Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 15, no. 7, July 2019, pp. 1802-1812
DOI: 10.23940/ijpe.19.07.p6.18021812

Target Tracking Algorithm based on Context-Aware Deep Feature Compression

Ying Wang[a], Aili Wang[a,*], Ronghui Wang[b], Haiyang Liu[a], and Yuji Iwahori[c]

[a]*Higher Education Key Lab for Measuring and Control Technology and Instrumentations of Heilongjiang Harbin University of Science and Technology, Harbin, 150001, China*
[b]*Heilongjiang Province Public Security Department, Harbin, 150001, China*
[c]*Computer Science, Chubu University, Aichi, 487-8501, Japan*

Abstract

The main focus of target tracking is robustness and efficiency. Because of challenges such as background clutter, occlusion, and rotation, high robustness and efficiency cannot be achieved simultaneously. A tracking framework of perceptual correlation filter is improved to achieve high-speed computation between real-time trackers. The main contribution to high-speed computing speed comes from improved depth feature compression, which is realized by combining content-aware features with multiple automatic encoders. In the pre-training stage, an automatic encoder is trained for each class separately. In order to obtain the feature map suitable for target tracking, the orthogonal loss function is added to the training stage and the fine-tuning self-encoder stage. Experiments show that the improved algorithm demonstrates great improvement in accuracy and speed.

Keywords: target tracking; deep learning; context-aware feature; automatic encoder

(Submitted on December 8, 2018; Revised on January 11, 2019; Accepted on February 15, 2019)

1. Introduction

Intelligent video surveillance as a key application in the field of computer vision has received extensive attention and research [1-2]. By analyzing the content of the current scene, we can obtain the information of changes in the scene. Finally, we can detect and determine the location of the target and complete the monitoring work [3]. This also lays the foundation for the importance of video surveillance research and the necessity of target tracking technology research [4].

In 2008, Grabner et al. used supervised learning combined with semi-supervised learning to complete the classification of targets using Adaboost classifier, so as to achieve the target tracking [5]. In 2011, on the basis of multi-instance learning, an online update module was added and successfully solved the drift problem [6]. In 2012, Kalal proposed the TLD tracking algorithm, which combines target detection and target tracking technology, uses convolution neural networks to train the model, and constantly updates the tracking template to correct the deviation caused by tracking. It can effectively solve occlusion, deformation, and other problems [7]. In 2015, Henriques et al. proposed the Kernelized Correlation Filter (KCF), which has been greatly optimized in terms of running speed. By using cyclic matrix, it greatly reduces training time consumption and logic complexity [8]. In the same year, Danelljan et al. made improvements on the basis of KCF [9]. They mainly studied the loss function and deleted the boundary effect by adding a bias term to the loss function [10]. Subsequently, experiments were carried out by using CNN features to verify that the tracking tasks can be accomplished by using different levels of features. The tracking algorithm based on the discriminant model mainly trains image information in convolutional neural networks by using depth learning technology, so as to extract features from input images. Finally, target representation is carried out according to the extracted features, and target location is locked in subsequent frames.

2. Framework of the Tracking Algorithm

The focus of this algorithm is a context-aware depth feature compression tracker, which consists of multiple automatic

* Corresponding author.
E-mail address: aili925@hrbust.edu.cn

coders, context-aware networks, and related filters. The structure of the tracker is shown in Figure 1. The convolution features are derived from the compression of the original deep features by the automatic encoder of the VGG network. A context-aware network is improved to select the most suitable automatic encoder for target tracking. Experiments show that the automatic encoder can run during on-line tracking. After initializing the target of the automatic encoder, the feature map obtained is used as the input of the on-line target correlation filter.

In Figure 1, the automatic encoder is selected by the content-aware network and fine-tuned once by the ROI patch of the initial frame. For the following frames, we first extract ROI patches, which regard the previous target location as the center. Then, the original deep convolution feature is acquired by VGG-Net and compressed by the fine-tuned expert automatic encoder. The compression feature is the feature mapping of the correlation filter, and the target position is determined by the peak position of the filter response. After each frame, the correlation filter is updated online by the compressed feature of the newly discovered target.

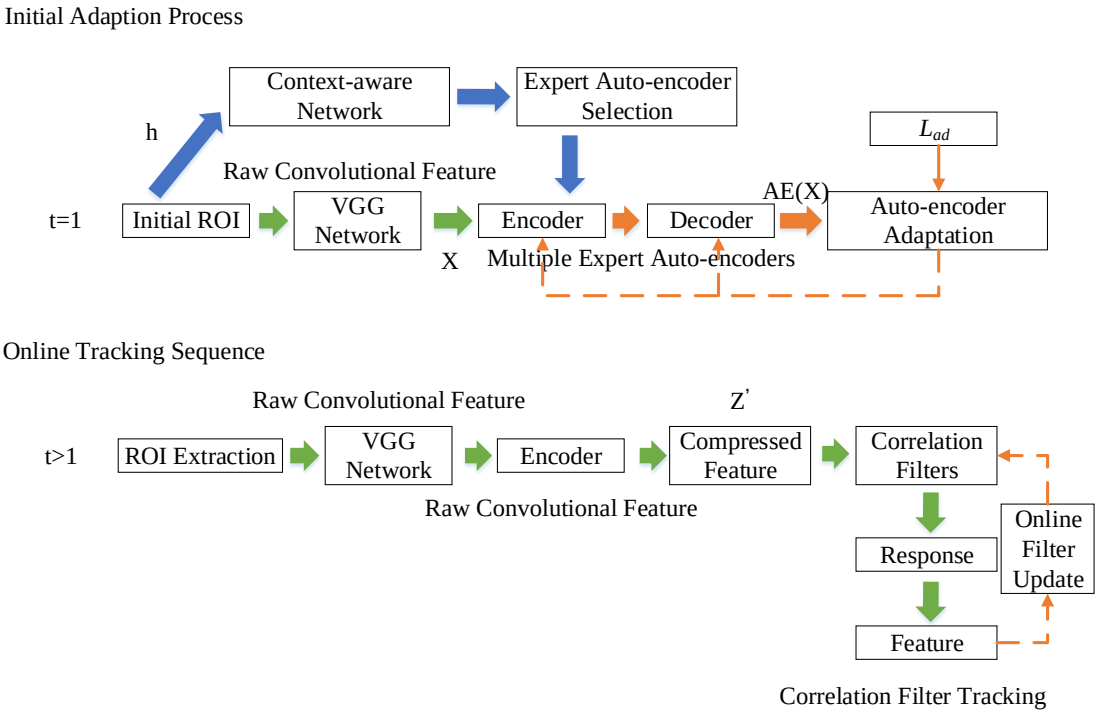


Figure 1. Context-aware network

3. Improved Auto-Encoder

The learning method of the automatic encoder is different from the traditional learning method. It adaptively learns according to the original input data samples. Therefore, the encoder can train different classes according to our different needs [11]. Because the training classes can be designated artificially, the learning process can be regarded as a directional training encoder process. For different needs, it can save a large amount of time wasted by repetitive work. If you want to build an automatic encoder, the main process includes three steps: firstly, build an automatic encoder, then build a corresponding decoder, and finally set up a loss function to ensure that the compression process does not lose too much of the original information.

Automatic encoder is a self-supervised learning algorithm that is essentially different from traditional unsupervised learning. Self-supervised learning generates corresponding labels mainly through the input image data. Since the self-supervised model is closely related to the loss function, an excellent loss function is indispensable to build a complete self-supervised model [12]. The advantages of automatic encoder are mainly embodied in two aspects: the denoising of data and the visual dimension reduction of complex models. At the same time, adding some appropriate constraints in the model can make the effect of automatic encoder more excellent.

The automatic encoder is a variant of the neural network. It has the structure and the core principle of the neural network. It can improve its performance by training, so that the input image can be converted into the expected image

according to the actual needs [13]. In the whole structure of the self-encoder, the inner hidden layer plays a decisive role in the coding output, so the whole network can be divided into two parts: the encoder for input information and the decoder for reconstructed information. If the decoder only has the ability of inverse transformation, then the decoder will have no meaning. Therefore, some constraints will be added to the encoder to ensure that the input and output are not exactly equal, but we can still get the target information at the output [14].

Figure 2 shows the structure of three-layer auto-encoder. In general, the automatic encoder has two processes that act on the input layer and the hidden layer.

The encoding process is shown as Equation (1):

$$h = g\theta_1(x) = \sigma(W_1 x + b_1) \tag{1}$$

The decoding process is shown as Equation (2):

$$\hat{x} = g\theta_2(h) = \sigma(W_2 x + b_2) \tag{2}$$

The compression loss is written as Equation (3):

$$J_E(W,b) = \frac{1}{m}\sum_{r=1}^{m}\frac{1}{2}\left\|\hat{x}^{(r)} - x^{(r)}\right\|^2 \tag{3}$$

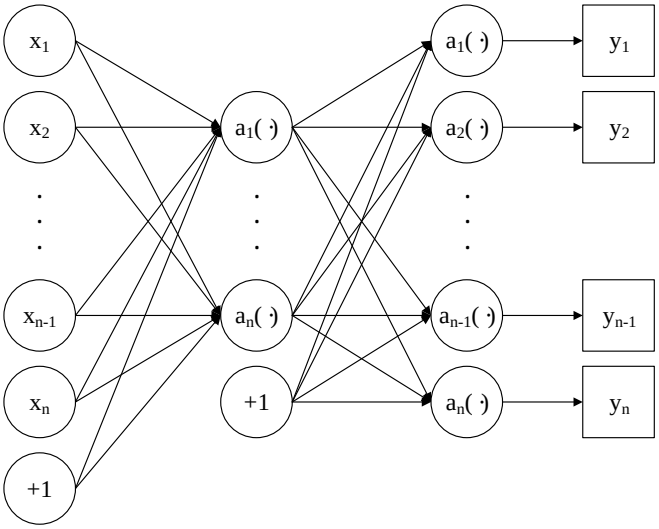


Figure 2. The structure of three-layer auto-encoder

In this paper, we choose a group of N_e self-encoders with the same structure, each of which covers different contexts. The compressed input is the original deep convolution feature map obtained from a convolution layer in the VGG network. In order to achieve a higher compression ratio, N_l coding compression layers are set up in the self-encoder, and an equivalent decompression layers are connected thereafter. The l^{th} compression layer f_l acts as a convolution layer, $\mathbb{R}^{w\times h\times c_l} \to \mathbb{R}^{w\times h\times c_{l+1}}$, so the resolution of feature mapping can be maintained by reducing the channel dimension c_l of the input image to the potential channel dimension c_{l+1}. The output of the f_l layer is used as the input of the f_{l+1} layer, so the channel dimension c decreases when the feature mapping passes through the coding layer. In the framework proposed in this chapter, a coding layer reduces the channel size by half, i.e., $c_{l+1} = c_l / 2$, $l \in \{1,\cdots,N_l\}$. Through the above information, the corresponding layer of decompression can be obtained, and the size of the input channel c_{k+1} can be expanded to c_k ($k \in \{1,\cdots,N_l\}$) for restoring the original size c_1 of the last layer x of the compression layer. Thus, self-coding can be expressed as Equation (4):

$$AE \equiv g_1(\cdots(g_{N_l}(f_{N_l}(\cdots(f_1(x))))))\in \mathbb{R}^{w\times h\times c_1} \tag{4}$$

The activation function of Rectified Linear Unit (ReLU) is added to all convolution layers, and the size of the convolution filter is set to 3×3.

4. Model and Network Construction of Content-Aware Deep Feature

4.1. Model Construction of Context-Aware Deep Feature Compression

The training process of the automatic encoder selected in this chapter can be divided into three stages. Firstly, all training samples are used to train the basic automatic encoder to find the context-independent initial compression feature mapping. Then, context clustering is applied to the initial compressed feature mapping of the automatic encoder to find the context-related clustering. Finally, these clustering methods are used to train the automatic coder and clustering samples initialized by the basic automatic coder [15].

There are two purposes of training with the basic automatic encoder: to cluster training samples using context-independent compression feature mapping and to find good initial weight parameters, so that the target automatic encoder can be fine-tuned [16]. The basic automatic encoder is trained by the original convolution feature mapping $\{x_j\}_{j=1}^m$ with batch (size of m), where x_j is obtained from the convolution layer. The convolution layer is contained in the VGG-Net, and the randomly selected training image I_j is fed from a large image database such as ImageNet, in which some spatial feature vectors of the convolution feature are randomly exchanged. Because the receiving field of the eigenvector covers different regions in the image, the exchange eigenvector is similar to the region in the exchange input image. Therefore, the exchange of feature vectors covering the background area and the target area leads to a similar effect as the background occlusion target. Therefore, the automatic encoder is trained to be robust under occlusion. After performing two denoising processes, we will represent $\{x_j\}_{j=1}^m$ as small batches. Then, the basic automatic encoder AE is trained by minimizing input feature mapping x_j using the distance between the noise sample $\hat{x}_j$ and its output $AE(x_j)$.

However, when only considering the distance between the input and the final output of the basic automatic encoder, the over-fitting problem and unstable training convergence are often observed. Therefore, a new loss based on multi-stage distance is selected, which consists of the distance between input and output obtained by some automatic encoders. A partial automatic encoder $\{AE_i(x)\}_{i=1}^{N_l}$ contains only a portion of the coding and decoding layer of its original automatic encoder $AE(x)$, and the size of the input and output matches that of the original automatic encoder, expressed as Equations (5) and (6):

$$AE_1(x) = g_1(f_1(x)) \tag{5}$$

$$AE_2(x) = g_1(g_1(g_1(f_1(x)))) \tag{6}$$

It can be concluded as Equation (7):

$$AE(x) = g_1(\cdots(g_{N_l}(f_{N_l}(\cdots(f_1(x))))))\tag{7}$$

The loss of distance between the input and final output is shown in Equation (8):

$$L_{ae} = \frac{1}{m}\sum_{j=1}^{m}\sum_{i=1}^{N_l}\left\|x_j - AE_i(\hat{x}_j)\right\|_2^2 \tag{8}$$

Where $AE_i(x)$ is the i^{th} part of the automatic encoder and m represents the size of small data sets.

Then, we cluster the training samples according to the feature map compressed by the basic automatic encoder, where N represents the total number of training samples. In order to avoid over-fitting of expert automatic coder due to the small clustering scale, we introduce a two-step clustering algorithm. In the first step, $2N$ sample are first obtained and are randomly selected from the feature graph compressed by the basic automatic encoder, which is twice the number of clusters required. In this experiment, 1000 times of random selection are repeated to find the sample with the largest Euclidean distance as the initial center of mass. Then, all training samples are clustered by the K-means clustering algorithm using compressed feature mapping ($k = 2N$). In the second step, the minimum number of samples is used to remove the centroid N of clustering. Then, the centroid N still exists, and because these centroids are used to cluster the training samples again, the clustering includes enough samples to avoid over-fitting. The cluster index is denoted here as $d_j \in \{1, \cdots, N\}$. Then, the training samples with context clustering index d are used to refine the basic automatic encoder and find the d th automatic encoder.

4.2. Network Design of Context-Aware Deep Feature Compression

For context-aware networks, we use a pre-trained VGG-M model, which contains a large amount of semantic information from pre-trained ImageNet. Given a 224×224 RGB input image, the content-aware network consists of three convolution layers $\{Conv1, Conv2, Conv3\}$ and the latter three full-connection layers $\{FC4, FC5, FC6\}$, where $\{Conv1, Conv2, Conv3, FC4\}$ are consistent with the corresponding layers in VGG-M. The activation function is then tightly linked to $FC5$, including 1024 output nodes. $FC6$ has N output nodes combined with the maximum pooling operation to estimate the probability that each automatic encoder is suitable for tracking the target.

The context-aware network takes training samples I_j as input and outputs the estimated probability of the samples belonging to the clustering index d_j . It is trained by batch image $\{I_j, d_j\}_{j=1}^{m}$ and cluster index pairs, where m is the size of the small batch of context-aware networks. The weights of $\{Conv1, Conv2, Conv3, FC4\}$ are determined by using stochastic gradient descent to minimize multiple loss functions L_{pr} , which is shown as Equation (9), and train the weights of $\{FC5, FC6\}$:

$$L_{pr} = \frac{1}{m}\sum_{j=1}^{m} H(d_j, h(I_j)) \tag{9}$$

Where H represents the cross-entropy loss, $h(I_j)$ is the clustering index I_j , and h represents the content-aware network.

As a part of target tracking, correlation filters are introduced in this paper. The functions of traditional correlation filters using single channel feature mapping are described below. Based on the properties of the Fourier domain cyclic matrix, correlation filters can be trained quickly, and high performance trackers can be obtained under low computational load. The vectorized single-channel training feature map $z \in \mathbb{R}^{wh\times 1}$ is given. At the same time, the vectorized target response map is obtained by using a two-dimensional Gauss window (size of $w \times h$). Thus, the vectorized correlation filter ω can be expressed by Equation (10):

$$\omega = \mathcal{F}^{-1}\left(\frac{\hat{z} \odot \hat{y}}{\hat{z} \odot \hat{z}^{*} + \lambda}\right) \tag{10}$$

Where $\hat{y}$ and $\hat{z}$ respectively represent the corresponding Fourier transform of vectors y and z , and $\hat{z}^{*}$ is a conjugate of $\hat{z}$, $\odot$ represents element multiplication, $\mathcal{F}^{-1}$ is the operator of inverse Fourier transform, and λ is a predefined regularization factor.

5. Process of Target Tracking

5.1. Initial Adaptation Process

The initial adaptation process consists of the following parts. Firstly, the region of interest (ROI) containing the target is

extracted from the initial frame, and the expert automatic encoder suitable for the target is selected through the context-aware network. Then, using the original convolution feature mapping of enhanced training samples from ROI, the selected expert automatic encoder is fine-tuned. ROI centers around the initial boundary of the target, which is 2.5 times larger than the size of the target to cover the surrounding area. Then, the width w and height h of the ROI are adjusted to 224×224 to match the expected input size of VGG-Net. For gray images, the gray value is copied three times to obtain $I^{(1)}$. Using the context-aware network h, the best automatic encoder for tracking scenes is selected according to the context information of the initial target, and the automatic encoder is represented as $AE^{h(I^{(1)})}$.

Even with the two denoising criteria described above, the compressed feature map of the automatic encoder still shows defects for targets that are blurred or flipped. Therefore, before fine-tuning the selected automatic encoder, $I^{(1)}$ is increased in several ways. In order to delete blurring, four enhanced images are obtained by Gauss filtering $I^{(1)}$ with variances {0.5, 1.0, 1.5, 2.0}. The other two enhanced images are obtained by flipping $I^{(1)}$ around the vertical axis and the horizontal axis respectively. Then, the original convolution feature mapping extracted from the enhanced samples can be represented as $\{x_j^{(1)}\}_{j=1}^{7}$.

The precision of the selected automatic encoder is different from the pre-training process of the expert automatic encoder. Due to the lack of training samples, it is difficult to obtain convergent results when applying denoising criteria. On the contrary, the orthogonality loss L_{ad} of correlation filters is adopted, which considers the orthogonality of correlation filters estimated from the compressed characteristic graph of expert self-encoder, and the definition of L_{ad} is shown in Equation (11):

$$L_{ad} = \sum_{i=1}^{N_l} \{ \left\| X_j^{(1)} - AE(X_j^{(1)}) \right\|_2^2 + \lambda_\theta \sum_{k,l=1}^{c^{i+1}} \theta(w_{jik}, w_{jil}) \} \tag{11}$$

Where $\theta(u,v) = (u,v)^2 / (\|u\|_2^2 \|v\|_2^2)$, w_{jik} is vector correlation filters, and Equation (3) is defined by using the k^{th} channel from the compressed feature map $f_i(\cdots(f_1(x_j^{(1)})))$ of the selected automatic encoder. The orthogonality loss of correlation filters allows for increased interaction between correlation filters estimated from different channels of the compressed feature graph. Fine tuning is achieved using random gradient descent to minimize L_{ad}.

The compressed feature map $Z^{\forall}$ can be obtained by a fine-tuned expert automatic encoder. Then, the channels in $Z^{\forall}$, which have greater response outside the target boundary box, are removed. These channels are found by estimating the channel direction ratio of the response of foreground and background features. Thus, the inter-channel ratio of the characteristic response of the channel can be estimated, as shown in Equation (12):

$$ratio^k = \frac{\left\| vec(Z_{bb}^{k,\forall}) \right\|_1}{\left\| vec(Z^{k,\forall}) \right\|_1} \tag{12}$$

Where the k channel feature map $Z^{k,\forall}$ of $Z^{\forall}$ is obtained by setting the value of the middle boundary box of $Z^{k,\forall}$ to 0. Then, after sorting all channels in descending order and ratio, only the first channel N_c of the compressed feature graph is used as the input of the correlation filter, and the feature map is expressed as $Z \in \mathbb{R}^{S \times S \times N_c}$ and S is the feature size.

5.2. Online Sequence Tracking

Firstly, the size of current frame *t* is obtained by using the same method as the initial adaptive method and adjusted to ROI, i.e., centering on the target center, which is 2.5 times the size of the target; therefore, the size is adjusted to 224×224. After the adjusted image is input into VGG-Net, the original deep convolution feature map is inserted into the adaptive automatic encoder to obtain the compression feature map $Z^{(t)} \in \mathbb{R}^{S \times S \times N}$.

Then, using Equation (9), we estimate the independent correlation filter $w^{k,(t)}$ of each feature map $Z^{k,(t)}$, where $Z^{(t)}$ represents the k^{th} channel of $Z(t)$. Next, the background region is suppressed by multiplying $Z^{k,(t)}$ to the cosine window of

the same size. For the first frame, we can estimate the correlation filter $w^{k,(1)}$ using $Z^{k,(1)}$ by Equation (13). For subsequent frames $(t>1)$, the correlation filter is as follows:

$$\bar{w}^{k,(t)} = (1-\gamma)\bar{w}^{k,(t-1)} + \gamma w^{k,(t)} \tag{13}$$

Where γ is the interpolation factor.

After estimating the correlation filter, we need to find the location $[x^t, y^t]$ of the target in the frame t. $[x^t, y^t]$ is assumed to be close to the target position $[x^{t-1}, y^{t-1}]$ in the previous frame. Then, an adaptive automatic encoder is used to obtain the compression feature mapping for tracking. $Z'^{(t)}$ and $\bar{w}^{k,(t)}$ will be inserted into Equation (10), and the channel response diagram $R^{k,(t)}$ will be provided.

Then, combine $R^{k,(t)}$ with an integrated response graph $R^{(t)}$. Therefore, a weighted averaging scheme is used, in which the verification score s^k is used as the weighting factor as shown in Equation (14):

$$s^k = \exp(-\lambda_s \left\| R^{k,(t)} - R_o^{k,(t)} \right\|_2^2) \tag{14}$$

Where $R_o^{k,(t)} = G(p^{k,(t)}, \sigma^2)_{S\times S}$. It is a two-dimensional Gauss window with a size of $S\times S$, and its variance σ is centered on $p^{k,(t)}$ with the peak point $R^{k,(t)}$. The comprehensive response diagram can be calculated according to Equation (15):

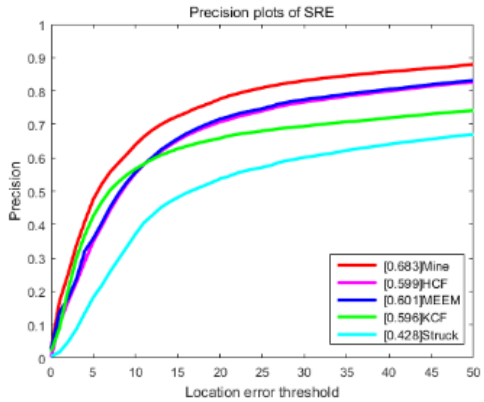
$$R^{(t)} = \sum_{k=1}^{N_c} s^k R^{k,(t)} \tag{15}$$

Finally, by inserting the response value near the peak value, the sub-pixel target location $p^{(t)}$ is obtained. The position $[x^t, y^t]$ of the target can be calculated by Equation (16):

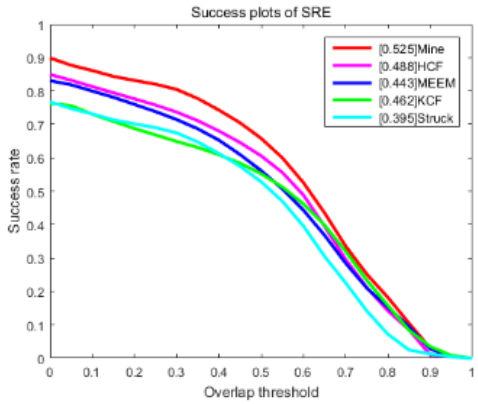
$$[x^t, y^t] = [x^{t-1}, y^{t-1}] + round([W, H] \square\ p^{(t)} / S) \tag{16}$$

6. Experimental Results and Analysis

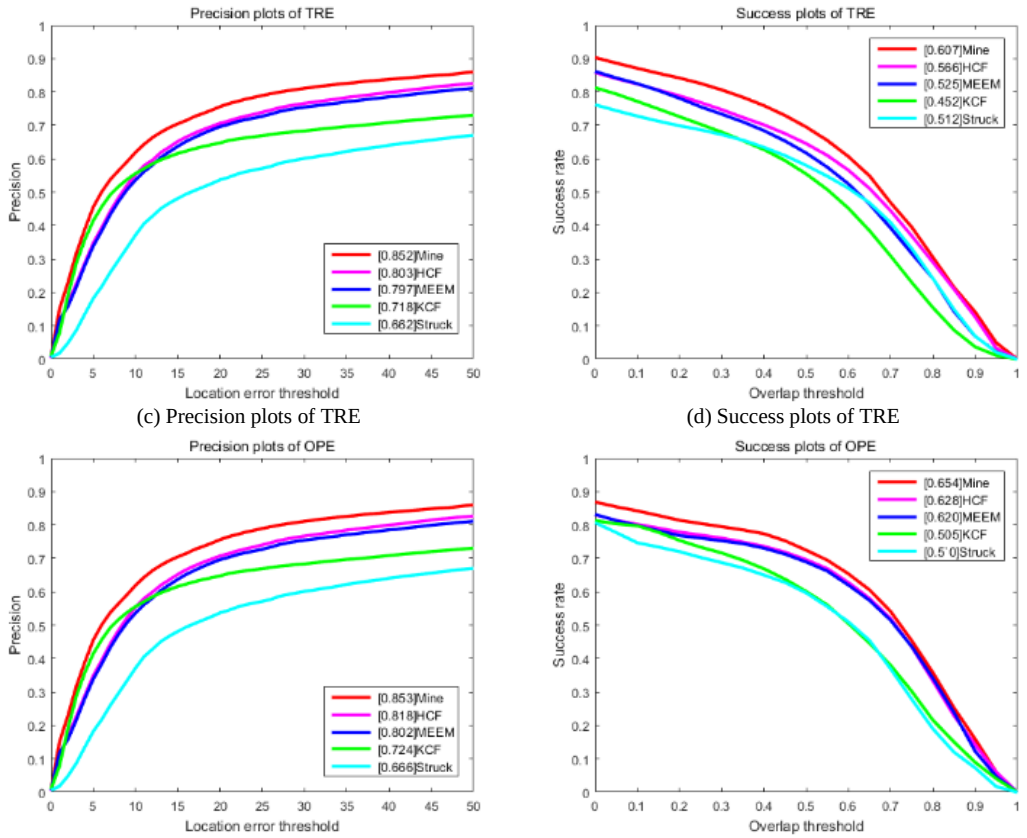
In this paper, the proposed method is evaluated on OTB-100. As can be seen in Figure 3, the values of SRE, TRE, and OPE are much higher than those of other trackers, and the lower the overlap threshold, the more advanced the method in this paper. With the increase in the overlap threshold, the success rate of all trackers will gradually tend to zero, and the gap will become smaller and smaller. For the accuracy chart, the accuracy of SRE, TRE, and OPE is 0 when the distance error threshold is 0. With the increase in the threshold value, the gap will be widened gradually, and the advantages of the algorithm in this paper will be more obvious.



(a) Precision plots of SRE



(b) Success plots of SRE



(c) Precision plots of TRE (d) Success plots of TRE

(e) Precision plots of OPE (f) Success plots of OPE

Figure 3. Success plots and precision plots of test sequences

Quantitative comparison was made with the HCF, MEEM, Struck, and KCF tracking methods in three aspects: distance precision (DP), overlap success (OS), and center location error (CLE), as shown in Table 1. The method in this chapter is expressed by H-CX. As can be seen from Table 1, H-CX demonstrates good consistency in the three aspects.

Table 1. Quantitative comparison of several trackers in the three aspects

Methods	DP (%)		OS (%)		CLE (pixel)	
	OTB-50	OTB-100	OTB-50	OTB-100	OTB-50	OTB-100
H-CX	82.37	76.88	70.76	60.59	13.16	18.20
HCF	80.39	75.53	66.80	59.15	14.33	20.72
MEEM	74.90	70.49	62.84	56.18	19.01	25.13
KCF	66.89	62.48	56.18	49.52	32.15	40.70
Struck	59.15	57.35	50.51	46.64	45.74	42.59

Table 2 gives the characteristics and speed comparison results of the above five tracking algorithms. It can be seen that H-CX is faster than other tracking algorithms, reaching 28.1 fps.

Table 2. The properties of the tracking algorithms

Tracking methods	Object representation	Model update	Frame speed (fps)
HCF	L, BP, DM	yes	21.5
MEEM	L, T	yes	10
Struck	H, Haar, DM	yes	20.2
KCF	L, SR, GM+DM	yes	0.51
H-CX	L, BP, DM	yes	28.1

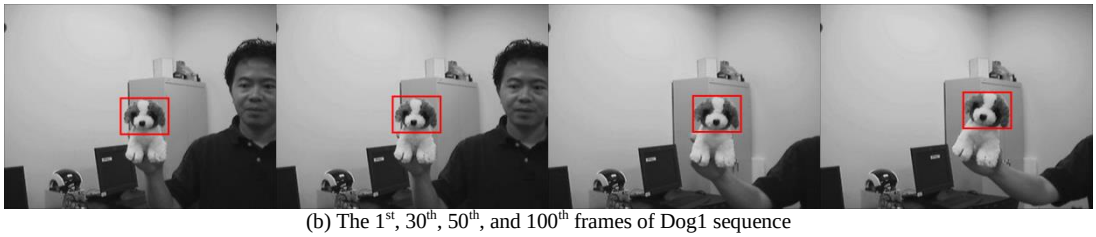
Table 3 gives the attributes of the video sequences in experiment, including illumination variation (IV), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR), out of view (OV), background clutter (BC), and low resolution (LR).

Table 3. The parameters of selected video frequency

Name	Frame number	Attribute
Biker	142	SV, OCC, MB, FM, OPR, OV, LR
Dog1	1350	SV, IPR, OPR
Panda	1000	SV, OCC, DEF, IPR, OPR, OV, LR
Dudek	1145	SV, OCC, DEF, FM, IPR, OPR, OV, BC
FleetFace	707	SV, DEF, MB, FM, IPR, OPR

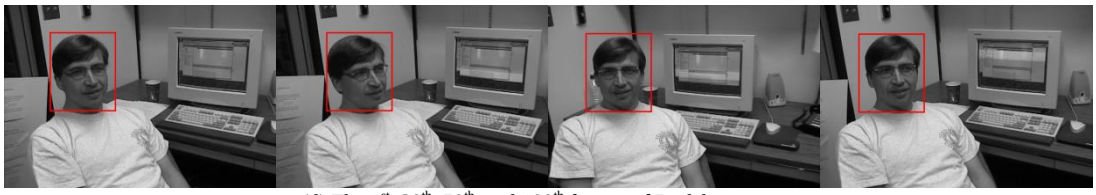
Figure 4 shows the tracking results of the video sequences. Each video sequence is analyzed by selecting the 1st, 30th, 50th, and 100th frames. As shown in Figure 3(a), the main target of the Biker sequence is a bicyclist, whose main test point is the combination of fast movement and in-plane rotation. Because the bicyclist's fast cycling will cause motion blurring in the picture and includes many turns, the tracker can obtain the position of the target in view of the above situation. From Figure 4(b), the target of the Dog1 sequence is a toy dog, which mainly combines rotation with scale change. In video, moving and rotating toy dogs will have a great impact on target recognition. Through experiments, it is found that the tracking algorithm in this paper can accurately locate the target position in and complete the tracking task. As shown in Figure 4(c), the main target of Panda sequence is a crawling panda. Its main test point is the combination of occlusion and deformation. Because the panda is moving all the time, it will be occluded by bamboo in the process of moving, which will change the appearance of the recognized target. At the same time, due to the obesity of the panda, it will make part of the body deform. In view of the above situation, the location of the target can be obtained in real time by our algorithm. As shown in Figure 4(d), this video sequence is more comprehensive and contains information about various challenges, such as rotation, deformation, and so on. The head of the character rotates left and right in the picture and changes expression. According to the experimental results, the tracker will not lose the target because of these challenges. As shown in Figure 4(e), a smiling face is the combination of moving and background clutter. As the characters move up and down, they constantly change the background of the target. At the same time, blurred images will be generated during the moving process. It can be seen in the figure that the tracker has not lost the target.

(a) The 1st、30th、50th、100th frames of Biker sequence



(b) The 1st, 30th, 50th, and 100th frames of Dog1 sequence

(c) The 1st, 30th, 50th, and 100th frames of Panda sequence



(d) The 1st, 30th, 50th, and 100th frames of Dudek sequence

(e) The 1st, 30th, 50th, and 100th frames of FleetFace sequence
Figure 4. Tracking results of selected video frequency

7. Conclusions

For the efficiency of target tracking, a tracking framework of perceptual correlation filter is improved to realize real-time and high-speed computation between trackers. The high-speed computing comes from improved depth feature compression, which is achieved by combining content-aware features with multiple automatic encoders. In the pre-training stage, an automatic encoder is trained for each class separately. In order to obtain the feature map suitable for target tracking, the orthogonal loss function is added to the training stage and the fine-tuning self-encoder stage.

References

1. D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual Object Tracking using Adaptive Correlation Filters," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2544-2550, 2010
2. J. Choi, H. Chang, J. Jeong, Y. Demiris, and J. Y. Choi, "Visual Tracking using Attention-Modulated Disintegration and Integration," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4321-4330, 2016
3. M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Convolutional Features for Correlation Filter based Visual Tracking," in *Proceedings of IEEE International Conference on Computer Vision Workshop*, pp. 621-629, 2015
4. M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Learning Spatially Regularized Correlation Filters for Visual Tracking," in *Proceedings of IEEE International Conference on Computer Vision*, pp. 4310-4318, 2015
5. B. Babenko, M. H. Yang, and S. Belongie, "Visual Tracking with Online Multiple Instance Learning," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 983-990, 2009
6. B. Babenko, M. H. Yang, and S. Belongie, "Robust Object Tracking with Online Multiple Instance Learning," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, Vol. 33, No. 8, pp. 1619-1632, 2011
7. T. C. Robert, "Mean Shift Blob Tracking through Scale Space," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 234-240, 2003
8. J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-Speed Tracking with Kernelized Correlation Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 37, No. 3, pp. 583-596, 2015
9. M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Learning Spatially Regularized Correlation Filters for Visual Tracking," in *Proceedings of IEEE International Conference on Computer Vision*, pp. 4310-4318, 2015
10. E. Park, H. Ju, Y. M. Jeong, and S. Y. Min, "Tracking-Learning-Detection Adopted Unsupervised Learning Algorithm," in *Proceedings of Seventh International Conference on Knowledge and Systems Engineering*, pp. 234-237, 2015
11. G. E. Hinton, S. Osindero, and Y. W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, Vol. 18, No. 7, pp. 1527-1554, 2006
12. J. Zhang, S. Ma, and S. S. MEEM, "Robust Tracking via Multiple Experts using Entropy Minimization," in *Proceedings of European Conference on Computer Vision*, pp. 188-203, 2014
13. Y. Wu, J. Lim, and M. H. Yang, "Online Object Tracking: A Benchmark," in *Proceedings of 2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2411-2418, 2013
14. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, et al., "Imagenet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, Vol. 115, No. 3, pp. 211-252, 2015
15. R. Zhao, W. L. Ouyang, H. S. Li, and X. G. Wang, "Saliency Detection by Multi-Context Deep Learning," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1265-1274, 2015
16. C. Ma, J. B. Huang, X. K. Yang, and M. H. Yang, "Hierarchical Convolutional Features for Visual Tracking," in *Proceedings of IEEE International Conference on Computer Vision*, pp. 3074-3082, 2015

Ying Wang is a master's student at Harbin University of Science and Technology. Her research interests include image superresolution and image classification.

Aili Wang received her B.S. degree from the Department of Electronic and Communication Engineering and her M.S. degree and Ph.D. from the Department of Information and Communication Engineering at Harbin Institute of Technology in 2002, 2004, and 2008, respectively. She joined Harbin University of Science and Technology as an assistant in 2004 and became an associate professor in 2010. She was a visiting professor at Chubu University in 2014. Her research interests include image superresolution, image fusion, and object tracking.

Ronghui Wang received his B.S. degree from the Department of Electronic and Communication Engineering at Harbin Institute of Technology in 2002.

Haiyang Liu is a master's student at Harbin University of Science and Technology. Her research interests include object tracking.

Yuji Iwahori received his B.S. degree from Nagoya Institute of Technology in 1983 and his M.S. degree and Ph.D. from Tokyo Institute of Technology in 1985 and 1988, respectively. He joined Nagoya Institute of Technology in 1988 and became a professor there in 2002. He joined Chubu University as a professor in 2004. He has also collaborated with UBC since 1991, IIT Guwahati since 2010, and Chulalongkorn University since 2014.