Fusion Based on Joint Sparse Representation," in *Proc. CVPR*, pages 1194-1201, 2014.

[10] X. Lan, A. J. Ma, P. C. Yuen and R. Chellappa, "Joint Sparse Representation and Robust Feature-Level Fusion for Multi-Cue Visual Tracking." *IEEE Trans Image Process*., 24(12): 5826-5841, 2015 .

[11] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. CVPR*, pages 886-893, 2005.

[12] T. Ojala, M. Pietikäinen and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary pattern," *IEEE Trans Pattern Anal. Mach. Intell.*, 24(7): 971-987, 2002.

[13] L. Sevilla-Lara and E. Learned-Miller, "Distribution fields for tracking," in *CVPR*, pages 1910-1917, 2012.

[14] D. Ross, J. Lim, R. Lin and M. Yang, "Incremental learning for robust visual tracking," in *Int. J. Comput. Vision*, 77(1-3): 125-141, 2008.

[15] J. Yang, K. Yu, Y. Gong, and T. S. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proc. CVPR*, pages 1794–1801, 2009.

[16] Z. Jiang, Z. Lin, and L. S. Davis, "Learning a discriminative dictionary for sparse coding via label consistent k-svd," in *Proc. CVPR*, pages 1697–1704, 2011.