Visual Tracking via Exemplar Regression Model

Xiao Ma[a], Qiao Liu[b], Xiaohuan Lu[a], Zhenyu He[a,*]

[a]*School of Computer Science, Harbin Institute of Technology Shenzhen Graduate School, China*
[b]*School of Mathematic and Computer Science , Guizhou Normal University, China*

Abstract

Visual tracking remains a challenging problem in computer vision due to the intricate variation of target appearances. Some progresses made in recent years have revealed that correlation filters, which formulate the tracking process by creating a regressor in the frequency domain, have achieved remarkable experimental results on a large amount of video tracking sequences. On the contrary, building the regressor in the spatial domain directly has been considered as a limited approach since the number of training samples is restricted. And without sufficient training samples, the regressor will have less discriminability. In this paper, we demonstrate that, by giving a very simple positive-negative prior knowledge for the training samples, the performance of the ridge regression model can be improved by a large margin, even better than its frequency domain competitors-the correlation filters, on most challenging sequences. In particular, we build a regressor (or a score function) by learning a linear combination of some selected training samples. The selected samples consist of a large number of negative samples and a few positive ones. We constrain the combination such that only the coefficients of positive samples are positive, while all coefficients of negative samples are negative. The coefficients are learnt under such a regression setting that makes the outputs fit overlap ratios of the bounding box, where the overlap ratios are measured by calculating the overlaps between the inputs and the estimated position in the last frame. We call this regression *exemplar regression* because of the novel positive-negative arrangement of the linear combination. In addition, we provide a off-the-shelf method, a non-negative least square approach, to solve this regression model more efficiently. We evaluate our approach on both the standard CVPR2013 benchmark and the 50 selected challenging sequences, which include dozens of state-of-the-art trackers and more than 70 datasets in total. In both of the two experiments, our algorithm achieves a promising performance, which outperforms the state-of-

the-art approaches.

Keywords: visual tracking, kernelized ridge regression, exemplar regression model.

1. Introduction

Given an initial bounding box of a certain target in the first frame, a visual tracker estimates this target's state, e.g. location and scale, in each frame of the image sequences. Visual tracking is a key component in numerous applications, such as vision-based control, visual surveillance, human-computer interfaces, intelligent transportation, and augmented reality. Although some significant progress has been made in several decades of visual tracking research, most trackers are still prone to failure in challenging scenarios such as partial occlusion, deformation, motion blur, fast motion, illumination variation, background clutter and scale variations.

Among the achievements in visual tracking research, the discriminative approaches [1, 2, 3, 4, 5, 6, 7, 8, 9] provide an online mechanism that adapts appearance variations of the target, and gain better results than their generative rivals [10, 11, 12, 13] on some hard sequences [14]. Traditional discriminative approaches [5, 6, 7] maintain a classifier trained online to distinguish the target object from its surrounding background. This process can be divided into two stages: searching and updating. During the searching stage, the classifier is utilized to estimate the target's location in a certain search region around the estimated position from the previous frame, typically using a sliding-window [5, 7] or particle filtering approach [10]. In the updating stage, the traditional discriminative approaches generate a set of binary labelled training samples which are used to update the classifier online.

Although the traditional discriminative approaches gain convincing results in some hard sequences, it is difficult to arrange the binary labels for training samples in the updating stage. Because it is difficult to determine a pre-defined threshold and rule (e.g. Euclidean distance of one sample from the estimated target location from last frame) to decide whether a sample should be positive or negative.

*I am corresponding author
Email address: zyhe@hitsz.edu.cn (Zhenyu He)

To avoid the confusion of label arrangement, some discriminative approaches [9, 8, 1, 2, 4, 3] use the regression model instead of the traditional binary classification model during the updating stage. The regression models output a real-value score for each training sample to fit some pre-defined distributions, such as the bounding box overlap ratios [9, 8] or a Euclidean distance [1, 2, 4, 3]. However, the small sample size training problem makes the traditional regression models, such as the ridge regression, hard to create a robust regressor.

Some progresses [1, 2, 4, 3] made in recent years have revealed that solving the ridge regression in the frequency domain can achieve a dense sampling since it avoids the small sample size training problem. These approaches are called *correlation filters*. In this paper, different from correlation filters, we propose a simple approach to handle the small sample size training problem in the spatial domain directly. Our approach gives a positive-negative prior knowledge for the training samples. We demonstrate that, by adding such prior knowledge, the performance of the conventional spatial domain ridge regression can be improved by a large margin, even better than its frequency domain competitors-the correlation filters, on most challenging sequences.

our ridge regression model learns its score by calculating a linear combination of weighted similarities of some selected samples , as called *support samples* in this paper. Among those weights of similarities, we constrain that only the support samples in the estimated positions in the past frames are positive and the rest of the weights are all negative. This constraint gives us a large number of negative weights in the linear combination, but a few positive ones. We call the modified ridge regression *exemplar regression*. We show that the support samples and the weights can be solved by a non-negative least square method.

The main contribution of this paper is summarized as follows:

- We provide a simple positive-negative constraint method for a common kernelized ridge regression model to construct a robust visual tracker - we call exemplar regression tracking (ERT). The experiments show that the proposed ERT approach gains the state-of-the-art results under the standard CVPR2013 benchmark [1] [14] and other challenging sequences.

[1]This benchmark is first published in IEEE conference on Computer Vision and Pattern Recognition (CVPR) in 2013: https://sites.google.com/site/trackerbenchmark/benchmarks/v10

- We provide an easy-to-implement approach to solve the ERT based on the off-the-shelf non-negative least square method.

The rest of the paper is organized as follows: Section 3 describes the proposed ERT approach. The implementation details of the proposed approach are introduced in Section 4. In Section 5 we discuss the method. In Section 6 we performan extensive experimental comparison with the state-of-the-art visual trackers and we draw a conclusion in section 7.

2. Related Works

We refer to [15, 16, 17] for the detailed visual tracking surveys. In this section, we briefly review the most related online single object tracking, especially for the regression based approaches. The visual tracking approaches can be generally categorized as either generative [10, 11, 18, 13, 19, 20, 21, 22, 23, 24], or discriminative [5, 6, 7, 8, 9, 1, 2, 3, 4, 25, 26] based on their appearance models.

Generative approaches build an appearance model then use this model to find the optimal candidate samples with a certain region in the image frame which has the minimum construction error. Black et al. [20] learn an off-line subspace model to represent the object of interest for tracking. In [22], every image sample is fragmented into several patches, each of them is represented by an intensity histogram and compared to the corresponding patch in the target region by the Earth Movers Distance. Ross et al. [10] introduce the incremental PCA (Principal Component Analysis) to capture the full range of appearances of the target in the past frames. Mei et al. [11, 27] employ the sparse representation to build a dictionary which contains the appearances from past frames, then select the optimal appearance from this dictionary to estimate the target's positions. Li et al. [18] further extend the $l1$ tracking [11] by using the orthogonal matching pursuit algorithm to solve the optimization problems efficiently. In [19], an accelerated version of $l1$ tracking[11] is proposed based on the accelerated proximal gradient (APG) approach. Jia et. al. [13] and Wang et. al [12] introduce the patch-based sparse representation to enhance the tracker's robustness. In [21], Kwon et al. combine multiple observation and motion models to handle large appearance and motion variation.

Table 1: Comparisons of the proposed exemplar regression with the popular regression based tracking approach Struck[9] and correlation filters (CFs)[1, 4, 3]

Regression models	Searching strategies	Regressors	Updating algorithms
Struck [9]	Sliding window	$f(\mathbf{x}) = \sum_i \omega_i k(\mathbf{x}_i, \mathbf{x})$	LaRank [28]
Exemplar regression	Particle filtering	$f(\mathbf{x}) = \frac{1}{\|\mathbf{x}\|} \sum_i \omega_i < \mathbf{x}_i, \mathbf{x} >$	Non-negative least square
Correlation filters [4, 3]	Convolution	$f(\mathbf{X}) = \mathscr{F}^{-1}(\frac{\mathscr{F}(\mathbf{X}) \odot (\mathscr{F}(\mathbf{X}_o))^*}{\mathscr{F}(\mathbf{X}_o) \odot (\mathscr{F}(\mathbf{X}_o))^* + \lambda})$	Linear ridge regression

The traditional discriminative approaches pose the tracking problem as a binary classification task with local search and determine the decision boundary for separating the target object from the background. Collins et al. [26] provide an approach which learns the discriminative features online to separate the target object from the background. In [25], an online boosting algorithm is proposed to select features for tracking. Babenko et al. [7] introduce the multiple instance learning approach to decide the labels of the training samples, and integrate an online boosting approach to select the Haar-like features for tracking. A semi-supervised learning approach [6] is proposed in which positive and negative samples are selected via an online classifier with structural constraints. Zhang et al. [5] employ a random projected compressed features space with a very high dimension to represent the target's appearances, and a binary classifier is used to find the optimal image sample in this compressed domain.

One of the typical regression based trackers is Struck[9], which builds a regression model upon the Structural Support Vector Machine (SSVM) framework, and embeds the novel LaRank [28] algorithm into the updating stages. In [9], a regressor is formulated by the weighted linear combination of a set of support vectors. These support vectors might come from different frames during the tracking process. Therefore, Struck has a good ability to adapt of appearance variances. [9] uses the Haar-like features and sliding-window searching strategy, which makes it difficult to handle scale variances of the target. In [8], the authors integrate the Lie group theory to promote Struck's scale-aware ability.

Another popular regression model is Correlation Filters (CFs) [1, 2, 4, 3]. In a few years, CFs has proved itself to be competitive with far more complicated approaches. The use of Fast Fourier

Transform (FFT) makes it run at a very high speed online(about hundreds of frames-per-second). CFs takes advantage of the famous theorem that the convolution of two image samples in a spatial domain is equivalent to an element-wise product in the Fourier domain. By formulating the CF's objective function in the Fourier domain, it can achieve dense sampling during the updating and searching stage without being very much time consuming.

Comparison of paradigms between the exemplar regression model, Struck and CFs is shown in Table 1. The original Struck [9] applies a sliding-window to find the optimal sample. The regressor in Struck is a linear combination of weighted kernel functions between support samples and the test sample, which is very similar to ours. Struck introduces the LaRank [28] to construct its regressor. CFs use the convolution between the constructed regressor (or filter) and the search region's image patch to find the optimal position. However, a single correlation filter can not handle the scale variation of the target. To promote the CFs scale-aware ability, DSST [3] creates another filter using the target's features at different scales to estimate the target's scale. CFs utilize the ridge regression in the Fourier domain to construct the filter.

The formulation of our regressor is similar to Struck [9], since they both consider the support samples from past frames and arrange the positive weights only for samples within the estimated positions. However, Struck introduces a complicated LaRank [28] to choose which samples should be added to the regressor's linear combination and how large are the weights that should be arranged. Without using the LaRank, we suggest using a more effective and easy-to-implement approach to do those selections and arrangements: non-negative least square.

We borrow the concept of *exemplar* from Exemplar-SVM [29]. For the detection problem, Exemplar-SVM trains a separate linear SVM classifier for every exemplar in the training set. Each Exemplar-SVM is defined by a single positive instance and millions of negatives.

3. Exemplar Regression Tracking

As a discriminative approach, our exemplar regression tracker has searching and updating stage during the tracking process. In the searching stage (Section 3.1), we analyze the probability of the observation model in the Bayesian tracking framework. The observation model is provided by the updating stage, which is in the form of the linear combination of weighted similarities between

chosen support samples and the test sample. The formulation of this exemplar regression model and how to choose the support samples and the weights of similarities are introduced in Section 3.2 and Section 3.3.

3.1. The searching stage

Visual tracking can be cast as a sequential Bayesian inference problem [10]. Given a set of observed image patches $\mathbf{I}_t$ up to the t-th frame, we aim to estimate the value of the state variable $Z_t \in \mathbb{R}^4$, which describes the bounding box of the observed image patch. The true posterior state distribution $Pr(Z_t|\mathbf{I}_t)$ is approximated by a set of N_s samples, called tracking candidates, $\mathbf{Z}_t = \{Z_t^1, Z_t^2, \cdots, Z_t^{N_s}\}$, and the optimal state is estimated by the MAP(Maximum A Posteriori) formulation:

$$\hat{Z}_t = \underset{Z_t^i}{\operatorname{argmax}}\, Pr(Z_t^i|\mathbf{I}_t), \tag{1}$$

where Z_t^i denotes the state variable of the i-th candidate sample at the t-th frame. Using Bayesian theory, the posterior probability $Pr(Z_t^i|\mathbf{I}_t)$ is inferred by:

$$Pr(Z_t^i|\mathbf{I}_t) \propto Pr(I_t^i|Z_t^i) \int Pr(Z_t^i|Z_{t-1}^i) Pr(Z_{t-1}^i|\mathbf{I}_{t-1}) dZ_{t-1}^i, \tag{2}$$

where $Pr(Z_t^i|Z_{t-1}^i)$ is the dynamic model. I_t^i is the t-th frame's i-th observed image patch. We use the same dynamic model which is described in [30, 10]. The observation model $Pr(I_t^i|Z_t^i)$ is determined by our proposed exemplar regression model.

Given an observed image patch I_t^i, we use $\mathbf{x}_t^i \in \mathbb{R}^d$ to represent the features extracted in I_t^i, where d is the dimension of this feature vector. In the rest of this paper, $\mathbf{x}_t^i$ is the *sample* in state Z_t^i. We introduce a function of $\mathbf{x}_t^i$ to approximate $Pr(I_t^i|Z_t^i)$:

$$Pr(I_t^i|Z_t^i) \propto f(\mathbf{x}_t^i). \tag{3}$$

The function $f(\cdot)$ helps us find the one that is most likely to be a target from the observed image patches $\mathbf{I}_t$, and gives this optimal one the highest output. Therefore, all we need to do in the searching stage is two things: 1) apply the particle filtering strategy to generate N_s state variables $\mathbf{Z}_t$ and their relevant samples; 2) utilize $f(\cdot)$ to find the optimal state variable.

3.2. The exemplar regression model

For a common kernelized ridge regression model, we want to find a linear function (the regressor) in a certain projected Hilbert space $\mathscr{S}$:

$$f(\mathbf{x}) =< \mathbf{W}, \Phi(\mathbf{x}) >_k, \tag{4}$$

where $\mathbf{W}$ is the unknown coefficient, $\Phi(\cdot) : \mathbb{R}^d \rightarrow \mathscr{S}$ maps the d dimension feature vector to that Hilbert space $\mathscr{S}$. $<,>_k$ denotes the inner-product in $\mathscr{S}$. We use the following objective function to solve $\mathbf{W}$ in the ridge regression setting:

$$\sum_i \| < \mathbf{W}, \Phi(\mathbf{x}^i) >_k - y^i \|_2^2 + \frac{\lambda}{2} < \mathbf{W}, \mathbf{W} >_k, \tag{5}$$

where $\{y^i\}, y^i \in \mathbb{R}$ and $\{\mathbf{x}^i\}$ are the training set, and λ is the regularization parameter.

From the *representer theorem* [31], the regressor f can be constructed by a linear combination of the weighted kernel function between support samples and the test sample [32]:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \omega_i \, Ker(\mathbf{x^i}, \mathbf{x}) \tag{6}$$

where $\mathbf{x^i}$ denotes the support samples. We call $\mathbf{X} = \{\mathbf{x}^1, \mathbf{x}^2, \cdots, \mathbf{x}^N\}$ the *support sample set*. ω_i represents the weight of the i-th kernel function value. In this paper, we choose the normalized inner-product in the $\mathbb{R}^d$ to represent $Ker(,)$:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \omega_i \frac{< \mathbf{x^i}, \mathbf{x} >}{\|\mathbf{x^i}\|_2 \|\mathbf{x}\|_2}, \tag{7}$$

Suppose all the support samples are normalized, which means $\|\mathbf{x_i}\|_2 = 1$, then we have:

$$f(\mathbf{x}) = \frac{1}{\|\mathbf{x}\|_2} \sum_{i=1}^{N} \omega_i < \mathbf{x^i}, \mathbf{x} >, \tag{8}$$

During the t-th frame's updating stage, we draw N_u samples around the estimated state variable $\hat{Z}_t : \mathbf{X_t} = [\mathbf{x_t^1}, \mathbf{x_t^2}, \cdots, \mathbf{x_t^{N_u}}] \in \mathbb{R}^{d \times N_u}$, and we call $\mathbf{X_t}$ the *training sample set*. We define a score value for each samples in $\mathbf{X}_t$ using the following function:

$$s(\mathbf{x}_t^i) = overlap(\hat{Z}_t, Z_t^i), \tag{9}$$

where the $overlap(\cdot,\cdot)$ function calculates the overlap between two bounding boxes of state variables. Z_t^i is the state variable of $\mathbf{x}_t^i$. $s(\cdot)$ gives the higher output which has the larger overlap with the estimated state $\hat{Z}_t^1$, and lower output with the smaller overlaps. Using $s(\cdot)$, we can generate the outputs for $\mathbf{X}_t$ in a regression model: $Y_t = [y_t^1, y_t^2, \cdots, y_t^{N_u}]^T = [s(\mathbf{x}_t^1), s(\mathbf{x}_t^2), \cdots, s(\mathbf{x}_t^{N_u})]^T \in \mathbb{R}^{N_u}$.

We define a loss function for the regressor $f(\cdot)$:

$$E(f(\mathbf{X}_t), Y_t) = \sum_{i=1}^{N_u} (y_t^i - f(\mathbf{x}_t^i))^2. \tag{10}$$

Putting Eq.8 into Eq.10 and adding the exemplar constraint, we have the exemplar regression model's objective function:

$$\begin{aligned} E(f(\mathbf{X}_t), Y_t) &= \sum_{i=1}^{N_u} (y_t^i - \frac{1}{\|\mathbf{x_t^i}\|_2} \sum_{j=1}^{N} \omega_j < \mathbf{x^j}, \mathbf{x}_t^i >)^2 \\ & s.t. \\ & \forall \delta(\mathbf{x^j}) \omega_j > 0, \end{aligned} \tag{11}$$

where

$$\delta(\mathbf{x}) = \begin{cases} 1 & \mathbf{x} \text{ is positive support sample} \\ -1 & \text{otherwise} \end{cases} \tag{12}$$

is the indicator which implements the exemplar constraint. Note that, for each frame, only the sample with the estimated state variable is positive, and the rest of the samples are negative.

3.3. Solving the exemplar regression using non-negative least square

In the updating stage, we want to add or remove support samples from the *candidate sample set* , which consists of the old support sample set $\mathbf{X}$ and the normalized training sample set $\bar{\mathbf{X}}_t$: $\mathscr{X}_t = \{\mathbf{X}, \mathbf{X}_t\} = \{\mathbf{x^1}, \mathbf{x^2}, \cdots, \mathbf{x^N}, \bar{\mathbf{x}}_\mathbf{t}^\mathbf{1}, \bar{\mathbf{x}}_\mathbf{t}^\mathbf{2}, \cdots, \bar{\mathbf{x}}_\mathbf{t}^{\mathbf{N_u}}\}$, where the first N samples are old normalized support samples of the un-updated regressor. The last N_u samples are from $\mathbf{X}_t$ and are all also

to be normalized. The arrangement above reveals that the updating process reselects the old support samples and adds the new support samples from the training samples, and the corresponding weights will also be re-allocated. This formulation helps the regression model to adapt the target appearance variations smoothly. For every candidate sample $\mathbf{x^i}$ ($i = 1, 2, \cdots, N + N_u$) $\in \mathscr{X}$, we define a weight ω_i and a label to indicate if it is positive or not. Note that the labels of the first N samples are determined by the un-updated regressor, and among the last N_u samples, only the $\bar{\mathbf{x}}_t^1$ is positive, while the rest are all negative.

For now, the new regressor is represented by:

$$f(\mathbf{x}) = \frac{1}{\|\mathbf{x}\|_2} \sum_{i=1}^{N+N_u} \delta(\mathbf{x}^i)|\omega_i| < \mathbf{x^i}, \mathbf{x} > . \tag{13}$$

Note that we want to use the training sample set $\mathbf{X}_t$ to learn a exemplar regression model. This means we want to find a set of support samples in candidate sample set $\mathscr{X}_t$ and their corresponding weights to minimize Eq.11.

For every sample $\mathbf{x}_t^i$ in training sample set $\mathbf{X}_t$, we can obtain a loss between the regressor output and the required one y_t^i:

$$E(f(\mathbf{x}_t^i), y_t^i) = (y_t^i - \frac{1}{\|\mathbf{x}_t^i\|_2} \sum_{j=1}^{N+N_u} \delta(\mathbf{x}^j)|\omega_j| < \mathbf{x}^j, \mathbf{x}_t^i >)^2, \tag{14}$$

where the $\mathbf{x}_t^i$ can be represented by the normalized $\bar{\mathbf{x}}_t^i$, which is $\mathbf{x}^{N+i}$ in the candidate sample set $\mathscr{X}_t$. Then Eq.14 can be simplified by:

$$E(f(\mathbf{x}_t^i), y_t^i) = (y_t^i - \sum_{j=1}^{N+N_u} \delta(\mathbf{x}^j)|\omega_j| < \mathbf{x}^j, \mathbf{x}^{N+i} >)^2. \tag{15}$$

The overall loss of the exemplar regression model on the training set $\mathbf{X}_t$ can be calculated by:

$$E(f(\mathbf{X}_t), Y_t) = \sum_{i=1}^{N_u} (y_t^i - \sum_{j=1}^{N+N_u} \delta(\mathbf{x}^j)|\omega_j| < \mathbf{x}^j, \mathbf{x}^{N+i} >)^2. \tag{16}$$

Let:

$$K_{j,N+i} = \delta(\mathbf{x}^j) < \mathbf{x}^j, \mathbf{x}^{N+i} >, \tag{17}$$

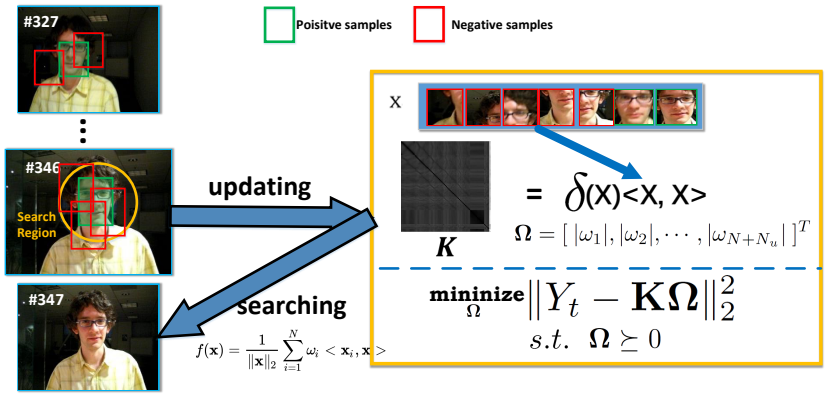


Figure 1: Overview of the proposed exemplar regression tracking (ERT) approach: the formulation of the regressor is a linear combination of weighted inner-products of the support samples with the test sample. This combination is formulated by the following two steps: 1) Construct an inner-product matrix K using the samples from the current frame and past frames, and the signs of those inner-products are constrained by the pre-defined positive and negative settings ($\delta(\mathbf{X})$); 2) Solve a non-negative least square problem to find the weights Ω.

then we can construct an *inner-product matrix* $\mathbf{K}$:

$$\mathbf{K} = \begin{bmatrix} K_{1,N+1}, & K_{2,N+1}, & \cdots, & K_{N+N_u,N+1} \\ \vdots, & \vdots, & & \vdots \\ K_{1,N+2}, & K_{2,N+2}, & \cdots, & K_{N+N_u,N+2} \\ K_{1,N+N_u}, & K_{2,N+N_u}, & \cdots, & K_{N+N_u,N+N_u} \end{bmatrix}. \tag{18}$$

We use a more compact way to formulate Eq.16:

$$E(f(\mathbf{X}_t), Y_t) = \|Y_t - \mathbf{K}\mathbf{\Omega}\|_2^2, \tag{19}$$

where:

$$\mathbf{\Omega} = [\ |\omega_1|, |\omega_2|, \cdots, |\omega_{N+N_u}|\ ]^T \tag{20}$$

Since $\mathbf{\Omega} \succeq 0$, the solution of Eq.19 can be obtained by the standard non-negative least square (NNLS) method [33]. An overview of the proposed method is shown in Fig. 1.

3.4. The updating stage

In the updating stage of the tracking process, we draw N_u training samples to update the old regressor using the approach provided above. First, we use the training samples and old support samples to construct the inner-product matrix by Eq.18. Then a standard NNLS approach is applied to solve Eq.19. This gives us a vector of weights (Eq. 20). We retain the samples with the large weights, and discard the samples with small weights. We take those retained samples to be the new support samples of the regressor.

4. Implementation details

During the tracking process, the image patches are all resized to 32×32. In the searching stage, the number of observed image patches $\mathscr{N}$ is set to 600, of which 500 samples are used to approximate translations, and 100 samples are arranged to estimate the scale and stretch transforms. This means that for the accuracy of scale estimation, after the translation of the target is estimated using the 500 samples, we draw another 100 samples around this estimated position to obtain the scale transforms. We use the PCA-HOG(Histogram of Oriented Gradient) features [34] to describe the observed image patches, for which the cell-size is set to 4 and the number of orientation bins is set to 9, and this process gives us a 2048 dimension feature vector for each image patch.

In the updating stage, we obtain the training sample set $\mathbf{X}_t$ on the polar grid, and like Struck [9], we set the radial to 5 and angular divisions to 16, giving us 81 training samples. The candidate sample set $\mathscr{X}_t$ is constructed like Fig.2, which consists of the 81 training samples and the N old support samples from the un-updated regressor. These samples are divided into different parts within $\mathscr{X}_t$ according to whether the samples are positive or not. This treatment helps us arrange the labels for the inner-product matrix K. We use the Matlab's built-in function $lsqnonneg()$ to solve Eq.19. A summary of this ERT method is shown in Algorithm 1.

4.1. Lazy update strategies

The exemplar model of the visual tracker can be updated frame-by-frame over the video sequences to adapt to the targets' changing appearance. However, this continuous update strategy makes the support sample set of the model become huge and the tracking process will become

Algorithm 1 The exemplar regression tracking

Inputs: Z_0: the initial state variables; N_u: the number of samples during the updating stage; N_s: the number of samples during the searching stage; ϵ: the threshold for selecting the support samples.

In every frame, we manage the regressor Eq. 13 involving N support samples;

Outputs: the estimated state variables $\hat{Z}_t$ in every frames.

while The video sequence is not ended **do**

    Draw N_s samples using the particle filtering, and use Eq.13 to find the optimal state variable;

    Construct the inner-product matrix $\mathbf{K}$ according to the description Section 3.3;

    Find the weights of samples in $\mathscr{X}_t$ by solving Eq.19;

    Select the support samples by removing the sample whose weight is less than ϵ;

    Update the regressor using the new support samples $\{\mathbf{x}^i\}$ and the corresponding weights $\{\omega_i\}$.

end while

very slow. Moreover, for most video sequences, the appearance of targets does not undergo large variations in the continuous small number of frames. Therefore, we provide three *lazy update strategies* to decide when to update the model dynamically:

4.1.1. Checking the positive samples

This strategy will check the similarity of the tracking result of the current frame with the positive samples of the support sample set. If there is at least one positive sample making the similarity less than a threshold, then the update process should be executed. This strategy helps the tracker update the model when the appearance does not change greatly.

4.1.2. Checking the contexts

The contexts are the image patches within the targets' bounding box with a certain padding ratio. This includes the targets and the surroundings. This strategy calculates the similarity of the contexts in the two continuous frames, and avoids repetitive updating when the targets are motionless.

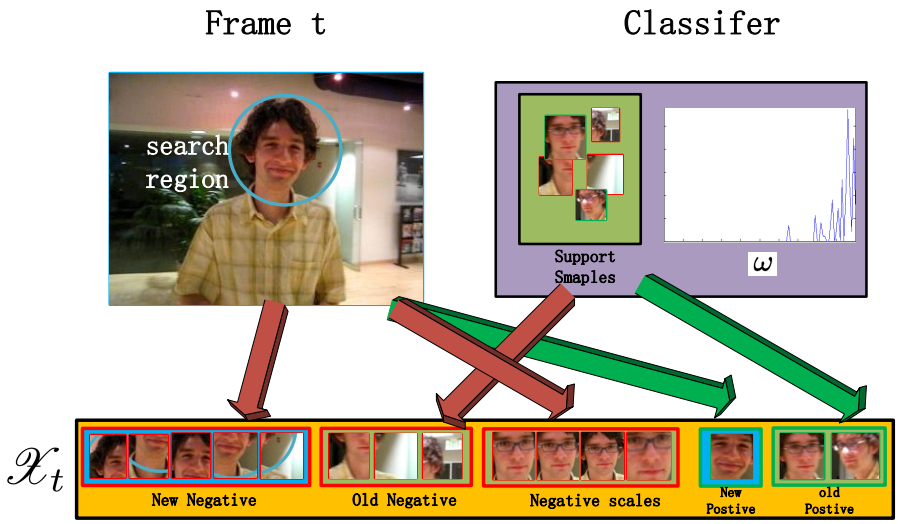


Figure 2: Construction of the candidate sample set $\mathscr{X}_t$.

4.1.3. Checking the variance of test samples

This strategy obtains the variance of the regressor's output of the test samples, and compares it with a given threshold to decide when to update the model.

5. Discussion

In this section, we present some discussions of our exemplar regression model. We explain the model from the templates construction perspective. We believe this explanation will be beneficial to the formulations provided above.

5.1. The templates construction perspective

A visual tracking problem can be formulated as a templates matching problem. The templates are the image samples which represent the targets' appearances. The templates are used to calculate the similarities between the test image samples in the new frame, and to choose the test sample with the greatest similarity to the tracking result. The templates can consist of a single image or multiple images. Our approach, Struck [9] and correlation filters [4, 1, 3] are typical single template models. Sparse coding based approaches [11, 13] preserve a set of templates and choose the most representative template for the matching process.

In every frame, our exemplar regression model generate a set of image samples around the estimated position. This process gives us a large number of samples which describe the target's appearances, while negative samples contain the background information. One of the naive approaches to create the single template is casting all the samples as training samples into the regression settings, such as ridge regression, or support vector regression (SVR). However, many of the samples are superfluous, and this redundancy makes the construction of the regression model nearly infeasible such that it can not be used for online visual tracking. Moreover, the template created by the naive approaches will have a weak discriminative ability because the number of samples that have background information is much larger than the number of samples of targets. To understand this, we must clarify the *templates* in the regression settings. Consider a linear regression model's score function:

$$y = \omega^T \mathbf{x} + b, \tag{21}$$

The ω can be solved by the ridge regression or support vector regression, and depends on the loss term in the objective function. Eq.22 can also be simply represented as:

$$y = sim(\omega, \mathbf{x}) + b = < \omega, \mathbf{x} > + b, \tag{22}$$

which means that the score is determined by the similarity between a certain template ω and the test sample $\mathbf{x}$. Note that this template explanation has already been used to model the visualization for convolutional networks [35] and in the context of Bayesian classification [36]. From the *representer theorem* [31], we know that template ω is spanned by the training samples:

$$\omega = \sum_i \alpha_i \mathbf{x}_i, \tag{23}$$

where $\mathbf{x}_i$ is the training sample, and α_i is the corresponding coefficient. Since there are a large number of training samples that contain background information, if we don't constrain the value of those coefficients, some coefficients of background image samples could be positive, which will make the constructed template ω have weak discriminative ability. Recall the exemplar constraint of our approach, which actually gives the positive samples (the estimated targets samples) positive coefficients, and negative coefficients for negative samples (containing background information).

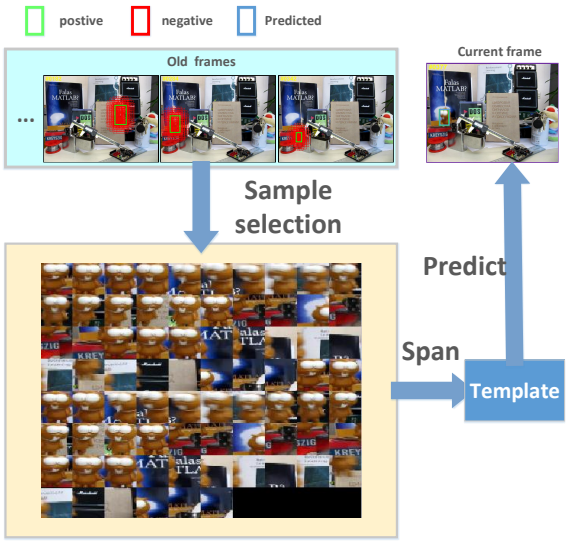


Figure 3: For each frame of the given image sequences, we can obtain a set of image samples consisting of only one positive sample and many negative samples. This gives us a large number of samples. Since many of them are redundant, we provide an efficient sample selection approach to filter the important sample set, and use this sample set to span the discriminative template.

This constraint can enhance the discriminative ability of template ω because it penalizes the background information within the negative samples' influence. Meanwhile, it strengthens the effects of the positive samples in the linear combination of template ω.

It is worth noting that the *sample selection* effect of the exemplar regression model. Since we discard the small weights of samples during the updating stage (see Section 3.4), the sample selection has choosen the more representative samples to be preserved. This sample selection helps the trackers to decide which samples generated frame-by-frame are important for representing the appearances of targets, and avoids the large sample set problem of the regression model. See Fig.3 for the summary of the templates construction perspective.

6. Experiments

We evaluate our exemplar regression tracking approach on two experiments. In the first experiment, we evaluate our approach on a large benchmark [14] that contains 50 videos with comparisons to state-of-the-art trackers. The overall experimental results are illustrated by both precision

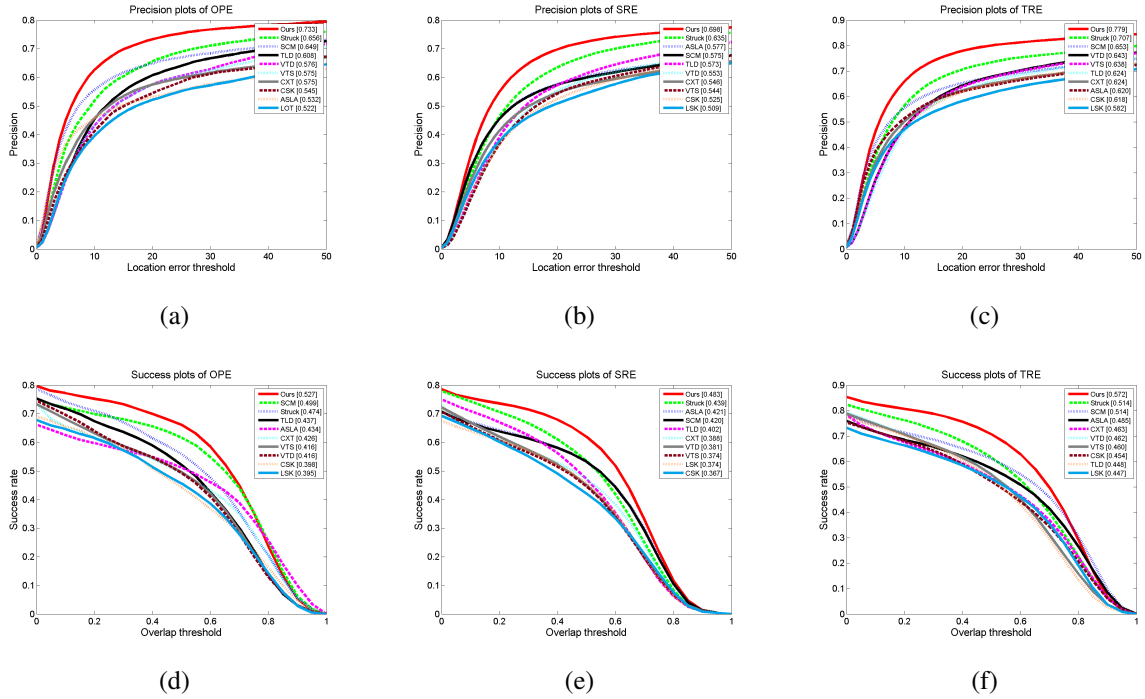


Figure 4: Distance precision and overlap success plots over standard 50 benchmark sequences [14] using OPE, TRE and SRE. The legend contains the area-under-the-curve score for the top-10 trackers.

plots and success plots. The results of eleven challenging attributes of fast motion, background clutter, blur, deformation, illumination, in-plane rotation, low resolution, occlusion, out-of-view, and out-of-plane rotation and scale variations are also provided. In the second experiment, to further demonstrate our approach's robustness, we select another 50 challenging sequences and compare our method with the state-of-the-art regression based trackers: Struck [9], KCF [4] and DSST [3]. The results of this experiment are also illustrated by both precision plots and success plots. In addition, a detailed analysis of some typical sequences is provided.

6.1. Experimental settings

Our ERT tracker is implemented in Matlab and runs at approximately 4 frames per second on an Intel Core i5 3.30GHz CPU with 8GB RAM. Note that for all of the experiments, we use the parameters provided in Section 4 and all are fixed.

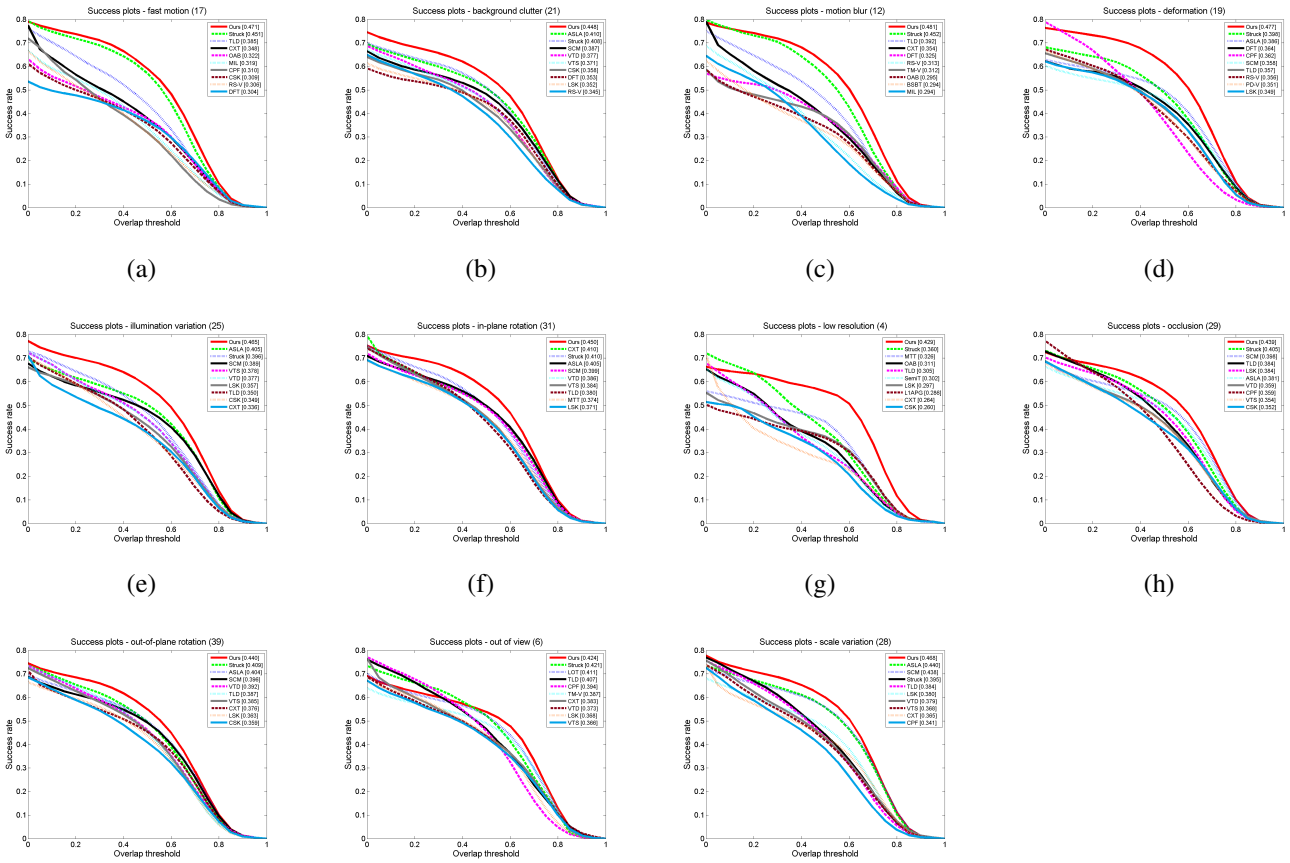


Figure 5: Overlap success plots over the eleven challenges of fast motion, background clutter, blur, deformation, illumination, in-plane rotation, low resolution, occlusion, out-of-view, and out-of-plane rotation and scale variations on the standard CVPR2013 benchmark. The legend contains the AUC score for the top-10 trackers. Our method performs favorably against the state-of-the-art trackers.

6.2. Evaluation criteria

In this section, we introduce the evaluation methodology used in the two experiments. We use the precision and success rate for quantitative evaluation:

6.2.1. Precision plot

The first evaluation metric is the CLE (Center Location Error), which is defined as the average Euclidean distance between the center locations of the tracked targets and the manually labeled ground truth. The average center location error over all the frames of one sequence is used to evaluate the overall performance for that sequence. We use the score at the 20 pixels CLE as the representative precision score for each tracker.

6.2.2. Success plot

Another evaluation metric is the bounding box overlap. We use the typical Pascal VOR (VOC Overlap Ratio) criterion [37]. Given the bounding box B_R of the result and the bounding box B_G of the ground truth, the VOR can be computed as VOR $= |\frac{B_R \cap B_G}{B_R \cup B_G}|$, where $\cap$ and $\cup$ represent the intersection and union of two regions, respectively, and $|\cdot|$ denotes the number of pixels in the region. To measure the performance on a given sequence, we count the number of successful frames whose VOR is larger than the given threshold t_o. The success plot shows the ratios of successful frames at the thresholds varied from 0 to 1. We use the AUC (Area Under Curve) of each success plot to rank the compared tracking approaches.

6.2.3. Precision plot on single challenging sequence

To compare the tracking approaches on some typical challenging sequences, we plot the tracking results of trackers on each frame in the sequence.

To evaluate the robustness of the visual tracking approaches, we use three different evaluations:

- OPE (One Pass Evaluation): this evaluation runs the compared trackers throughout a test sequence with an initialization ground truth position in the first frame;
- SRE (Spatial Robustness Evaluation): to evaluate whether a tracking method is sensitive to initialization errors, this evaluation generates the object states by slightly shifting or scaling

the ground truth bounding box of a target object. In our experiments, we use eight spatial shifts (four center shifts and four corner shifts), and four scale variations,according to the benchmark [14]. We run the SRE evaluation 12 times, and the final scores of SRE are averaged to rank the compared tracking approaches.

- TRE (Temporal Robustness Evaluation): in this evaluation, each compared tracking approach is evaluated numerous times from different starting frames across an image sequence. In each test, an algorithm is evaluated from a particular starting frame, with the initialization of the corresponding ground truth object state, until the end of an image sequence. In our experiments, we run TRE 20 times and use the averaged results to generate the TRE scores.

6.3. Experiment on CVPR2013 benchmark

In this section, we give the detailed experiment results of the standard CVPR2013 benchmark [14]. We compare our exemplar regression tracking approach with 29 trackers: CPF [38], LOT [39], IVT [10], ASLA [13], SCM [40], L1APG [19], MTT [41], VTD [21], VTS [42], LSK [43], ORIA [44], DFT [45], KMS [46], SMS [47], VR-V [26], Frag [22], OAB [25], SemiT [48], BSBT [49], MIL [7], CT [5], TLD [6], Struck [9], CSK [2] and CXT [50]. These 29 trackers cover the classical discriminative and generative visual tracking approaches, and some of them gain the state-of-the-art tracking results.

The CVPR2013 benchmark provides 50 sequences. To evaluate the trackers robustness, these sequences are categorized with 11 challenging attributes: fast motion (FM), background clutter (BC), motion blur (MB), deformation (DEF), illumination (IV), in-plane rotation (IPR), low resolution (LR), occlusion (OCC), out-of-view (OV), and out-of-plane rotation (OPR) and scale variations (SV). Each sequence includes several attributes. Among the 50 sequences, there are 39 sequences with OR attributes, 31 sequences with IR attributes, 29 sequences with OCC attributes, 28 sequences with SV attributes, 25 sequences with IV attributes, 21 sequences with BC attributes, 19 sequences with DEF attributes, 17 sequences with FM attributes, 12 sequences with MB attributes, 6 sequences with OOV attributes and 4 with LR attributes. In Fig. 6, we give the detailed results for each of the 11 challenging attributes. Our approach achieves the best benchmark results on all the 29 classical visual trackers except the OV attribute.

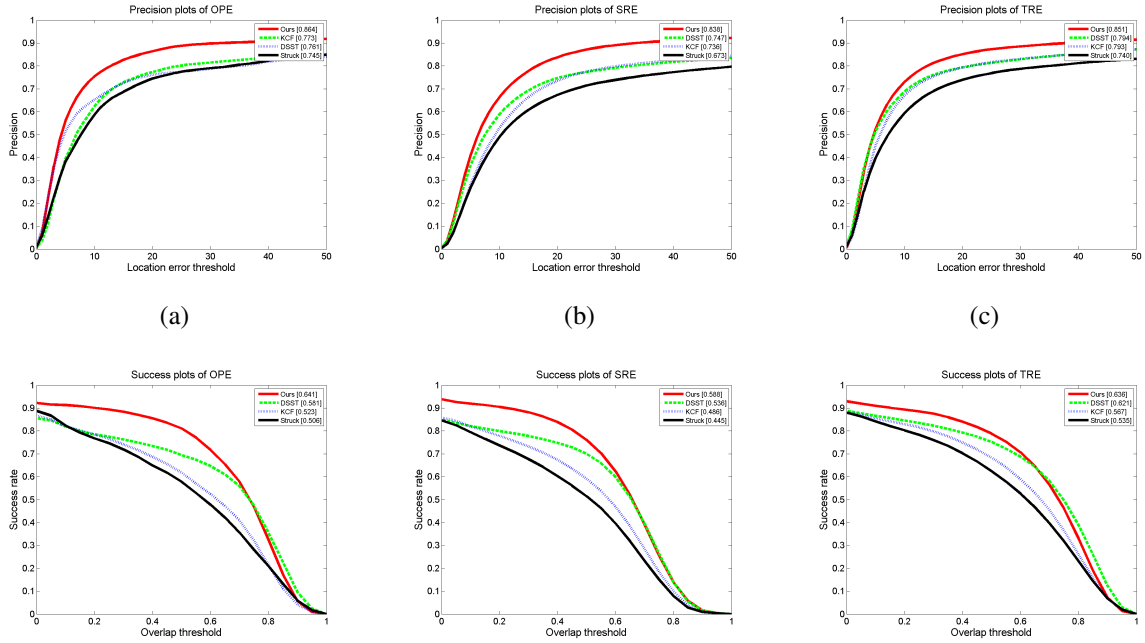


Figure 6: Comparisons between our method with the state-of-the-art regression based visual trackers, Struck [9], DSST [3] and KCF [4] on the selected 50 sequences using the OPE, SRE and TRE. Our method outperforms other trackers on the three evaluations, and by nearly 10% on the OPE.

In Fig. 4, we give the overall OPE, TRE and SRE results on the benchmark sequences. Our approach outperforms the second best approach Struck [9] with a large margine. The average CLE and VOR results of our approach and 7 visual trackers are presented in Tab. 2.

Table 2: The average comparisons of our approach with the 7 trackers on the 50 sequences of CVPR2013 benchmark in CLE at a threshold of 20 pixels and VOR at a threshold of 0.5. The value format of each table cell is "CLE/VOR". The best result of each sequence is highlighted by bold red and the second best is highlighted by bold blue.

	Ours	Struck[9]	MIL[7]	ASLA[13]	TLD[6]	CSK[2]	SCM[40]	L1APG[19]
CLE/VOR	**0.73/0.66**	**0.66**/0.56	0.47/0.37	0.53/0.51	0.61/0.52	0.54/0.44	0.65/**0.62**	0.48/0.44

6.4. Experiment on the 50 challenging sequences

To further demonstrate our approach's robustness comparing the state-of-the-art regression trackers: Struck [9], KCF [4] and DSST [3], we run our approach and the three approaches on the

Figure 7: Tracking results of four kinds of regression based trackers: our approach, structured output SVM(Struck [9]), correlation filters (KCF [4] and DSST [3]) on 16 challenging sequences(from left to right and top down are Human6, Human3, Human5, BlurBody, Lemming, Singer2, Car1, Couple, Freeman1, Freeman3, Human7, Human9, Trellis, David3Outdoor, CarScale, Jumping).

selected 50 challenging sequences: BlurBody, BlurCar1, BlurCar4, BlurFace, Board, Box, Boy, Car1, Car4, CarDark, CarScale, Coke, Couple, Crossing, Crowds, David, David2, David3Outdoor, Deer, Dog, Dog1, Doll, Dudek, FaceOcc1, FaceOcc2, Fish, FleetFace, Football1, Freeman1, Freeman3, Human3, Human4, Human5, Human6, Human7, Human9, Jumping, Lemming, Man, singer1, Skating1, Subway, Surfer, Sylvester, Tiger1, Tiger2, Trellis, Walking, Walking2 and Woman.

The overall experiment results are shown in Fig. 6, where our method is better than the three trackers by nearly 10% on OPE. The average comparisons are presented in Tab. 3.

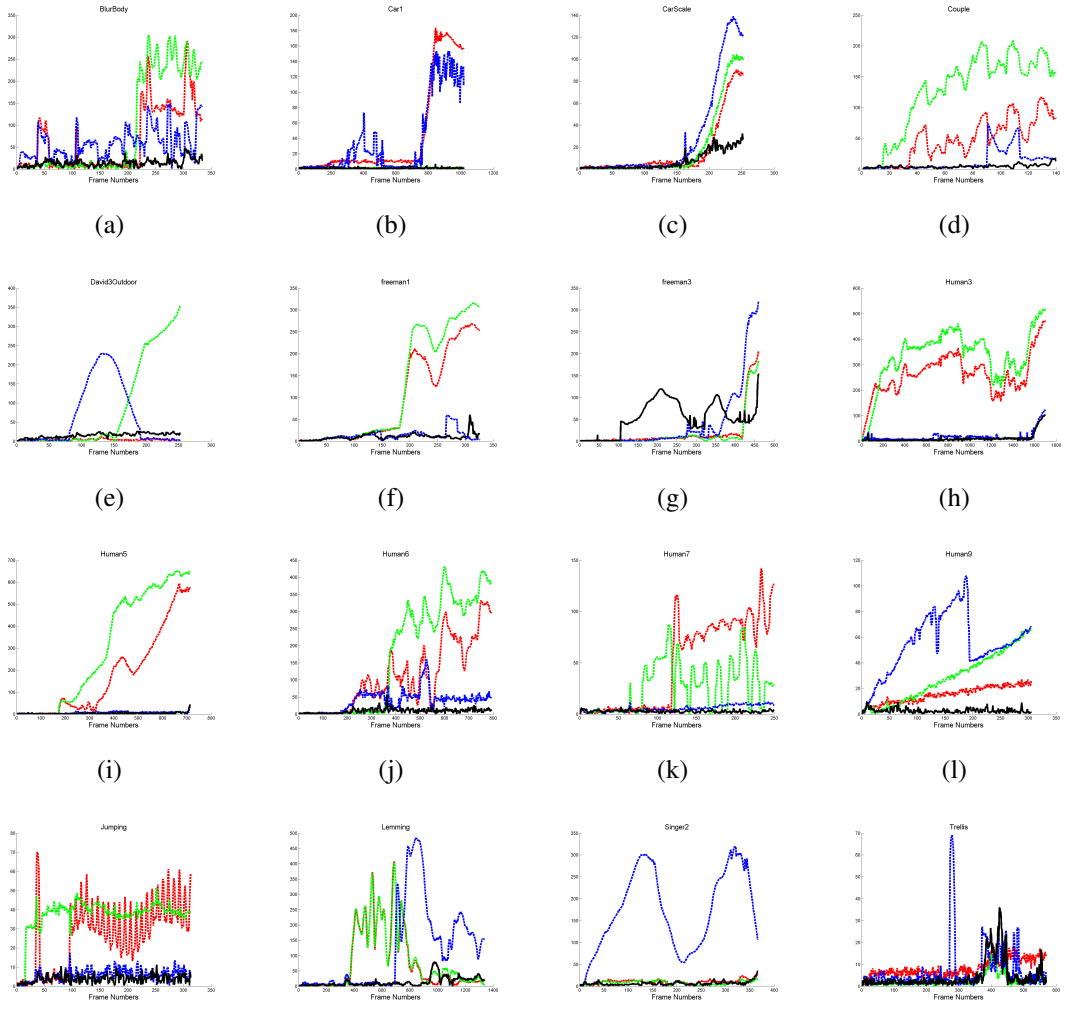
We compare our approach with the three state-of-the-art regression based approaches Struck

Table 3: The average comparisons of our approach with the 3 state-of-the are regression based trackers on the selected 50 sequences. The format are same as Tab. 2.

	Ours	KCF[4]	DSST[3]	Struck[9]
CLE/VOR	**0.86**/**0.81**	**0.77**/0.62	0.76/**0.70**	0.74/0.58

[9], KCF [4] and DSST [3] on 16 challenging sequences for the 50 sequences in detail (Fig. 7). The KCF approach is based on a correlation filter learned from HOG features [51]. The KCF approach performs well in handling significant deformation and fast motion (BlurBody, Trellis, and Singer2) due to the robust representation of HOG features and effectiveness of the temporal context correlation model. However, it fails to handle large scale variation (Human6, Human5, Car1, Freeman3 and CarScale) because of the fixed sizes convolution searching strategy. Besides, it drifts when there exists large motion blur (Couple, Human7 and Jumping), since the searching regions of KCF are determined by the initial bounding box and fixed during the tracking process. In addition, because of the linear updating of the filter, KCF can handle short-term occlusions (David3Outdoor) smoothly, but can not handle the long-term (Human5, Human3 and Lemming). The Struck [9] approach can not handle the large scale variation (Human6, Human5, Car1, Freeman3 and CarScale) either because of the traditional fixed size Haar-like features [52] representation. In addition, it drifts when the targets on the gray images are not obvious because of the limitation of the Haar-like features representation (Singer2). The DSST [3] enhances the scale aware ability of KCF by integrating a new $1 - D$ filter to predict the scale variations of the targets. Therefore, it obtains better experimental results on Car1, Trellis, CarScale and Singer2 than KCF. However, it does not resolve the inherent drawbacks of correlation filters such as the linear updating strategy which can make the trackers fail on the long-term occlusions (Lemming, Human3 and Human5), while the problem of the limited size of the searching regions means that it cannot handle the large motion blur (Human9, Couple and Jumping).

Our approach achieves remarkable results on all the 16 challenging sequences (Fig. 8). The utilization of the PCA-HOG features representation makes our approach robust when handling the illumination variations (Singer2 and Trellis). The sample selection effect of the exemplar regression model helps the tracker handle the long-term and short-term occlusions (Lemming,



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

Figure 8: Frame-by-frame comparison of center location errors on the 16 challenging sequences illustrated in Figure 7. Generally, our method is able to track targets accurately and stably.The meanings of the colors of lines are the same as in Figure 7.

David3Outdoor, and Human5) and gain good results on significant deformation (Singer2 and Freeman1). By integrating the particle filtering and the fixed size templates (32×32), our approach can handle large scale variations (Freeman3, Carscale and Human6).

7. Conclusion

Although few researchers have noticed this, by giving a very simple negative-positive constraint for the training samples set of a common kernelized regression model, a state-of-the-art robust visual tracker can be constructed. We constrain the linear combination of the score function of the kernelized regression model to make sure that only the support samples in the estimated positions from past frames are positive and the rest of the weights are all negative. We show that this novel linear combination can be solved by a simple off-the-shelf non-negative least square method.

8. Acknowledge

This study was supported by Science and Technology Project of Shenzhen (NO. JSGG20150331152017052). This study was also supported in part by Shenzhen IOT key technology and application systems integration engineering laboratory.

References

[1] D. Bolme, J. Beveridge, B. Draper, Y. M. Lui, Visual object tracking using adaptive correlation filters, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, pp. 2544–2550. 2, 3, 4, 5, 14

[2] J. F. Henriques, R. Caseiro, P. Martins, J. Batista, Exploiting the circulant structure of tracking-by-detection with kernels, in: Computer Vision–ECCV 2012, Springer, 2012, pp. 702–715. 2, 3, 4, 5, 20, 21

[3] M. Danelljan, G. Häger, F. Khan, M. Felsberg, Accurate scale estimation for robust visual tracking, in: British Machine Vision Conference, Nottingham, September 1-5, 2014, BMVA Press, 2014. 2, 3, 4, 5, 6, 14, 17, 21, 22, 23

[4] J. Henriques, R. Caseiro, P. Martins, J. Batista, High-speed tracking with kernelized correlation filters, Pattern Analysis and Machine Intelligence, IEEE Transactions on 37 (3) (2015) 583–596. 2, 3, 4, 5, 14, 17, 21, 22, 23

[5] K. Zhang, L. Zhang, M.-H. Yang, Fast compressive tracking, Pattern Analysis and Machine Intelligence, IEEE Transactions on 36 (10) (2014) 2002–2015. 2, 4, 5, 20

[6] Z. Kalal, K. Mikolajczyk, J. Matas, Tracking-learning-detection, Pattern Analysis and Machine Intelligence, IEEE Transactions on 34 (7) (2012) 1409–1422. 2, 4, 5, 20, 21

[7] B. Babenko, M.-H. Yang, S. Belongie, Visual tracking with online multiple instance learning, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, 2009, pp. 983–990. 2, 4, 5, 20, 21

[8] G. Zhu, F. Porikli, Y. Ming, H. Li, Lie-struck: Affine tracking on lie groups using structured svm, in: Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on, 2015, pp. 63–70. 2, 3, 4, 5

[9] S. Hare, A. Saffari, P. Torr, Struck: Structured output tracking with kernels, in: Computer Vision (ICCV), 2011 IEEE International Conference on, 2011, pp. 263–270. 2, 3, 4, 5, 6, 12, 14, 17, 20, 21, 22, 23

[10] D. A. Ross, J. Lim, R.-S. Lin, M.-H. Yang, Incremental learning for robust visual tracking, International Journal of Computer Vision 77 (1-3) (2008) 125–141. 2, 4, 7, 20

[11] X. Mei, H. Ling, Robust visual tracking using $l1$ minimization, in: Computer Vision, 2009 IEEE 12th International Conference on, 2009, pp. 1436–1443. 2, 4, 14

[12] Q. Wang, F. Chen, W. Xu, M.-H. Yang, Online discriminative object tracking with local sparse representation, in: Applications of Computer Vision (WACV), 2012 IEEE Workshop on, 2012, pp. 425–432. 2, 4

[13] X. Jia, H. Lu, M.-H. Yang, Visual tracking via adaptive structural local sparse appearance model, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, 2012, pp. 1822–1829. 2, 4, 14, 20, 21

[14] Y. Wu, J. Lim, M.-H. Yang, Online object tracking: A benchmark, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013. 2, 3, 16, 17, 20

[15] A. Yilmaz, O. Javed, M. Shah, Object tracking: A survey, Acm Computing Surveys 38 (1) (2006) 8193. 4

[16] M.-H. Yang, J. Ho, Toward robust online visual tracking, in: Distributed Video Sensor Networks, Springer, 2011, pp. 119–136. 4

[17] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, M. Shah, Visual tracking: an experimental survey, Pattern Analysis and Machine Intelligence, IEEE Transactions on 36 (7) (2014) 1442–1468. 4

[18] H. Li, C. Shen, Q. Shi, Real-time visual tracking using compressive sensing, in: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE, 2011, pp. 1305–1312. 4

[19] H. Ji, Real time robust l1 tracker using accelerated proximal gradient approach, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 1830–1837. 4, 20, 21

[20] M. J. Black, A. D. Jepson, Eigentracking: Robust matching and tracking of articulated objects using a view-based representation, International Journal of Computer Vision 26 (1) (1998) 63–84. 4

[21] J. Kwon, K. M. Lee, Visual tracking decomposition, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE, 2010, pp. 1269–1276. 4, 20

[22] A. Adam, E. Rivlin, I. Shimshoni, Robust fragments-based tracking using the integral histogram, in: Computer vision and pattern recognition, 2006 IEEE Computer Society Conference on, Vol. 1, IEEE, 2006, pp. 798–805. 4, 20

[23] X. Li, Z. He, X. You, C. P. Chen, A novel joint tracker based on occlusion detection, Knowledge-Based Systems 71 (2014) 409–418. 4

[24] N. V. Lopes, P. Couto, A. Jurio, P. Melo-Pinto, Hierarchical fuzzy logic based approach for object tracking,

Knowledge-Based Systems 54 (2013) 255–268. 4

[25] H. Grabner, M. Grabner, H. Bischof, Real-time tracking via on-line boosting., in: BMVC, Vol. 1, 2006, p. 6. 4, 5, 20

[26] R. T. Collins, Y. Liu, M. Leordeanu, Online selection of discriminative tracking features, Pattern Analysis and Machine Intelligence, IEEE Transactions on 27 (10) (2005) 1631–1643. 4, 5, 20

[27] X. Mei, H. Ling, Robust visual tracking and vehicle classification via sparse representation, Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (11) (2011) 2259–2272. 4

[28] A. Bordes, L. Bottou, P. Gallinari, J. Weston, Solving multiclass support vector machines with larank, in: Z. Ghahramani (Ed.), Proceedings of the 24th International Machine Learning Conference, OmniPress, Corvallis, Oregon, 2007, pp. 89–96. 5, 6

[29] T. Malisiewicz, A. Gupta, A. Efros, Ensemble of exemplar-svms for object detection and beyond, in: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE, 2011, pp. 89–96. 6

[30] M. Isard, A. Blake, Condensationconditional density propagation for visual tracking, International journal of computer vision 29 (1) (1998) 5–28. 7

[31] B. Scholkopf, R. Herbrich, A. J. Smola, R. Williamson, A generalized representer theorem, Proceedings of Annual Conference on Computational Learning Theory 42 (3) (2000) 416–426. 8, 15

[32] C. M. Bishop, Pattern recognition and machine learning, springer, 2006. 8

[33] C. L. Lawson, R. J. Hanson, Solving least squares problems, Vol. 161, SIAM, 1974. 11

[34] P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, Pattern Analysis and Machine Intelligence, IEEE Transactions on 32 (9) (2010) 1627–1645. 12

[35] K. Simonyan, A. Vedaldi, A. Zisserman, Deep inside convolutional networks: Visualising image classification models and saliency maps, CoRR abs/1312.6034.
URL http://arxiv.org/abs/1312.6034 15

[36] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, K. R. Mller, How to explain individual classification decisions, Journal of Machine Learning Research 11 (9) (2010) 1803–1831. 15

[37] L. Cehovin, M. Kristan, A. Leonardis, Is my new tracker really better than yours?, in: Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on, IEEE, 2014, pp. 540–547. 19

[38] P. Pérez, C. Hue, J. Vermaak, M. Gangnet, Color based probabilistic tracking, in: Computer visionECCV 2002, Springer, 2002, pp. 661–675. 20

[39] S. Oron, A. Bar Hillel, D. Levi, S. Avidan, Locally orderless tracking, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 1940–1947. 20

[40] W. Zhong, H. Lu, M.-H. Yang, Robust object tracking via sparsity-based collaborative model, in: Computer vision and pattern recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 1838–1845. 20, 21

[41] T. Zhang, B. Ghanem, S. Liu, N. Ahuja, Robust visual tracking via multi-task sparse learning, in: Computer

Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 2042–2049. 20

[42] J. Kwon, K. M. Lee, Tracking by sampling trackers, in: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE, 2011, pp. 1195–1202. 20

[43] B. Liu, J. Huang, L. Yang, C. Kulikowsk, Robust tracking using local sparse appearance model and k-selection, in: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE, 2011, pp. 1313–1320. 20

[44] Y. Wu, B. Shen, H. Ling, Online robust image alignment via iterative convex optimization, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 1808–1814. 20

[45] L. Sevilla-Lara, E. Learned-Miller, Distribution fields for tracking, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 1910–1917. 20

[46] D. Comaniciu, V. Ramesh, P. Meer, Kernel-based object tracking, Pattern Analysis and Machine Intelligence, IEEE Transactions on 25 (5) (2003) 564–577. 20

[47] R. T. Collins, Mean-shift blob tracking through scale space, in: Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on, Vol. 2, IEEE, 2003, pp. II–234. 20

[48] H. Grabner, C. Leistner, H. Bischof, Semi-supervised on-line boosting for robust tracking, in: Computer Vision–ECCV 2008, Springer, 2008, pp. 234–247. 20

[49] S. Stalder, H. Grabner, L. Van Gool, Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition, in: Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 1409–1416. 20

[50] T. B. Dinh, N. Vo, G. Medioni, Context tracker: Exploring supporters and distracters in unconstrained environments, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 1177–1184. 20

[51] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, Vol. 1, IEEE, 2005, pp. 886–893. 23

[52] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, Vol. 1, IEEE, 2001, pp. I–511. 23