A Scale Adaptive Tracking Algorithm Based on Kernel Ridge Regression and Fast Fourier Transform

Lang Zhang(✉), Zhiqiang Hou, Wangsheng Yu, and Wanjun Xu

The Information and Navigation Institute of Air Force Engineering University,
Xi'an 710077, China
zhanglangwy@126.com

Abstract. The change of object's scale is an important reason leading to tracking failure in visual tracking. A scale adaptive tracking algorithm based on kernel ridge regression and Fast Fourier Transform is proposed in this paper. Firstly, the algorithm build regression model using appearance information of object, and then get the position of object in the search region using the regression model. Finally, it estimates the best scale by considering the weight image of all pixels in the candidate region. The experimental results show that the proposed algorithm not only can track the object real time, but also adapt to the changing of object's scale and the interference of background. Compared with the traditional ones, it owns good robustness and efficiency.

Keywords: Visual tracking · Scale adaptive · Kernel ridge regression · Fast Fourier Transform

1 Introduction

Visual tracking is an important task in computer vision [1–3]. Many scholars pay close attention to this research area, because it is widely applied in civil and military fields, such as visual surveillance, intelligent transportation, medial diagnose, military guidance, et al. However, object's scale changing and background's interference are the major reasons leading to tracking failure.

In the past decade, many methods are proposed in visual tracking. The mean shift tracking algorithm [4, 5] is a classical kernel tracking methods, which accomplish the tracking task using object's color histogram. However, for the lack of space information of object and the influence of background feature, the optimal location of object obtained by Bhattacharyya coefficient may not be the exact target location. Zhuang [6] proposes an algorithm which build discriminative sparse similarity map using multiple positive target templates and hundreds of negative templates, and then find the candidate that scores highest in the evaluation model based on discriminative sparse [7] similarity map. However, the algorithm can't get optimal candidate when the tracking environment is complicated. In [8], the author describes target using distribution fields which can alleviate the influence of space information. But there will be drifted or even wrong location when the object's scale and pose change obviously. Ning proposes a

Y.-J. Zhang (Ed.): ICIG 2015, Part I, LNCS 9217, pp. 434–446, 2015.
DOI: 10.1007/978-3-319-21978-3_38

corrected background-weighted histogram algorithm (CBWH) [9] which can reduce background's interference by transforming only the target model but not the target candidate model.

Henriques proposed a Tracking-by-detection algorithm [10] based on circular structure, which establish target model by constructing circular structure matrix. The algorithm not only can effectively alleviate the interference of background, but also can improve tracking efficiency. However, the algorithm has one critical drawback: it is likely to lose the object when the object's scale changes obviously. Considering the advantages and disadvantages of the algorithm, we propose a scale adaptive tracking algorithm based on kernel ridge regression and Fast Fourier Transform. Firstly, the algorithm establish regression model and get object's location in the search region using regression model. Secondly, it presents a method that estimates the best scale by considering the weight image [11] of all pixels in the candidate region.

2 Regression Model

Kernel ridge regression (KRR) [12] is an important algorithm which can solve non-linear problem that can't deal with using linear algorithms in original space. In this part, we will introduce the algorithm theory of KRR and method that improves the solving efficiency based on Fast Fourier Transform.

2.1 Algorithm Theory of KRR

Linear regression is a statistic method that is used to establish relation between two or more variables. A linear regression has the form

$$\boldsymbol{y} = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + \zeta \tag{1}$$

Where, $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$ is the dot product of vectors, ζ is the deviation between true and evaluation.

The loss function is defined as

$$\mathrm{J}(\boldsymbol{w}) = \sum_{i=1}^{N} (y_i - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i - \zeta)^2 + \lambda||\boldsymbol{w}||_2^2 \tag{2}$$

Where, λ controls the amount of regularization.

Trick regression is given by

$$\hat{\boldsymbol{w}}_{ridge} = \boldsymbol{x}^{\mathrm{T}}(\lambda \mathrm{I_N} + \boldsymbol{x}^{\mathrm{T}}\boldsymbol{x})^{-1}\boldsymbol{y} \tag{3}$$

However, trick regression is limited by non-linear problem. It is well known that the kernel trick can improve performance further by allowing a rich high-dimensional feature space. The kernel trick regression is computed as

$$\hat{\boldsymbol{w}}_{kernel} = \boldsymbol{x}^{\mathrm{T}}(\lambda \mathrm{I}_{\mathrm{N}} + K)^{-1}\boldsymbol{y} \tag{4}$$

Note that the input $\boldsymbol{x}$ is mapped to the feature space and K is defined as $K = \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}$. In order to solve conveniently, we defined

$$\boldsymbol{\alpha} = (\lambda I_N + \mathbf{K})^{-1}\mathbf{y} \tag{5}$$

We get

$$\hat{\mathbf{w}} = \mathbf{x}^T\boldsymbol{\alpha} = \sum_{i=1}^{N} \boldsymbol{\alpha}_i \mathbf{x}_i \tag{6}$$

After getting new candidate $\mathbf{x}'$, using the model discussed above, the responding vector of all positions is given by

$$\hat{\mathbf{y}} = \hat{\mathbf{w}}^T\mathbf{x}' = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i^T \mathbf{x}' = \sum_{i=1}^{N} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}') \tag{7}$$

2.2 The Circulate Convolution in Kernel Space

The convolution of two vectors is defined as

$$\boldsymbol{p} * \boldsymbol{q} = \sum_i \boldsymbol{p}_i \boldsymbol{q}_{\tau-i} \tag{8}$$

Where vectors are $\boldsymbol{p} \in R^{\mathrm{M}}$ and $\boldsymbol{q} \in R^{\mathrm{N}}$, and the result is a $(M + N + 1) \times 1$ vector.

We can transform convolution to product by constructing circulate matrix:

$$\boldsymbol{p}' = \begin{cases} p_i, & i \leq \mathrm{M} \\ 0, & \mathrm{M} < i \leq \mathrm{N} + \mathrm{M} - 1 \end{cases} \tag{9}$$

$$\boldsymbol{q}' = \begin{cases} q_i, & i \leq \mathrm{N} \\ 0, & \mathrm{N} < i \leq \mathrm{N} + \mathrm{M} - 1 \end{cases} \tag{10}$$

$$C(\boldsymbol{p}')\boldsymbol{q}'^{\mathrm{T}} = \begin{bmatrix} p_1' & p_{\mathrm{M+N-1}}' & \cdots & p_2' \\ p_2' & p_1' & \cdots & p_3' \\ \vdots & \vdots & \ddots & \vdots \\ p_{\mathrm{M+N-1}}' & p_{\mathrm{M+N-2}}' & \cdots & p_1' \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ 0 \end{bmatrix} \tag{11}$$

According to convolution theorem, we can get:

$$C(\boldsymbol{p}')\boldsymbol{q}' = F^{-1}(F(\boldsymbol{p}')F(\boldsymbol{q}')) \tag{12}$$

Given a single image X, expressed as a $n \times 1$ vector x. A $n \times n$ circulate matrix $C(\boldsymbol{x})$ is obtained:

$$C(\boldsymbol{x}) = [\,\mathrm{E}_0\boldsymbol{x} \quad \mathrm{E}_1\boldsymbol{x} \quad \cdots \quad \mathrm{E}_{n-1}\boldsymbol{x}\,] \tag{13}$$

Where E_i is the permutation matrix that cyclically shifts x by i.

$$\mathbf{E}_i = \begin{bmatrix} \overbrace{0 \quad \cdots \quad 0}^{i} & 1 & \cdots & 0 \\ \vdots \quad \vdots \quad \vdots & \vdots & \vdots & \vdots \\ 0 \quad \cdots \quad 1 & 0 & \cdots & 0 \end{bmatrix}_{\mathrm{n\times n}} \tag{14}$$

The matrix $\mathbf{K}$ with elements $\mathbf{K}_{ij} = \kappa(\mathrm{E}_i\boldsymbol{x}, \mathrm{E}_j\boldsymbol{x})$ which is composed with elements of $C(\boldsymbol{x})$ is circulate [9]. We will define $\mathbf{K}$ with Gaussian kernel and the convolution of kernel space is compute as

$$\begin{aligned} \mathbf{k}_i^{gauss} &= \kappa(\boldsymbol{x}', \mathrm{E}_i\boldsymbol{x}) \\ &= \exp(-\frac{||\boldsymbol{x}'||^2 + ||\boldsymbol{x}||^2 - 2\boldsymbol{x}'^{\mathrm{T}}\mathrm{E}_i\boldsymbol{x}}{2\delta^2}) \end{aligned} \tag{15}$$

As E_i is the permutation matrix, kernel function is given by

$$\begin{aligned} \mathbf{k}^{gauss} &= \exp(-\frac{||\boldsymbol{x}||^2 + ||\boldsymbol{x}'||^2 - 2C(\boldsymbol{x}')\boldsymbol{x}}{2\delta^2}) \\ &= \exp(-\frac{||\boldsymbol{x}||^2 + ||\boldsymbol{x}'||^2 - 2F^{-1}(F(\boldsymbol{x}')F(\boldsymbol{x}))}{2\delta^2}) \end{aligned} \tag{16}$$

We can transform Eq. (5) into frequency domains

$$\boldsymbol{\alpha} = (\lambda I_N + \mathbf{K})^{-1}\mathbf{y} = F^{-1}(\frac{F(\mathbf{y})}{F(\mathbf{K}) + \lambda}) \tag{17}$$

Equation (7) is computed as

$$\hat{\mathbf{y}} = F^{-1}(F(\mathbf{K})F(\boldsymbol{\alpha})) \tag{18}$$

We will get the object location which is corresponding to the best response.

3 Scale Theory

3.1 Mean Shift Algorithm Theory

The mean shift algorithm was widely used in visual tracking. The basic idea is that Bhattacharyya coefficient and other information-theoretic similarity measures are employed to measure the similarity between the target model and the current target region. Generally, the model is represented by normalized histogram vector.

The normalized pixels are denoted by $\{x_i\}_{i=1,2,\ldots,n}$ in the target region, which has n pixels. The probability of a feature u, which is actually one of the m color histogram bins. The target model is computed as

$$\boldsymbol{q}_u = C\sum_{i=1}^{n} k(\|x_i^*\|^2)\delta[b(x_i^*) - u] \tag{19}$$

Where, $b_f : R^2 \rightarrow \{1, \cdots, m\}$ associates the pixels x_0 to the histogram bin, $k()$ is an isotropic kernel profile and is δ the Kronecker delta function, Constant C is an normalization about $\sum\limits_{u=1}^{m} q_u = 1$.

Similarity, the probability of feature in the candidate target model is given by

$$\boldsymbol{p}_u(y) = C_h\sum_{i=1}^{n_h} k(\left\|\frac{y - x_i}{h}\right\|^2)\delta[b(x_i) - u] \tag{20}$$

Where y is the center of the target candidate region.

The similarity between target model and candidate target model is measured by Bhattacharyya coefficient which is given by

$$\rho(y) \equiv \rho[\boldsymbol{p}_u(y), \boldsymbol{q}_u] = \sum_{u=1}^{m} \sqrt{\boldsymbol{p}_u(y)\boldsymbol{q}_u} \tag{21}$$

In order to find the best location of the object, a key procedure is the computation of an offset form current location y_0 to the new location y_1 according to the iteration equation:

$$y_1 = \frac{\sum\limits_{i=1}^{n_h} x_i w_i g\left(\|(y_0 - x_i)/h\|^2\right)}{\sum\limits_{i=1}^{n_h} w_i g\left(\|(y_0 - x_i)/h\|^2\right)} \tag{22}$$

Where, $w_i = \sum\limits_{u=1}^{m} \sqrt{\frac{q_u}{p_u(y_0)}}\sigma[b(x_i) - u], g(x) = -k'(x)$

3.2 Estimates the Best Scale of the Object

In this section, we will propose a convenient algorithm which is based on [11] to estimate the best scale of the object. The algorithm can precisely estimate object's scale by utilising the zeroth-order moment of the weight image of all pixels in the target candidate region and the similarity between target model and target candidate model.

The changing of target is usually a gradual process in the sequential frames. Thus we can assume that the changing of target's scale and location is smooth and this assumption owes reasonably well in most video sequences. With the assumption, we will track the target in a larger candidate region than its size to ensure that the target is in this candidate region based on the area of the target in the previous frame. The weight image is defined as computing the weight of every pixels, which is the square root of the ratio of its colour probability in the target model to its colour probability in the target candidate model. For a pixel x_i in the target candidate region, its weight is given by

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{q_u}{p_u(y_0)}}\sigma[b(x_i) - u] \tag{23}$$

The weight value of every pixel represents the possibility that it belongs to the object. So the weight image can be regarded as the density distribution function of the object in the target candidate region. Figure 1 shows the weight image of target candidate region with different scale of target.

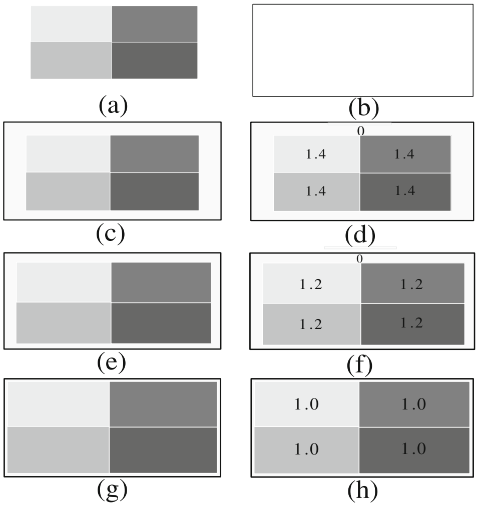


Fig. 1. The weight image of target candidate region with different scale of target

Figure 1a shows a target consisted with four grey levels. Figure 1b represents the candidate region that is larger than the target. Figure 1c, e, g and d, f, h respectively illustrate the target candidate region with different scale of object and corresponding weight images calculated by our algorithm.

Form Fig. 1, we can see clearly that the weight image will change dynamically with the changing of object's scale. Particularly, the weight image is closely related to the object's change. The closer the real scale of the object is to the candidate region, the better the weight approaches to 1. Based on those properties, the sum of the weights of all pixels, that is, the zeroth-order moment is considered as the predetermination which reflect the scale of the target.

$$\mathrm{M}_0 = \sum_{i=1}^{n} w(x_i) \tag{24}$$

However, owing to the existence of the background pixels, the probability of the target features is less than that in the target model. So (23) will enlarge the weights of target pixels and restrain the weights of background pixels. Thus, the target pixels will contribute more to target area estimation, whereas the background pixels contribute less. This can be seen in Fig. 1d, f and h. On the other hand, the Bhattacharyya coefficient is used to represent the similarity between the target model and the target candidate model. A smaller Bhattacharyya coefficient means that there are more background features and fewer target features in the target candidate region. If we use M_0 as the estimation of the target area, then according to (24), when the weights form the target become bigger, the estimation error by using M_0 as the evaluation of target scale will be bigger, vice versa. Therefore, the Bhattacharyya coefficient is a good indicator of how reliable it is by using M_0 as the target area. That is to say, with the increase of the Bhattacharyya coefficient, the estimation accuracy will also increase. So the Bhattacharyya coefficient is used to adjust M_0 in estimating the target area. The estimating equation is

$$\mathrm{A} = c(\rho)\mathrm{M}_0 \tag{25}$$

Where, $c(\rho)$ is a monotonically increasing function with respect to the Bhattacharyya coefficient $\rho(0 \le \rho \le 1)$. $c(\rho)$ is defined as

$$c(\rho) = \exp\left(\frac{\rho - 1}{\sigma_1}\right) \tag{26}$$

where σ_1 is a constant. From (25) and (26), we can get that when the target candidate model approaches to the target model, that is, when ρ approaches to 1, $c(\rho)$ approaches to 1 and in this case it is reliable to use M_0 as the estimation of target scale. When the candidate model is not identical to the target model, that is, when ρ decreases, M_0 will be much bigger than the target scale, but $c(\rho)$ is less than 1. So A can avoid being biased too much from the real target scale. The experiment show that setting σ_1 between 1 and 2 can achieve accurate estimation of target scale [11].

In this paper, target's state is showed by rectangle and the ratio of length to wide is constant K. With the estimation of target scale, the length and wide of target is given by

$$h = \sqrt{\mathrm{KA}}, w = \sqrt{{}^{\mathrm{A}}\!/_{\mathrm{K}}} \tag{27}$$

4 Model Update Strategy

Model update is an important step in visual tracking, which is help to avoid the drifting of model. In order to adapt the change of target and alleviate the interference of background, we update the regression model and histogram model of target.

Regression Model Update: The target's regression model is updated every frame by combining the fixed reference model extracted from previous frame and the result form the most recent frame with an update weight β. The update equation is

$$\hat{\boldsymbol{w}}_{new} = \beta\hat{\boldsymbol{w}}_{cur} + (1-\beta)\hat{\boldsymbol{w}}_{old} \tag{28}$$

Where $\hat{\boldsymbol{w}}_{new}$ is new model, $\hat{\boldsymbol{w}}_{old}$ is the old model used by last frame, $\hat{\boldsymbol{w}}_{cur}$ is the current model obtained from current frame.

Histogram Model Update: We set a threshold ρ' for the target's histogram model update. Then we analyze the Bhattacharyya coefficient to determine when to update the histogram model:

(1) if $\rho \geq \rho'$, the target's histogram have no significant changes, so don't update the histogram model.

(2) if $\rho < \rho'$, the target's histogram may changes greatly, so we update the histogram model with a update weight α. The update equation is

$$q_{u(new)} = \alpha p_{u(cur)} + (1-\alpha)q_{u(old)} \tag{29}$$

Where $q_{u(new)}$ is the new histogram, $q_{u(old)}$ is the previous histogram model, $p_{u(cur)}$ is the current histogram model obtained from current frame.

5 The Proposed Algorithm

5.1 The Basic Theory

In order to improve robustness of tracking algorithm, we propose a scale adaptive tracking algorithm based on kernel ridge regression and Fast Fourier Transform. Firstly, the algorithm establish regression model and search object's location using regression model. Secondly, the paper presents a method that estimates the best scale of target by considering the weight image.

5.2 Algorithm Step

The proposed tracking algorithm is summarized as follows:

Step 1: Initialize the target's state and obtain the training data. Calculate the regression model by (6) and (17) and the target histogram model $\boldsymbol{q}_u$ by (19)

Step 2: Search object's location $\hat{y}$ by (18)

Step 3: Calculate the target candidate histogram model $\boldsymbol{p}_u(\hat{y})$ by (20) and the Bhattacharyya coefficient ρ by (21)

Step 4: Calculate the weight image of the candidate region by (23)

Step 5: Estimates the best scale A of target by (25) and Calculate the length h and wide w by (28). Then get the tracking result of current frame

Step 6: Calculate the regression model with new training data and update the regression model by (28)

Step 7: Synthetically analyze the Bhattacharyya coefficient ρ and threshold ρ', determine if it is necessary to update the histogram model by (29), and continue the tracking for the next frame

5.3 Algorithm Flow

The whole flow chart is presented in Fig. 2.

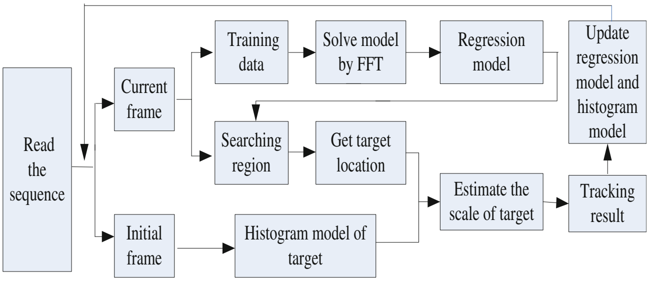
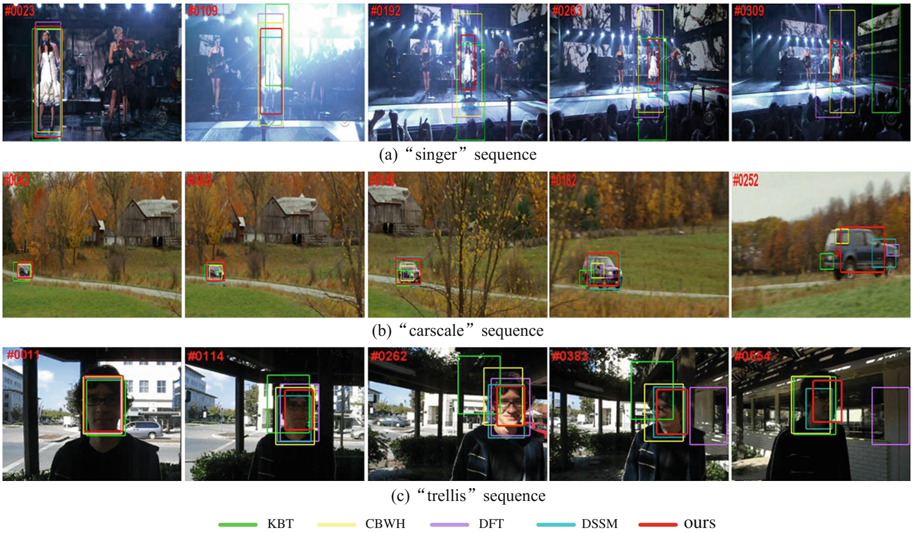


Fig. 2. Flow chart of the our tacking algorithm

6 Experimental Results and Discussions

Several representative sequences are used to compare the proposed algorithm (Ours) with the kernel-based tracker (KBT) [4], the CBWH [9] based on corrected background-weighted histogram, the DFT [8] based on distribution fields and the DSSM [6] based on discriminative sparse similarity map. For all the sequences, we select a four times bigger region centered with the object as the candidate region of our algorithm. The regularization parameter λ is set at 0.01. The update weight β of regression model is set at 0.075 and the update weight α of histogram model is set at 0.1. Note that both of α and β is getting from experiences. The experimental experiences show that setting scale factor σ_1 between 1 and 2 can achieve accurate estimation of target scale. All the experiments are implemented under the 2.6 GHz PC with 2 GB memory and the programming environment is MATLAB R2009a.

The first experiment is on the singer sequence, which has obviously scale shrinking and the interference of background. Figure 3(a) shows the tracking results of the sequences. We can see, as the background interference, KBT and DFT have large tracking error in the frame of 192 and 262. CBWH can decrease the distraction of background, but with the algorithm can't adapt the changes of target's scale, tracking result is also bad. Although DSSM can track target successfully, our algorithm has better result in estimating target's location and adapting the changing of target' scale (263 and 309 frame).



(a) "singer" sequence

(b) "carscale" sequence

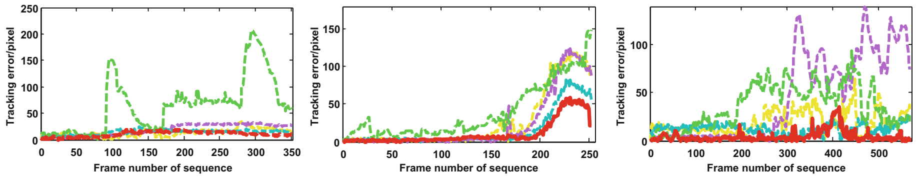
(c) "trellis" sequence

Fig. 3. Qualitative comparison of the tracking algorithms

The second experiment is a challenging video sequence, which has acutely scale enlarging, pose changing and local shelter. Figure 3(b) shows the tracking results of the sequences. Since CBWH and DFT lack necessary scale adapt mechanism, they can't adapt the obvious change of scale. KBT and DSSM have large tracking error after target's pose changing obviously along with acutely enlarging of scale (182 and 252 frame). While our tracking algorithm can get the target location and scale with acceptable precision.

The third video is the Trellis sequence. In this video sequence, complicated background interference and scale changing are the major challenge. The tracking results is showed in Fig. 3(c). Along with the scale changing of target, the background distracts severely, and both KBT, DFT and CBWH have obviously tracking deviation. DSSM and our algorithm can track target successfully.

In order to quantitatively analyse the real-time performance of our algorithm and referenced algorithms, we use the average running time to compare the efficiency in the



(a) “singer” sequence (b) “carscale” sequence (c) “trellis” sequence

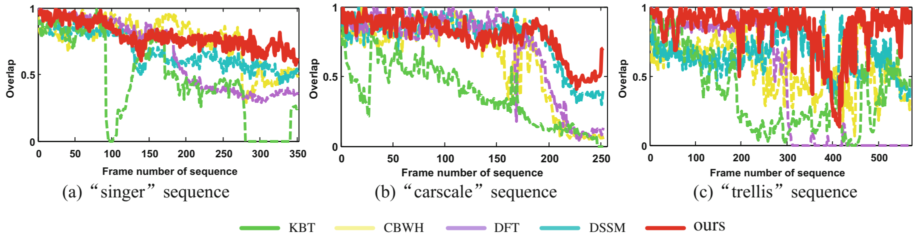
Fig. 4. Center location error curves

above video sequences. Table 1 shows the comparison results of the average running time of per frame, which is get by running all the algorithms with same configuration and using the same dataset for fair comparison. The results proves that the proposed algorithm is best in real-time performance.

Table 1. Average running time of per frame

	KBT	CBWH	DFT	DSST	OURS
Carscale	0.128	0.193	0.501	1.874	0.060
Trellis	0.142	0.166	0.467	1.722	0.043
Singer	0.196	0.402	0.622	1.783	0.081

In order to analyse the performance of tracking accuracy, we use the center location error and overlap to compare the accuracy in the above video sequences. It should be noted that a smaller center location error and a bigger overlap rate implies a more accurate result. The center location error is the Euclidean distance between center location and the ground truth. Figure 4 shows the center location error curves of the trackers. The overlap rate reflects the covering level of tracking result and the corresponding ground truth. The overlap rate curves of our algorithm and compared algorithms is showed in Fig. 5. Tables 2 and 3 shows the average center location errors and the average overlap rates respectively. As shown in the table, the proposed algorithm exceeds the referenced algorithms.



(a) “singer” sequence (b) “carscale” sequence (c) “trellis” sequence

Fig. 5. Overlap rate curves

Table 2. Average center location error

	KBT	CBWH	DFT	DSST	OURS
Carscale	38.809	26.337	23.239	15.192	11.079
Trellis	37.005	19.074	44.808	10.326	7.226
Singer	65.870	12.765	18.882	14.023	10.482

Table 3. Average overlap rate

	KBT	CBWH	DFT	DSST	OURS
Carscale	0.297	0.598	0.700	0.773	0.789
Trellis	0.369	0.573	0.455	0.748	0.813
Singer	0.462	0.749	0.642	0.671	0.804

7 Conclusions

In conclusion, for the drawback of traditional regression tracking algorithm, we proposed a scale adaptive tracking algorithm, which build target's model based on kernel ridge regression and evaluate target's scale using weight image. First, the algorithm establish regression model and get object's location in the search region using regression model. In order to improve the tracking efficiency, we transform the operations of time domain into frequency domain by using Fast Fourier Transform. Second, the paper present a method that estimates the best scale of target by considering the weight image of all pixels in the candidate region. The experimental results and quantitative evaluation demonstrate that the proposed algorithm owns good accuracy and efficiency.

Acknowledgment. This work is supported by Nation Natural Science Foundation of China (No. 61175029, 61473309).

References

1. Yang, H.X., Shao, L., Zheng, F., et al.: Recent advances and trends in visual tracking: a review. Neurocomputing **74**, 3823–3831 (2011)
2. Hou, Z.Q., Han, C.Z.: A survey of visual tracking. Acta Automatica Sin. **32**(4), 603–617 (2006)
3. Smeulders, A., Chu, D., Cucchiara, R., et al.: Visual tracking: an experimental survey. IEEE Trans. Pattern Anal. Mach. Intell. (2013) (Epub ahead of print)
4. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. IEEE Trans. Pattern Anal. Mach. Intell. **25**(5), 564–577 (2003)
5. Collins, R.T.: Mean-shift blob tracking through scale space. In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), Madison, WI, USA, pp. 234–240. IEEE (2003)
6. Zhuang, B.H., Lu, H.C., Xiao, Z.Y., et al.: Visual tracking via discriminative sparse similarity map. IEEE Trans. Image Process. **23**(4), 1872–1881 (2014)

7. Wang, D., Lu, H.C., Yang, M.: Online object tracking with sparse prototypes. IEEE Trans. Image Process **22**(1), 314–325 (2013)
8. Sevilla, L., Learned, E.: Tracking with distribution fields. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA. IEEE (2012)
9. Ning, J.F, Zhang, L., Zhang, D., et al.: Robust mean shift tracking with corrected background-weighted histogram. IET Comput. Vis. (2010)
10. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 702–715. Springer, Heidelberg (2012)
11. Ning, J.F., Zhang, L., Zhang, D., et al.: Scale and orientation adaptive mean shift tracking. IET Comput. Vis. **6**(1), 52–61 (2012)
12. Li, Q., Shao, C.: Nonlinear systems identification based on kernel ridge regression and its application. J. Syst. Simul. **21**(8), 2152–2155 (2009)